TIMEFLEX: APLICACIÓN WEB PARA PLANIFICACIÓN DE TURNOS DE TRABAJO

TIMEFLEX: WEB APPLICATION FOR SHIFT PLANNING

TRABAJO FIN DE GRADO
CURSO 2024-2025

AUTORES
ANDRÉS MARÍ PIQUERAS
GABRIEL FERNÁNDEZ SACRISTÁN
ALEJANDRO LÓPEZ LÓPEZ DE LA COVA
ÁLVARO JUAN MARTÍN SÁNCHEZ-MONTAÑEZ

DIRECTOR
PABLO GORDILLO ALGUACIL

GRADO EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

TIMEFLEX: APLICACIÓN WEB PARA PLANIFICACIÓN DE TURNOS DE TRABAJO
TIMEFLEX: WEB APPLICATION FOR SHIFT PLANNING

TRABAJO DE FIN DE GRADO EN INGENIERÍA INFORMÁTICA

AUTORES

ANDRÉS MARÍ PIQUERAS: **SOBRESALIENTE (9,5)**

GABRIEL FERNÁNDEZ SACRISTÁN: **SOBRESALIENTE (9,5)**

ALEJANDRO LÓPEZ LÓPEZ DE LA COVA: **SOBRESALIENTE (9,5)**

ÁLVARO JUAN MARTÍN SÁNCHEZ-MONTAÑEZ: **SOBRESALIENTE (9,5)**

DIRECTOR

PABLO GORDILLO ALGUACIL

CONVOCATORIA: JUNIO 2025

GRADO EN INGENIERÍA INFORMÁTICA

FACULTAD DE INFORMÁTICA

UNIVERSIDAD COMPLUTENSE DE MADRID

6 DE JUNIO DE 2025

DEDICATORIA

A nuestras familias, que nos han apoyado y han estado a nuestro lado en cada paso del proceso, a nuestros amigos, que nos han echado una mano cuando lo hemos necesitado y nos han facilitado siempre obtener datos o cualquier otra información que nos sirviera de ayuda, y finalmente, al resto de compañeros del equipo por su implicación en el proyecto y principalmente por los años de los que hemos disfrutado juntos en esta facultad y fuera de ella.

AGRADECIMIENTOS

Queremos agradecer a Pablo Gordillo, nuestro tutor, por su implicación y guía en el desarrollo del proyecto, así como el tiempo empleado en las reuniones de seguimiento. Siempre ha estado ahí para echarnos una mano cuando lo hemos necesitado, al igual que Eugenio Pablo Concepción, otro excelente profesor de esta facultad. Nuestra experiencia como alumnos suyos nos ha permitido apreciar que desde el primer momento se implica al máximo en su pasión, la docencia, y no ha dudado nunca en ayudarnos o lanzarnos recomendaciones siempre que se lo hemos solicitado. Gracias a ambos.

RESUMEN

TimeFlex: Aplicación Web para planificación de turnos de trabajo

En la actualidad, existen problemas incrementales que afectan a las empresas a la hora de gestionar de forma eficiente los horarios y turnos de sus empleados, especialmente a aquellas en las que estos elementos desempeñan un papel clave: sanidad, docencia y hostelería. En algunos de estos ámbitos, la generación de estos turnos de trabajo se sigue realizando de manera manual, con el consiguiente desperdicio de recursos tanto en tiempo como en empleados.

En este contexto, TimeFlex surge como una solución que permite gestionar los horarios, turnos de trabajo y vacaciones de los empleados de las organizaciones. TimeFlex ofrece una aplicación web amigable con el usuario que permite generar los turnos y horarios de forma automática, en función de las solicitudes de los empleados y maximizando su satisfacción. Esto ofrece mejoras tanto a los empleados de las organizaciones, ofreciéndoles una interfaz web a través de la cual gestionar sus solicitudes, como a los empleados de recursos humanos (RRHH), permitiendo generar y optimizar de forma automática los horarios y vacaciones de los empleados.

TimeFlex está basado en PHP y el *framework* Laravel, además de SQLite y JavaScript con bibliotecas derivadas. Este proyecto aborda las numerosas casuísticas implicadas en los diferentes casos de uso de estas industrias con el firme propósito de proveer una aplicación web fiable y útil que aporte valor a las empresas haciendo uso de programación con restricciones.

Palabras clave

Optimización, Restricciones, Turnos, Horarios, Satisfacción, Vacaciones, Recursos Humanos, Solicitudes

ABSTRACT

TimeFlex: Web Application for Shift Planning

Nowadays, there are incremental issues affecting companies when it comes to efficiently managing employee schedules and shifts, especially those in which these elements play a key role: healthcare, education and hospitality. In some of these areas, the generation of these work shifts is still being done manually, with the consequent waste of resources in terms of both time and employees.

In this context, TimeFlex emerges as a solution to manage schedules, work shifts and holidays of employees within organizations. TimeFlex offers a user-friendly web application that enables automatic generation of shifts and schedules, based on employees request and maximizing employee satisfaction. This offers improvements both to employees of organizations, offering them a web interface through which to manage their requests, and to Human Resources (HR) employees, allowing them to automatically generate and optimize workers schedules and vacations.

TimeFlex is based on PHP and the Laravel framework, as well as SQLite and JavaScript with derived libraries. This project addresses the numerous casuistries involved in the different use cases of these industries with the firm purpose of providing a reliable and useful web application that brings value to companies using constraint programming.

Keywords

Optimization, Constraints, Shifts, Schedules, Satisfaction, Holidays, Vacations, Human Resources, Requests

ÍNDICE DE CONTENIDOS

Capítulo 1 - Introducción 1

1.1 Motivación 1

1.2 Objetivos 2

1.3 Plan de trabajo 3

1.4 Organización del documento 4

Capítulo 2 - TimeFlex: Automatización de turnos de trabajo 7

2.1 Descripción del producto 7

2.2 Usuarios objetivo 7

2.3 Análisis del mercado y la competencia 9

Capítulo 3 - Análisis previo y metodología 15

3.1 Investigación con usuarios 15

3.2 Análisis cualitativo y cuantitativo 18

3.3 Resultados y conclusiones de la investigación 19

3.3.1 Empleados 20

3.3.2 Gerentes de RRHH 21

3.3.3 Conclusiones comunes 21

3.4 Caracterización de los usuarios 22

3.5 Escenarios de uso 23

3.6 Customer Journey Maps 24

Capítulo 4 - Diseño de la aplicación 27

4.1 Arquitectura del sistema 27

4.1.1 Capa de presentación y lógica de negocio 30

4.1.2 Capa de servicios 44
4.2 Estructura de la base de datos 47
Capítulo 5 - Descripción funcional de los módulos 57
5.1 Calendarios 57
5.2 Formularios 60
5.3 Estadísticas 63
5.4 Notificaciones 65
Capítulo 6 - Implementación y desarrollo 67
6.1 Capa de negocio y presentación 67
6.1.1 Calendarios 67
6.1.2 Formularios 70
6.1.3 Estadísticas 72
6.1.4 Notificaciones 74
6.2 Servicio de optimización 76
6.3 Seguridad 84
6.4 Gestión y verificación 87
6.4.1 Sistema de logs y trazabilidad 87
6.4.2 Datos de pruebas 87
Capítulo 7 - Instalación y manual de uso 89
7.1 Instalación 89
7.2 Manual de uso 91
7.2.1 Manual de uso para Gerente de RRHH 91
7.2.2 Manual de uso para Empleado 113
Capítulo 8 - Conclusiones 123
Capítulo 9 - Futuras líneas de trabajo 127

Bibliografía.....131
Anexos.....141
Apéndice A - Contribuciones personales.....141
Apéndice B - Introduction.....149
Apéndice C - Conclusions and Future Work.....153
Apéndice D - Política de privacidad.....159
Apéndice E - Encuestas empleadas en la fase de investigación.....161

ÍNDICE DE FIGURAS

Figura 2-1. Aspecto visual de la interfaz de UniTime.....12

Figura 4-1. Diagrama de la arquitectura.....27

Figura 4-2. Diagrama de la interfaz web.....28

Figura 4-3. Diagrama de la capa de presentación y lógica de negocio.....28

Figura 4-4. Diagrama del microservicio en FastAPI.....29

Figura 4-5. Diagrama de los servicios externos.....30

Figura 4-6. Gráfico comparativo respecto al interés despertado por frameworks de PHP. 32

Figura 4-7. Comparación entre el rendimiento de distintos frameworks de PHP (fuente Kinsta).....35

Figura 4-8. Toggles interactivos para las preferencias de notificaciones.....37

Figura 4-9. Página de preguntas frecuentes utilizando un diseño adaptado de Tailwind CSS. 38

Figura 4-10. Diagrama completo de la base de datos. 55

Figura 6-1. Flujo del módulo de calendarios. 70

Figura 6-2. Flujo del módulo de formularios.....72

Figura 6-3. Flujo del módulo de estadísticas.....74

Figura 6-4. Icono indicando nuevas notificaciones. 75

Figura 6-5. Flujo del módulo de notificaciones. 76

Figura 6-6. Ejemplo de flujo de petición a microservicio. 77

Figura 6-7. Ejemplo de flujo de respuesta desde microservicio.....78

Figura 7-1. Pantalla de inicio de sesión.....92

Figura 7-2. Panel de administración.....93

Figura 7-3. Registro de empleado.....95

Figura 7-4. Edición de empleado. 96
Figura 7-5. Información y estadísticas de empleado. 97
Figura 7-6. Vista de empleados. 98
Figura 7-7. Opciones para secciones. 99
Figura 7-8. Registro de sección. 100
Figura 7-9. Edición de sección. 100
Figura 7-10. Vista principal de administración de horarios. 101
Figura 7-11. Registro de horario. 102
Figura 7-12. Edición de horario 103
Figura 7-13. Registro de tipos de turno. 104
Figura 7-14. Cambio de turno desde administración. 105
Figura 7-15. Horario específico de sección. 106
Figura 7-16. Asignación de turnos manual. 107
Figura 7-17. Estadísticas de horario. 108
Figura 7-18. Vista de administración de formularios de preferencia. 109
Figura 7-19. Registro de formulario de preferencia. 110
Figura 7-20. Edición de formulario de preferencia. 111
Figura 7-21. Respuestas de formularios de preferencia. 112
Figura 7-22. Estadísticas generales de la empresa. 113
Figura 7-23. Vista de panel de usuario. 114
Figura 7-24. Estadísticas personales y de equipo. 115
Figura 7-25. Evolución de satisfacción personal y de equipo. 115
Figura 7-26. Información sobre compañeros. 116
Figura 7-27. Vista principal de horarios. 117
Figura 7-28. Horario de equipo. 117

Figura 7-29. Horario personal. 118
Figura 7-30. Estadísticas del horario de equipo.....118
Figura 7-31. Solicitud para cambio de turno con un compañero. 119
Figura 7-32. Vista principal de formularios de preferencia. 120
Figura 7-33. Formulario de preferencia a responder. 120
Figura 7-34. Vista de equipo. 121
Figura 7-35. Vista de ayuda y preguntas frecuentes para usuarios. 122

ÍNDICE DE TABLAS

Tabla 2-1. Comparación entre las distintas herramientas de optimización de turnos. 13

Tabla 4-1. Comparativa entre distintos frameworks de PHP. 33

Tabla 4-2. Comparativa entre distintas herramientas de visualización de estadísticas. ..40

Tabla 4-3. Comparativa entre distintos frameworks de desarrollo de APIs. 46

Tabla 4-4. Tabla users y sus relaciones. 49

Tabla 4-5. Tabla sections y sus relaciones. 50

Tabla 4-6. Tabla forms y sus relaciones. 51

Tabla 4-7. Tabla questions y sus relaciones. 52

Tabla 4-8. Tabla schedules y sus relaciones. 53

Tabla 4-9. Tabla shifts y sus relaciones. 54

Capítulo 1 - Introducción

Este capítulo introduce y presenta el proyecto de aplicación web TimeFlex, cuyo objetivo es optimizar la gestión de turnos, horarios y asignación de vacaciones en empresas y organizaciones.

La Sección 1.1 describe la principal motivación que hay detrás del presente proyecto, así como el problema que se pretende afrontar, la Sección 1.2 presenta los principales objetivos de este proyecto, la Sección 1.3 contiene el plan de trabajo seguido durante su ejecución y la Sección 1.4 detalla la organización del documento.

1.1 Motivación

Los turnos y horarios son elementos clave de numerosos sectores como la docencia, la sanidad o la restauración. Sin embargo, existe un descontento generalizado entre los empleados debido a los ineficientes métodos actuales para asignar y gestionar estos turnos. En algunos casos, esta asignación se realiza incluso a papel, de forma manual, evidenciando un alto contraste con la oleada tecnológica actual y dificultando el uso de métricas objetivas a la hora de generar estos horarios.

Además, los trabajadores de recursos humanos (RRHH) se enfrentan a una larga lista de restricciones y preferencias cada vez que tratan de configurar un horario. Intentar cuadrar todas ellas supone un gran esfuerzo en tiempo y recursos, además de no poder garantizar que la solución encontrada sea la más satisfactoria para todos los empleados. Esto puede resultar frustrante tanto para los empleados de este departamento como para el resto de los trabajadores de la empresa.

Las entrevistas realizadas tanto con trabajadores de hospitales y centros de salud, así como profesores y empleados del sector hostelero han servido para verificar los hechos anteriores y añadir nuevas conclusiones. Por un lado, se necesita un sistema de asignación de turnos que priorice la satisfacción de los empleados, basado en criterios lo más objetivos posibles. Por otro lado, los trabajadores requieren una aplicación tecnológica con la que puedan interactuar de forma intuitiva y realizar acciones que mejoren su experiencia laboral como solicitar cambios de turno o vacaciones.

1.2 Objetivos

Como solución a los problemas descritos anteriormente, emerge la idea del proyecto TimeFlex, una aplicación web mediante la cual los empleados de RRHH pueden generar horarios, turnos y asignación de vacaciones. El principal objetivo es desarrollar una herramienta que contenga distintos módulos que permitan tanto una gestión eficiente de cambios de turno y otras peticiones de los empleados, así como la generación de formularios para llevar a cabo las solicitudes por parte de los empleados. Además, la herramienta incluirá un módulo de estadísticas que permita presentar de forma visual el resultado de las asignaciones, así como métricas agrupadas por distintos grupos y sectores. También se persigue ofrecer funcionalidades asociadas a las distintas solicitudes desde el punto de vista tanto del encargado de RRHH como de usuarios normales, gestionando así los casos de uso de los diferentes roles para ofrecer un servicio completo a las organizaciones. Por último, se persigue ofrecer un diseño cuidado, con una interfaz amigable e intuitiva para el usuario.

Para lograr este objetivo, se definieron los siguientes pasos:

- Ofrecer una solución automática para el departamento de RRHH a la hora de generar horarios y asignar turnos de trabajo dentro de las organizaciones.
- Definición de métricas y objetivos que permitan evaluar la satisfacción de los empleados.
- Garantizar que los horarios y turnos generados maximicen la satisfacción de los empleados involucrados.
- Diseñar una interfaz intuitiva e interactiva que ofrezca una buena experiencia de usuario.
- Implementar funcionalidades que permitan gestionar las tareas relacionadas con los turnos de trabajo y libranzas. Entre ellas destacan la petición de cambios de turno, la visualización e interacción con estadísticas y *dashboards*, y la generación de formularios para conocer las opiniones de los trabajadores.
- Ofrecer un sistema de notificaciones para comunicar cambios y gestiones con opciones de personalización a disposición del usuario.

1.3 Plan de trabajo

El plan de trabajo seguido para el desarrollo del proyecto TimeFlex se compone de seis fases ordenadas cronológicamente:

Fase 1 - Estudio y Análisis de las necesidades

Mediante la realización de entrevistas personales y cuestionarios a trabajadores de los sectores de interés, se identificaron las principales necesidades y problemas de los empleados. De esta forma, se definieron los objetivos de la aplicación, descritos en la sección anterior.

Fase 2 - Diseño de la arquitectura del sistema

Teniendo en cuenta los requisitos recopilados en la fase anterior, se estudiaron las herramientas y arquitecturas más adecuadas para el desarrollo del proyecto. Esto incluye el diseño de la interfaz, la estructura de la base de datos y las distintas capas de la aplicación para cumplir con los objetivos propuestos. Además, se estudiaron productos software con enfoques similares a TimeFlex.

Fase 3 - Desarrollo del *frontend* y *backend*

En esta fase, se llevó a cabo la implementación de las funcionalidades básicas de la aplicación como el inicio de sesión, la gestión de roles o el sistema de notificaciones. Además, se diseñó la interfaz y se codificaron la mayoría de las funcionalidades avanzadas como la creación de formularios, la generación de estadísticas o la solicitud de cambios de turno, con la lógica adecuada para la transcripción de la información a la base de datos.

Fase 4 - Optimización de turnos mediante API

La incorporación de un sistema de resolución de restricciones especializado en la asignación de turnos y horarios se llevó a cabo en esta etapa. Utilizando una API y herramientas como el resolutor de restricciones Z3 [76] [87], desarrollado por Microsoft, se automatizó la asignación tanto de turnos de trabajo como de vacaciones.

Fase 5 - Validación y pruebas

Se realizaron pruebas sobre los distintos casos de uso y funcionalidades implementadas, asegurando su correcto funcionamiento y garantizando una correcta experiencia de usuario.

Fase 6 - Documentación y ajustes finales

La documentación asociada a la aplicación web se elaboró en esta etapa, detallando toda la información relevante sobre el proyecto.

Las fases 3, 4 y 5 descritas anteriormente siguieron un proceso iterativo, en el que se incorporó la retroalimentación obtenida de manera progresiva para ir construyendo las distintas versiones y prototipos de la aplicación.

El resultado final obtenido tras seguir estos pasos es una aplicación web que cumple con los requisitos definidos inicialmente. Se ha desarrollado una herramienta que automatiza de forma eficiente tanto las peticiones de cambios de turno como la asignación de horarios y vacaciones, ofreciendo interfaces personalizadas para los encargados de RRHH y para los usuarios normales, adaptadas a sus distintos casos de uso. Se ha hecho hincapié en lograr que la interfaz sea intuitiva y fácil de usar para los usuarios.

1.4 Organización del documento

El presente documento está organizado como sigue: el Capítulo 2 describe las principales funcionalidades ofrecidas por TimeFlex, profundizando en los usuarios objetivos y realizando un estudio de la competencia. El Capítulo 3 presenta la investigación llevada a cabo con usuarios mediante entrevistas y encuestas, así como los resultados obtenidos. Además, caracteriza a los usuarios, así como a sus escenarios de uso y su recorrido. El Capítulo 4 detalla la arquitectura empleada en el proyecto y las distintas capas involucradas. También contiene la estructura de la base de datos utilizada. El Capítulo 5 contiene información detallada acerca de las diferentes funcionalidades de cada uno de los módulos de la aplicación. El Capítulo 6 describe la implementación técnica y herramientas utilizadas tanto en los distintos módulos de la aplicación como en el servicio de optimización. Adicionalmente, incluye la gestión de

la seguridad y aspectos técnicos relativos al entorno de ejecución, la usabilidad y el rendimiento de la aplicación. El Capítulo 7 contiene los pasos a seguir para la correcta instalación del proyecto TimeFlex, además de manuales de uso de la herramienta. El Capítulo 8 contiene las conclusiones y el Capítulo 9 las futuras líneas de trabajo.

Capítulo 2 - TimeFlex: Automatización de turnos de trabajo

Este capítulo presenta la herramienta desarrollada, así como las principales funcionalidades de esta. La Sección 2.1 describe el producto, la Sección 2.2 define a los usuarios objetivo de la aplicación, y finalmente, la Sección 2.3 contiene un estudio del mercado y la competencia, profundizando en herramientas con un enfoque similar a TimeFlex.

2.1 Descripción del producto

El proyecto ***TimeFlex*** tiene como objetivo el desarrollo de una aplicación integral para la gestión eficiente de los turnos laborales y las solicitudes de vacaciones de los empleados de una empresa de forma automática. La aplicación dispone de formularios que pueden rellenar los empleados con sus preferencias de horarios.

Estos formularios son utilizados para generar los horarios individualizados de todos los trabajadores de la empresa de forma automática. Además, durante la generación de turnos, se intenta satisfacer las necesidades de los empleados introducidas previamente en los formularios.

A su vez, los empleados también dispondrán de otro formulario para introducir las fechas de vacaciones de las cuales les gustaría disponer y al igual que con los horarios, el equipo de RRHH generará automáticamente las vacaciones de todos los integrantes de la empresa en base a sus preferencias. Finalmente, los empleados rasos podrán, además, consultar su horario de trabajo, realizar solicitudes de cambios, recibir notificaciones personalizadas y proporcionar retroalimentación sobre sus horarios y turnos.

2.2 Usuarios objetivo

TimeFlex distingue dos principales roles de usuarios objetivos: los empleados de la empresa, y los encargados de elaborar la asignación de turnos de trabajo (personal de RRHH en adelante):

- **Empleados de la empresa**: Usuarios que utilizan la aplicación para gestionar sus horarios, turnos y vacaciones. TimeFlex proporciona a estos usuarios una herramienta intuitiva y flexible que les permita optimizar su tiempo laboral y mejorar su satisfacción en el trabajo. Los principales objetivos identificados con este rol son:
    - Tener un medio sencillo a través del cual indicar sus preferencias de horarios, turnos de trabajo y vacaciones.
    - Recibir de forma rápida y clara los turnos que le han sido asignados, a través de un sistema de notificaciones.
    - Tener un sistema o canal a través del cual el usuario pueda enviar sus opiniones o una retroalimentación sobre el calendario de turnos.
    - Opciones de visualización para los horarios y turnos asignados.
- **Personal de RRHH**: Agrupa a los usuarios encargados de gestionar los turnos, vacaciones y horarios de los empleados. Para ello tiene que equilibrar las necesidades de la empresa en cuanto a número de trabajadores por turnos, con las preferencias de los empleados. Estos usuarios necesitan una solución eficiente que reduzca la carga administrativa y minimice los errores en la planificación de turnos.

  Los principales objetivos identificados con este rol son:
    - Diseñar formularios personalizados para recopilar las preferencias de turnos de los empleados.
    - Automatizar la asignación de turnos, teniendo en cuenta las preferencias de los empleados y las necesidades operativas de la empresa.
    - Optimizar el proceso para que sea eficiente y cumpla con las normativas laborales para así poder evitar confusiones y errores.
    - Enviar notificaciones claras y personalizadas sobre los turnos asignados.
    - Recibir retroalimentación de los empleados para mejorar futuros procesos.

2.3 Análisis del mercado y la competencia

Durante el desarrollo del proyecto, se han evaluado aplicaciones existentes dedicadas a la gestión de horarios. Este análisis no solo ha permitido identificar las tendencias que más predominan en este sector, sino también detectar carencias y oportunidades que podrían ser aprovechadas para diseñar una solución competitiva y adaptada a las necesidades actuales de las empresas. La información pública sobre los aspectos técnicos de varias de estas herramientas es limitada: las soluciones de pago protegen sus algoritmos mediante propiedad intelectual, mientras que varias de las alternativas gratuitas no disponen del nivel de desarrollo técnico requerido en TimeFlex.

Los principales hallazgos del estudio de la competencia son:

UKG – UKG Ready Scheduler. UKG Ready Scheduler [71] es una herramienta prestigiosa en el mercado para la gestión de horarios, galardonada como la mejor empresa de software de RRHH según la plataforma G2 [2]. Esta herramienta dispone de un enfoque claro en la automatización y optimización. Su principal valor está en la capacidad de generar horarios ajustados a criterios predefinidos, como las habilidades de los empleados, su experiencia y la normativa vigente.

Una de las características más destacadas de esta herramienta es la interacción en tiempo real: los empleados pueden comunicar cambios en su disponibilidad a través de una interfaz sencilla, lo que mejora la flexibilidad y la gestión colaborativa. Sin embargo, esta funcionalidad tiene un alto coste asociado, ya que el precio oscila entre 27 y 37 dólares por empleado al mes, a lo que se suman las tasas de implementación iniciales. Esto convierte a UKG en una solución más orientada a grandes empresas con amplios presupuestos, limitando su accesibilidad para pymes.

Desde un punto de vista técnico, UKG establece un estándar elevado en términos de optimización. Permite optimizar horarios en función de múltiples aspectos como el presupuesto, las certificaciones de los empleados, el cumplimiento normativo o la demanda. Además, emplean herramientas de inteligencia artificial en su proceso de optimización [78]**.** Sin embargo, al no hacer públicos sus algoritmos ni sus métodos concretos de optimización la información disponible al respecto es limitada.

Opcenter Scheduling SMT. Opcenter [48] es una solución de pago desarrollada por Siemens dedicada a la gestión de procesos industriales. Aunque incluye funcionalidades de programación como la simulación de escenarios condicionales ("*what-if*") para evaluar el impacto en cambios de capacidad o demanda [54], su diseño está orientado principalmente a la planificación de materiales y la optimización de cadenas de producción, con poco interés en la interacción con los empleados o la flexibilidad en la gestión de horarios.

Su nivel de personalización lo hace adecuado para industrias con necesidades específicas y complejas, la interfaz de usuario es técnica y poco intuitiva, lo que podría dificultar su adopción en empresas fuera del ámbito industrial. Además, no incluye herramientas que permitan a los empleados participar directamente en la planificación o comunicar su disponibilidad.

OptaPlanner. OptaPlanner [49] es una herramienta de software de código abierto, programada en Java, diseñada para optimizar horarios y asignaciones de recursos. Su enfoque está claramente dirigido a departamentos de RRHH que buscan soluciones personalizables y económicas. Una de sus principales fortalezas es su capacidad para implementar algoritmos avanzados que resuelven problemas complejos de planificación, especialmente en organizaciones con múltiples restricciones (por ejemplo, disponibilidad de recursos, cumplimiento de normativas, etc.). Entre estos algoritmos destacan los metaheurísticos como la búsqueda tabú [81] [82] o la aceptación tardía, basadas en técnicas de búsqueda de óptimos locales, que exploran soluciones complejas sin estancarse con soluciones parciales. Esto se complementa con heurísticas constructivas, que generan soluciones iniciales eficientes en un tiempo finito, permitiendo asentar bases iniciales sólidas para las soluciones metaheurísticas, que resuelven finalmente el problema [12][30].

Sin embargo, OptaPlanner no está diseñada como una herramienta orientada al usuario final, puesto que carece de interfaces dentro de la propia herramienta que permitan a los trabajadores interactuar con ella o comunicar cambios en tiempo real. Para disponer de una interfaz gráfica interactiva, es necesario utilizar una solución adicional como OptaWeb Employee Rostering [50], desarrollada como complemento para OptaPlanner.

Esto provoca que la solución resulte adecuada para gestionar recursos desde un nivel técnico, pero insuficiente para organizaciones que buscan una herramienta integral que promueva la participación activa de los empleados en el proceso de planificación.

FET (Free Timetabling Software). FET [24] es una solución de código abierto para la generación de horarios, enfocada en la simplicidad y la accesibilidad. A pesar de su carácter gratuito, que lo hace atractivo para pequeños equipos o instituciones con presupuestos reducidos, su diseño anticuado y funcionalidades limitadas lo relegan a un segundo plano en comparación con otras opciones más modernas como UKG Ready Scheduler [71]. La herramienta no considera las necesidades de los empleados ni permite la optimización avanzada; simplemente genera horarios básicos en función de reglas predefinidas por el usuario desde su interfaz gráfica. Esta simplicidad puede ser útil en entornos con requisitos muy básicos, pero no responde a las demandas de empresas que buscan soluciones innovadoras y flexibles.

La herramienta no considera las necesidades de los empleados ni permite la optimización avanzada; simplemente genera horarios básicos en función de reglas predefinidas por el usuario desde su interfaz gráfica. Esta simplicidad puede ser útil en entornos con requisitos muy básicos, pero no responde a las demandas de empresas que buscan soluciones innovadoras y flexibles.

UniTime. UniTime [72] es una herramienta específica para la gestión de horarios en instituciones educativas. Su principal funcionalidad radica en la asignación de aulas y la gestión del uso de espacios físicos. Aunque es útil para este propósito, su diseño gráfico resulta desactualizado al carecer de aspectos relevantes como diseño *responsive* o iconos intuitivos dentro de la aplicación.

Además, su interfaz puede resultar difícil de manejar para el usuario final, ya que muchas acciones requieren múltiples pasos donde las opciones no están etiquetadas adecuadamente y las pantallas principales suelen estar sobrecargadas de información de los distintos formularios y configuraciones.

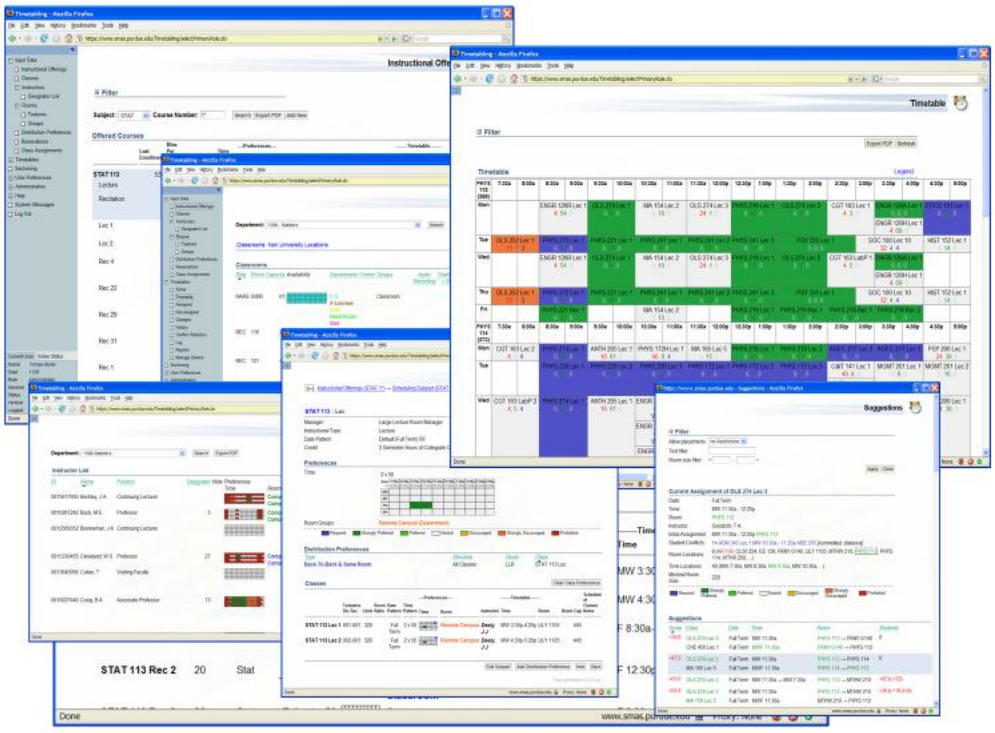

Figura 2-1. Aspecto visual de la interfaz de UniTime.

La Figura 2-1 muestra el aspecto visual de la interfaz de UniTime. Además, UniTime carece de herramientas de optimización que permitan generar horarios eficientes. En lugar de ello, simplemente asigna recursos disponibles en función de su disponibilidad. Por lo tanto, aunque es adecuada para contextos educativos, no aporta valor en un entorno empresarial o en la gestión de horarios de personal.

La Tabla 2-1 compara las distintas herramientas de optimización de turnos estudiadas.

Características					
Automatización y optimización	✓	✓	✓	✗	✗
Interacción en tiempo real	✓	✗	✗	✗	✗
Accesibilidad para pymes	✗	✓	✓	✓	✓
Interfaz intuitiva	✓	✗	✗	✗	✗
Participación activa de empleados	✓	✗	✗	✗	✗
Coste	Alto	Alto	Gratuito	Gratuito	Gratuito
Uso en industrias específicas	General	Industrial	General	Educativo	Educativo
Enfoque en normativas y restricciones	✓	✓	✓	✗	✗
Optimización avanzada	✓	✓	✓	✗	✗

Tabla 2-1. Comparación entre las distintas herramientas de optimización de turnos.

La conclusión del estudio realizado es que TimeFlex se posiciona como la única solución gratuita de código abierto con interfaz gráfica integrada que resuelve problemas de asignación de turnos y horarios con algoritmos de optimización de forma automática.

Capítulo 3 - Análisis previo y metodología

Este capítulo está dedicado tanto a la investigación realizada con los usuarios mediante encuestas y entrevistas como a la caracterización de estos en base a los resultados y conclusiones obtenidas. Esta caracterización comprende también sus escenarios de uso y sus recorridos dentro de la aplicación. La Sección 3.1 detalla la investigación llevada a cabo con usuarios a través del diseño y realización de encuestas. La Sección 3.2 describe el análisis realizado con los resultados de la investigación mediante métodos cualitativos y cuantitativos. La Sección 3.3 profundiza en las conclusiones obtenidas a partir de los resultados de las encuestas y entrevistas realizadas. La Sección 3.4 comprende la caracterización de los usuarios, así como la identificación de distintos roles dentro de la aplicación. La Sección 3.5 detalla los escenarios de uso de cada tipo de usuario de la aplicación. Por último, la Sección 3.6 describe las etapas clave en el recorrido de los usuarios, así como los puntos críticos en su experiencia con la aplicación TimeFlex.

3.1 Investigación con usuarios

Para comprender las necesidades y comportamientos de los usuarios objetivo de TimeFlex, se ha llevado a cabo una investigación basada en dos encuestas dirigidas a empleados y personal de RRHH de sectores como educación, sanidad y hostelería.

Las encuestas, diseñadas con preguntas abiertas y cerradas, facilitaron la recopilación de datos estructurados, la identificación de patrones generales y la medición objetiva de la aceptación de las propuestas. Este estudio, además, ha sido utilizado posteriormente en la fase de diseño de la herramienta (ver Capítulo 4).

Diseño de las encuestas. Se decidió realizar una encuesta que combinara tanto un enfoque cuantitativo como cualitativo en las preguntas, para posteriormente llevar a cabo un análisis mixto de las respuestas. Este análisis se basa en interpretar la información medible de las preguntas cerradas junto a la información contextual y matices aportados en las preguntas abiertas. De esta manera, se obtiene una visión global y completa de cada aspecto tratado en las encuestas. A continuación, se detallan los dos tipos de preguntas realizadas:

- **Preguntas cerradas:** Diseñadas para obtener datos cuantitativos, permitiendo identificar patrones estadísticos y tendencias en las respuestas de los participantes. Un ejemplo de pregunta cerrada es: "Has usado antes alguna aplicación para optimizar turnos?"
- **Preguntas abiertas:** Formuladas para obtener respuestas cualitativas más detalladas, otorgando a los participantes la oportunidad de expresar opiniones y ofrecer contexto adicional a las respuestas cuantitativas. Esto permite identificar las necesidades de los potenciales usuarios de TimeFlex. Un ejemplo de pregunta abierta es: "¿Cuál crees que es el equilibrio adecuado entre simplicidad y funcionalidades avanzadas para ti?"

  Las dos encuestas se estructuran en las siguientes secciones:

  - Experiencia y expectativas de una aplicación de optimización de turnos.
  - Horarios (preferencias, presentación, etc.).
  - Notificaciones (preferencias, frecuencia, etc.).
  - Solicitudes de cambio de turno (formato, pasos a seguir, etc.).
  - Interfaz gráfica.
  - Equilibrio entre simplicidad y funcionalidad avanzada.
  - Dispositivos preferidos.

En las secciones donde las opciones de respuesta eran limitadas o podían traducirse en categorías fácilmente medibles (como dispositivos preferidos), predominaron las preguntas cerradas, favoreciendo el análisis cuantitativo. En cambio, en secciones con una mayor variedad de posibilidades (como preferencias de asistencia y funcionalidades), se optó por incluir un mayor número de preguntas abiertas para realizar un análisis cualitativo detallado. Esta planificación cuidadosa optimizó el análisis de los datos obtenidos. Por ejemplo, en el diseño y la interacción con la aplicación, se han planteado preguntas cerradas como:

- "¿Prefieres una aplicación con menos opciones pero más fácil de usar, o una más completa y con más funcionalidades?".

- "¿Te gustaría tener una función de *chatbot* que te guíe a través del proceso o te ayude con preguntas frecuentes?".
- "¿Te gustaría que la aplicación te ofreciera análisis sobre cuándo podrías necesitar contratar más personal o ajustar los horarios?".
- "¿Te gustaría poder simular diferentes escenarios de horarios y visualizar el impacto en la satisfacción y productividad de los empleados?".
- "¿Crees que tienes en cuenta dichas preferencias de forma equitativa?".

Estas preguntas ayudan a validar las ideas iniciales sobre la aplicación. Al mismo tiempo, se han incluido preguntas abiertas como:

- "¿Qué tipo de asistencia o tutoriales te gustaría ver dentro de la aplicación para familiarizarte con sus funcionalidades?".
- "¿Cuál crees que es el equilibrio adecuado entre simplicidad y funcionalidades avanzadas para ti?".
- "¿Cuál sería tu experiencia ideal en términos de facilidad y rapidez?".
- "¿Cómo prefieres gestionar y priorizar las solicitudes de cambio de turno de los empleados?".
- "¿Qué inconvenientes son los más comunes a los que te enfrentas?".

Este enfoque permitió no solo evaluar la aceptación de las propuestas iniciales, sino también descubrir nuevas necesidades y sugerencias de los usuarios. Las encuestas diseñadas están incluidas en el Apéndice E.

Entrevistas personales. De forma adicional, se han llevado a cabo entrevistas personales con personas relacionadas con los ámbitos sanitario, docente y hostelero. Estas entrevistas aportaron una perspectiva práctica sobre los desafíos y necesidades específicas de los usuarios, apoyada en su experiencia profesional. La diversificación de sectores y roles (empleados y personal de RRHH) entre los encuestados permitió evitar sesgos y reflejar las necesidades de los distintos tipos de usuario de forma efectiva. De esta manera, se obtuvo una visión global y se complementaron los resultados obtenidos en las encuestas más estructuradas.

3.2 Análisis cualitativo y cuantitativo

Análisis cuantitativo. Para analizar las respuestas sobre datos cuantitativos obtenidas en la investigación, se ha realizado un análisis descriptivo que permitió identificar patrones y tendencias en las respuestas de los participantes de manera clara y objetiva. Este análisis se centró en las respuestas cerradas de la encuesta, proporcionando una visión general del comportamiento y las preferencias de los usuarios objetivo.

En primer lugar, se han utilizado frecuencias absolutas y relativas para medir la popularidad de diversas opciones de respuesta, como las preferencias en diseño de la aplicación, dispositivos utilizados y funcionalidades deseadas. Esto permitió calcular porcentajes que reflejan el grado de aceptación o satisfacción de ciertas características, como la preferencia por una interfaz simplificada.

Además, se han empleado varias medidas de dispersión (desviación estándar) y de tendencia central (media y mediana) en preguntas valorables en escalas numéricas, como la frecuencia con la que se desea recibir alertas de la aplicación. Estas métricas ayudaron a identificar las tendencias principales y a interpretar la variabilidad presente en las respuestas.

Finalmente, el análisis descriptivo se completó con la generación de gráficos y tablas para representar visualmente los resultados, facilitando la interpretación y permitiendo obtener conclusiones relevantes. Los gráficos más utilizados han sido los de sectores/circulares y los diagramas de barras. Estos recursos visuales han permitido observar rápidamente las preferencias y patrones clave, asegurando que el diseño de la aplicación se adapte a las necesidades principales de los futuros usuarios.

Análisis cualitativo. Para analizar las respuestas cualitativas de las encuestas, se ha utilizado un análisis de contenido. Este método permite codificar y categorizar las respuestas abiertas, organizándolas en temas recurrentes que reflejan las expectativas, inquietudes y preferencias tanto de los empleados como del departamento de RRHH. El proceso de análisis incluyó la creación de categorías clave como "Preferencias de interacción" o "Gestión de turnos y vacaciones".

Posteriormente, se identificaron patrones que permitieron descubrir cuestiones clave como la necesidad de una aplicación flexible y personalizada, la importancia de una comunicación efectiva y en tiempo real, y el valor de la automatización en la gestión de turnos y ausencias. A través de este análisis, se identificaron también frustraciones comunes de los usuarios como la preocupación por la privacidad o la resistencia al cambio. Esto permitió concluir que se debe tener en cuenta dichos aspectos de cara al desarrollo final de la aplicación.

La interpretación de todos estos resultados sirvió para complementar el análisis cuantitativo, proporcionando una comprensión integral y profundizada de las necesidades y experiencias de los futuros usuarios de TimeFlex.

Integración de resultados. La integración de datos cualitativos y cuantitativos se consigue mediante la triangulación de datos, es decir, se comparan y combinan los datos cuantitativos de las respuestas cerradas con los cualitativos de las respuestas abiertas mediante el siguiente método de dos fases:

(i) Análisis de los datos cuantitativos en búsqueda de los patrones más repetidos y significativos.

(ii) Una vez identificados dichos patrones, se consultan las preguntas abiertas relacionadas con ellos y se analizan las respuestas de los usuarios con el fin de obtener una explicación o contexto generalizado para el patrón.

Este método nos permite corroborar los patrones detectados en las encuestas con las explicaciones detalladas de los participantes, proporcionando un contexto y una justificación de estas tendencias. Como resultado de este proceso, se obtienen conclusiones relevantes para la investigación cuyo contenido se apoya tanto en datos como en opiniones de profesionales del sector.

3.3 Resultados y conclusiones de la investigación

En cuanto a los resultados de la investigación, las encuestas obtuvieron 44 respuestas: 27 para la de empleados y 17 para la de gerentes de RRHH. Con respecto a las entrevistas, se realizaron 8 en total, con los siguientes perfiles: (i) 3 empleados de hospitales, (ii) 2 gerentes de RRHH de hospitales y centros de salud, (iii) 2 empleados de

centros educativos y (iv) 1 gerente de RRHH del sector de restauración. Tras aplicar el método explicado anteriormente y analizar los resultados obtenidos, se han identificado hallazgos de interés. A continuación, se detallan las conclusiones obtenidas para cada arquetipo, y finalmente aquellas comunes a ambos grupos.

3.3.1 Empleados

Las principales conclusiones obtenidas del análisis descrito anteriormente para mejorar la experiencia de los trabajadores rasos radican en la simplicidad, la personalización y la eficiencia. Los empleados buscan una herramienta que les permita gestionar sus horarios, turnos y vacaciones de forma sencilla, a la vez que desean tener control total sobre aspectos como las notificaciones y la apariencia de la aplicación. Este enfoque hacia la personalización es percibido como esencial por el 88,8% de los encuestados para aumentar la satisfacción y garantizar una mayor productividad.

Además, se han observado ciertos comportamientos recurrentes, como la necesidad frecuente de consultar y comparar horarios (62,9% de los participantes), y la predisposición a ofrecer retroalimentación (85,1% de los encuestados), que han revelado la importancia que otorgan a la colaboración y la comunicación fluida en el entorno laboral. Los empleados valoran la capacidad de visualizar sus turnos y los de sus compañeros de manera intuitiva, y esperan que las herramientas de retroalimentación sean eficaces y fáciles de usar. Este *feedback* continuo es considerado fundamental para el éxito de la aplicación y persigue potenciar el bienestar laboral, lo cual, a largo plazo, se espera que se traduzca en un aumento del rendimiento y la motivación de los empleados.

Sin embargo, este análisis también ha identificado algunas dificultades potenciales que deben considerarse en el diseño y desarrollo de TimeFlex. La adaptación a la interfaz y la gestión de notificaciones podrían generar desafíos, especialmente para aquellos más acostumbrados a métodos tradicionales, pues un 74,07% de los encuestados han expresado al menos una de estas inquietudes. También es importante tener en cuenta problemas técnicos como la compatibilidad con ciertos dispositivos y la sobrecarga de información, con el objetivo de evitar frustraciones entre el personal. En definitiva, el éxito de TimeFlex entre los empleados rasos dependerá de

su capacidad para equilibrar la simplicidad, personalización y eficiencia con la resolución efectiva de las dificultades que puedan surgir en la experiencia de los usuarios.

3.3.2 Gerentes de RRHH

Este grupo considera que la implementación de una aplicación eficaz puede marcar la diferencia en términos de productividad y satisfacción laboral. Sus principales necesidades además de la eficiencia comprenden la organización y la optimización del tiempo. Una amplia mayoría de este tipo de trabajadores (94,11%) considera que la automatización de la asignación de turnos, junto con la capacidad de supervisar y ajustar el proceso según sea necesario, facilitaría enormemente la labor del equipo, reduciendo el margen de error y mejorando la distribución de los turnos.

Además, un 82,35% de los encuestados expresó su deseo por tener la capacidad de personalizar aspectos clave de la aplicación, como notificaciones y paneles de visualización, lo que permitiría a los responsables de RRHH adaptar la herramienta a sus necesidades y mejorar la gestión del tiempo. Sin embargo, existen también obstáculos como la dificultad para adaptarse a nuevas herramientas, la potencial sobrecarga de información y la gestión de conflictos, que deben ser tomados en consideración.

Su enfoque proactivo en el uso de retroalimentación (76,47%) se considera esencial para lograr mejoras tanto en la plataforma como en los procesos internos. Este ciclo constante de retroalimentación y optimización se espera que resulte en una experiencia de usuario más fluida y eficiente, tanto para los empleados como para el equipo de RRHH.

3.3.3 Conclusiones comunes

Las conclusiones mencionadas previamente de forma particular para cada arquetipo se agrupan en las siguientes conclusiones globales:

- Fuerte preferencia por una interfaz simple e intuitiva.
- Necesidad de disponer de opciones de personalización, especialmente para el sistema de notificaciones (frecuencia, tipos, …).

- Facilitar la retroalimentación en la aplicación.
- Garantizar la eficiencia de las funcionalidades de la aplicación.
- Considerar potenciales dificultades para los usuarios, poniendo el foco en evitar sobrecargar al usuario de información.

Tras obtener estas conclusiones, se realizó el modelado de cada tipo de usuario de la aplicación. Esto incluyó sus respectivos *Customer Journey Maps* (ver Sección 3.6), la especificación de los requisitos para cada perfil de usuario y el diseño de un prototipo de baja fidelidad para el proyecto. Al cierre de cada fase, se llevó a cabo un análisis de la información obtenida con el objetivo de recibir retroalimentación y corregir las inconsistencias detectadas durante el proceso.

3.4 Caracterización de los usuarios

Como se ha mencionado previamente, se han creado perfiles detallados para representar y modelar a los diferentes tipos de usuarios de TimeFlex. De esta manera se han identificado los siguientes roles, y necesidades:

1. **Empleados:**
    - **Perfil:** Trabajadores que operan bajo sistemas de turnos en sectores como sanidad, docencia u hostelería.
    - **Objetivos:** Gestionar turnos y vacaciones de forma ágil, personalizar su experiencia en la aplicación, y consultar horarios propios y de compañeros para una mejor coordinación.
    - **Frustraciones:**
        - Problemas técnicos (compatibilidad móvil y rendimiento).
        - Falta de transparencia en la asignación de turnos.
        - Dificultades iniciales para adaptarse a la interfaz.
        - Sensación de no ser escuchados en el proceso de retroalimentación.

- **Necesidades clave**: Simplicidad, personalización y transparencia en la gestión de sus horarios.

2. **Gerentes de RRHH:**
    - **Perfil:** Encargados de gestionar los turnos y vacaciones de los trabajadores mientras equilibran las preferencias individuales y las necesidades operativas.
    - **Objetivos:** Automatizar la asignación de turnos, reducir la carga administrativa, garantizar transparencia y cumplir con las normativas laborales.
    - **Frustraciones:**
        - Complejidad en el diseño y envío de formularios.
        - Baja tasa de respuesta de los empleados a los formularios enviados.
        - Problemas de flexibilidad de la herramienta para responder a cambios imprevistos.

3.5 Escenarios de uso

Junto a los perfiles de usuarios, se han desarrollado escenarios para ilustrar cómo cada arquetipo identificado interactúa con la aplicación, reflejando sus procesos habituales y cómo se abordan sus problemas y necesidades:

1. **Escenarios para usuarios:**
    - **Consulta de horarios:** El usuario inicia sesión en la aplicación, accede a su calendario y verifica sus turnos asignados. Si detecta un conflicto, puede solicitar un cambio de turno directamente desde la plataforma.
    - **Personalización y retroalimentación:** El usuario ajusta sus notificaciones para recibir sólo la información relevante y utiliza el formulario de retroalimentación para expresar sus opiniones sobre los turnos asignados.

2. **Escenarios para gerentes de RRHH:**
    - **Automatización de turnos:** El gerente diseña un formulario para recopilar preferencias de turnos de los empleados. Una vez que los formularios están completados, utiliza el algoritmo de optimización para generar los horarios finales.
    - **Notificaciones y análisis:** Envía notificaciones personalizadas a los empleados con sus horarios asignados. Posteriormente, revisa reportes gráficos para identificar áreas de mejora y evaluar la satisfacción de los usuarios.

Estos escenarios han sido diseñados y validados mediante prototipos de baja fidelidad, asegurando que las funcionalidades principales de la aplicación se alineen con las expectativas de los usuarios y fomenten una experiencia satisfactoria y eficiente.

3.6 Customer Journey Maps

Haciendo uso de las conclusiones obtenidas en la Sección 3.3, se han desarrollado los *Customer Journey Maps*, representaciones del recorrido que siguen los clientes (usuarios) para lograr un objetivo específico al interactuar con nuestra aplicación. Estos permiten identificar las etapas clave del recorrido del usuario (empleados y personal de RRHH) al interactuar con la aplicación TimeFlex. Estas etapas reflejan los puntos críticos en la experiencia de usuario y permiten abordar necesidades y puntos de mejora específicos:

1. **Fase de preparación:**
    - **Empleados:** Rellenar formularios con sus preferencias de horarios y vacaciones, permitiendo a la aplicación generar sus turnos automáticamente.
    - **Gerentes de RRHH:** Diseñar formularios personalizados y configurar notificaciones para asegurar que los empleados completen los datos necesarios.
    - **Emociones:** Frustración por la complejidad de los formularios y preocupación/ansiedad por posibles errores en la información recopilada.

2. **Fase de recopilación de respuestas:**
    - **Empleados:** Enviar formularios completados dentro del plazo estipulado.
    - **Gerentes de RRHH:** Realizar un seguimiento para asegurar que los formularios estén completos y listos para la generación de turnos.
    - **Emociones:** Ansiedad por el retraso en las respuestas y frustración por errores en los formularios.
3. **Fase de generación y optimización de turnos:**
    - **Empleados:** Esperar la asignación de sus turnos y vacaciones.
    - **Gerentes de RRHH:** Ejecutar el algoritmo de optimización para generar horarios, equilibrando las preferencias de los empleados con las necesidades operativas.
    - **Emociones:** Inquietud por la transparencia y precisión del algoritmo, impaciencia por el tiempo de procesamiento.
4. **Fase de notificación:**
    - **Empleados:** Recibir notificaciones personalizadas con sus horarios asignados.
    - **Gerentes de RRHH:** Enviar las notificaciones y supervisar la recepción por parte de los empleados.
    - **Emociones:** Alivio y satisfacción por completar el proceso, pero también incertidumbre inicial sobre si los horarios cumplen las expectativas.
5. **Fase de retroalimentación y ajustes:**
    - **Empleados:** Enviar comentarios sobre la asignación de turnos y solicitar cambios o ajustes.
    - **Gerentes de RRHH:** Analizar las solicitudes de retroalimentación para mejorar futuras asignaciones.
    - **Emociones:** Frustración si no se resuelven las solicitudes, pero satisfacción cuando se implementan ajustes efectivos.

Puntos de contacto y emociones. Cada etapa del recorrido está asociada a puntos de interacción específicos y emociones vinculadas:

1. **Formularios de preferencias:**
    - Punto de contacto: Página de formularios en la aplicación.
    - Emociones: Inicialmente puede haber agobio por el número de preguntas; sin embargo, la claridad y simplificación de los formularios reducen la frustración.
2. **Interacción con el algoritmo de optimización:**
    - Punto de contacto: Pantalla de "Horarios" y generación de turnos.
    - Emociones: Impaciencia durante el procesamiento, alivio al recibir horarios acordes a las preferencias.
3. **Notificaciones personalizadas:**
    - Punto de contacto: Pantallas emergentes o notificaciones push en dispositivos móviles.
    - Emociones: Alivio al recibir confirmación inmediata de horarios asignados.
4. **Retroalimentación:**
    - Punto de contacto: Formulario de retroalimentación o encuestas rápidas.
    - Emociones: Satisfacción al expresar opiniones, pero posible frustración si las respuestas no son tomadas en cuenta.

Capítulo 4 - Diseño de la aplicación

En este capítulo se detalla el diseño y arquitectura de la aplicación. La Sección 4.1 profundiza en las diferentes capas de la aplicación (presentación, lógica de negocio, servicios) y la Sección 4.2 detalla la estructura de la base de datos utilizada.

4.1 Arquitectura del sistema

La aplicación ha sido diseñada utilizando una arquitectura en capas, es decir, una estructura que permite desarrollar software separando distintas funcionalidades en sistemas diferentes, permitiendo una mayor escalabilidad y mantenibilidad de código. En este caso, la arquitectura está compuesta por tres capas principales.

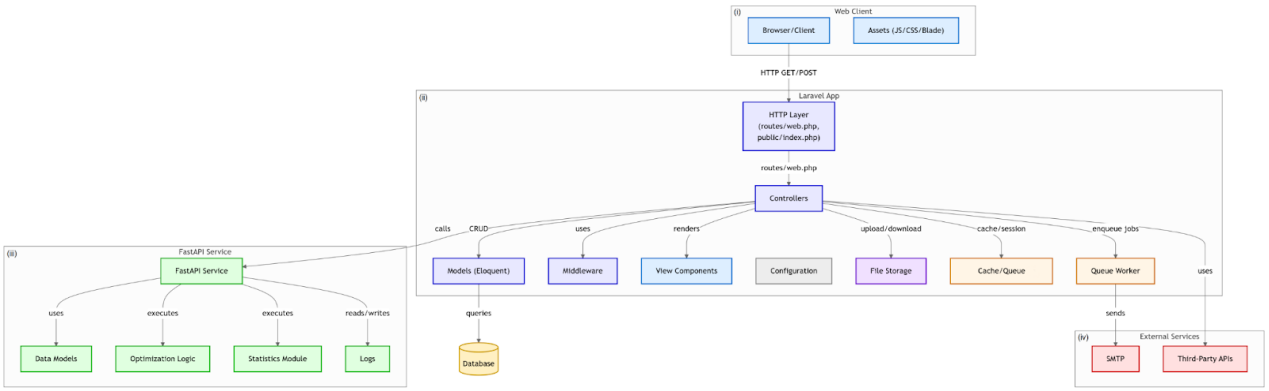

Figura 4-1. Diagrama de la arquitectura.

La Figura 4-1 muestra la arquitectura por componentes de TimeFlex, organizada de manera modular para representar claramente las responsabilidades de cada parte del sistema. Como puede observarse, se divide en 4 grandes grupos: (i) interfaz web, (ii) capa de presentación y lógica de negocio, (iii) microservicio y, (iv) servicios externos.

Los colores de cada componente en los que se muestra cada componente representan de manera visual la funcionalidad de cada uno: el color azul claro representa los elementos del cliente y la interfaz de usuario, el color morado es utilizado en las componentes de la aplicación Laravel, el verde agrupa los componentes básicos del microservicio FastAPI [21], los servicios relacionados con colas o caché están representados en naranja, en rojo aparecen los servicios externos, en gris se muestran los archivos de configuración y por último, en amarillo la base de datos central.

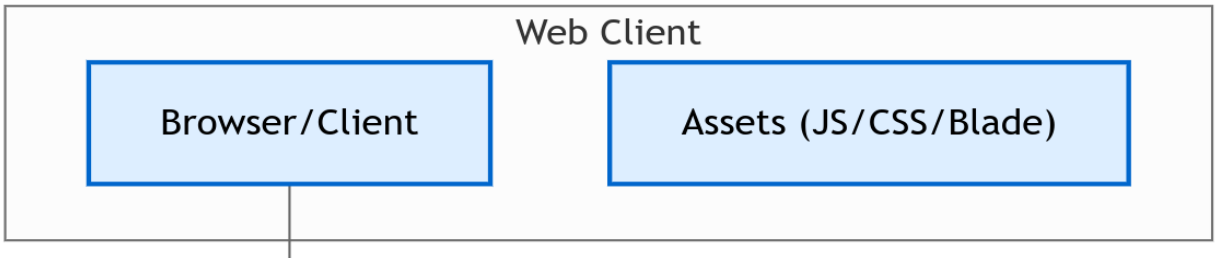

Figura 4-2. Diagrama de la interfaz web.

(i) Interfaz web. La Figura 4-2 muestra los componentes de la interfaz web. Incluye el navegador del usuario, así como recursos estáticos que componen la interfaz (JavaScript [95] , CSS [14] , etc.). Esta capa permite a los usuarios interactuar con el sistema a través de vistas adaptadas a sus necesidades.

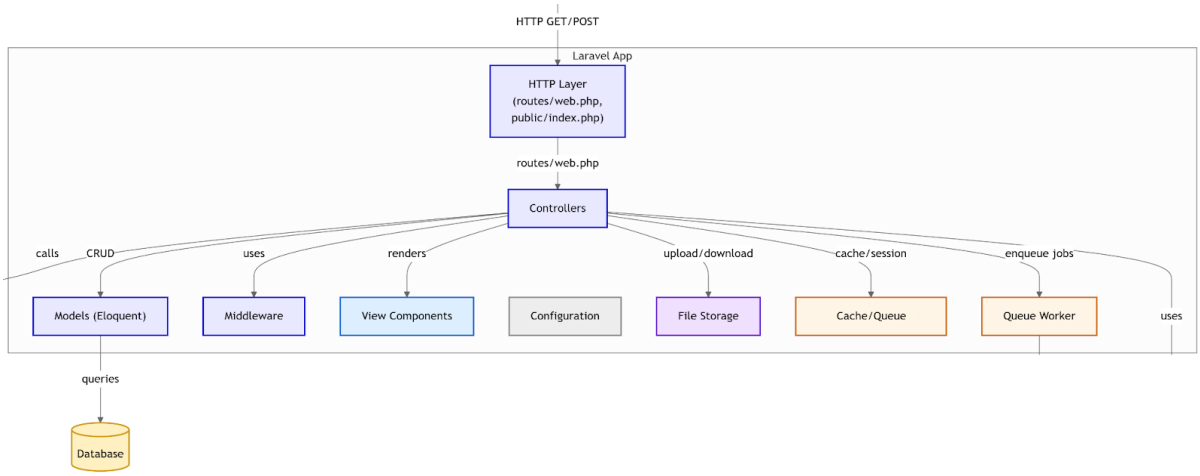

Figura 4-3. Diagrama de la capa de presentación y lógica de negocio.

(ii) Capa de presentación y lógica de negocio en Laravel. La Figura 4-3 contiene los componentes de la capa de presentación y lógica de negocio. Representa el núcleo de la aplicación, desarrollada en Laravel . Se encarga de procesar las solicitudes HTTP mediante un sistema de rutas bien estructurado. Contiene los siguientes componentes clave:

- ***Controllers***: Gestionan las acciones solicitadas por el usuario, se dividen los controladores dependiendo del ámbito de la acción.

- **Modelos *Eloquent***: Clases que representan las entidades del sistema, encapsulando la lógica y manipulación de datos en la base de datos (representada en color amarillo).
- ***Middlewares***: Encargados de gestionar aspectos como la autenticación, el control de acceso, el registro de actividad y otras validaciones producidas antes o después de las ejecuciones.
- ***View Components***: Fragmentos modulares y reutilizables de interfaz basados en Blade [3], que renderizan la interfaz de usuario, separando lógica de presentación.
- **Gestión de configuración, almacenamiento de archivos, caché**: Incluye el manejo de sesiones, almacenamiento de archivos, y cacheo de consultas y vistas para mejorar el rendimiento de la aplicación.

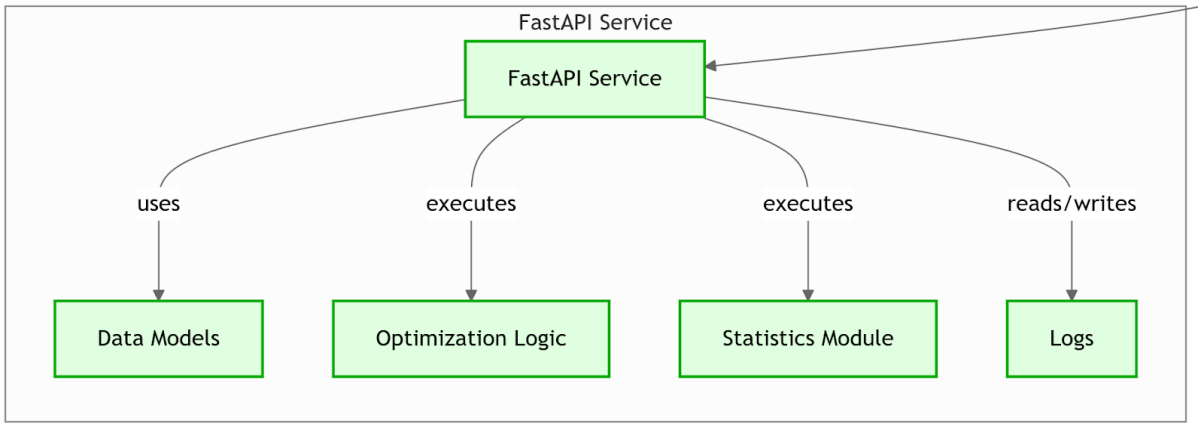

Figura 4-4. Diagrama del microservicio en FastAPI.

(iii) Microservicio en FastAPI. La Figura 4-4 contiene los principales componentes del microservicio incluido en TimeFlex. Esta capa contiene un microservicio desacoplado, desarrollado en Python con la librería FastAPI . Este servicio expone una API que es utilizada por Laravel y contiene los siguientes submódulos:

- **Modelo de datos**: Se encarga de validar, estructurar y transformar los datos de entrada. La Sección 4.1.2 describe en detalle los sistemas de validación del microservicio.

- **Lógica de optimización basado en Z3:** Implementa algoritmos de optimización utilizando el resolutor de restricciones Z3 [76] [87]. La Sección 6.2 presenta el resolutor y las restricciones utilizadas en detalle.
- **Lógica y cálculo de satisfacción de usuarios:** Módulo que evalúa el nivel de satisfacción de los usuarios, cuyos resultados son utilizados por la lógica de optimización.
- **Sistema de logs**: Registra accesos, errores, métricas de consumo y otros eventos relevantes para el monitoreo y la trazabilidad del microservicio.

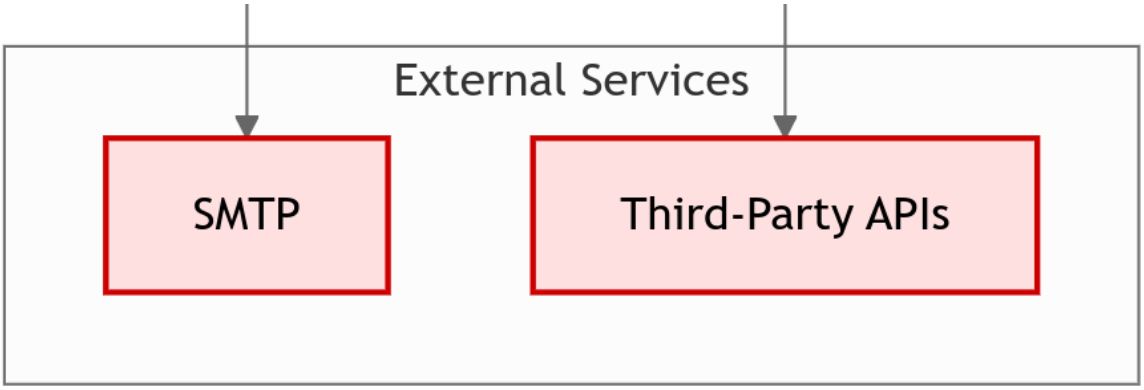

Figura 4-5. Diagrama de los servicios externos.

(iv) Servicios externos. Los componentes relacionados con el servicio externo se muestran en la Figura 4-5. En este componente se encuentran integraciones con servicios de terceros, como SMTP (*Simple Mail Transfer Protocol*) [61], que facilitará en el futuro el envío automatizado de correos electrónicos desde la plataforma. También se incluye la conexión a APIs externas, utilizadas actualmente para cargar imágenes, iconos u otros recursos. Estas integraciones facilitan la escalabilidad, permitiendo la incorporación y sincronización de otros sistemas, como *Outlook* [51] o *Google Calendar* (ver Capítulo 9 para más información detallada).

4.1.1 Capa de presentación y lógica de negocio

Desde que fue creado en 1994, PHP ha vivido una gran evolución mediante la publicación recurrente de nuevas versiones. Su popularidad como lenguaje para el desarrollo web es incuestionable, sin embargo, también ha sido objeto de numerosas críticas debido a múltiples carencias asociadas a versiones anteriores del lenguaje.

En las versiones 4 y 5 de PHP, la falta de estructura y arquitectura se hizo evidente ya que se podía mezclar código PHP con HTML sin orden ni capas, y no había ningún tipo de organización de controladores ni vistas. Además, el código era difícilmente reutilizable y la seguridad era insuficiente puesto que las aplicaciones web eran vulnerables frente a ataques CSRF [85] [89] , XSS [91] [92] o inyecciones SQL [80] [93](se habla en profundidad de la seguridad de la aplicación en la Sección 6.3).

Estas carencias motivaron la publicación de nuevas versiones y la creación de *frameworks* que ofrecieran una solución a estos problemas. Estos *frameworks* trajeron mejoras significativas como facilitar la implementación del patrón modelo-vista-controlador (MVC), modularidad, o mejores herramientas de seguridad y escalabilidad. Entre ellos, destaca Laravel , utilizado en TimeFlex tanto para la interfaz de usuario como para la lógica de negocio. Sin embargo, existen otros *frameworks* de PHP que podrían haber sido utilizados:

- **CodeIgniter** destaca por su simplicidad y por ser más ligero en cuanto a recursos que sus competidores, pero sin sacrificar rendimiento ni seguridad.
- **Symfony** sobresale por su modularidad, ofreciendo una gran flexibilidad para adaptar la arquitectura del proyecto a los requisitos de negocio.
- **CakePHP** es más apropiada para pequeños proyectos, ya que, si bien permite a los usuarios desarrollar código rápidamente usando diseños simples, presenta mayores limitaciones en términos de configuración y modularidad. Por lo tanto, no se recomienda para proyectos que implementen una lógica compleja o que requieren un alto nivel de personalización.

Pese a que Laravel no es el único *framework* para PHP que existe, es el que despierta más interés, como se puede observar en la Figura 4-6. Esta compara el interés generado por cada una de las herramientas mencionadas anteriormente, medido en número de búsquedas en internet, desde el 1 de enero de 2015 hasta abril de 2025.

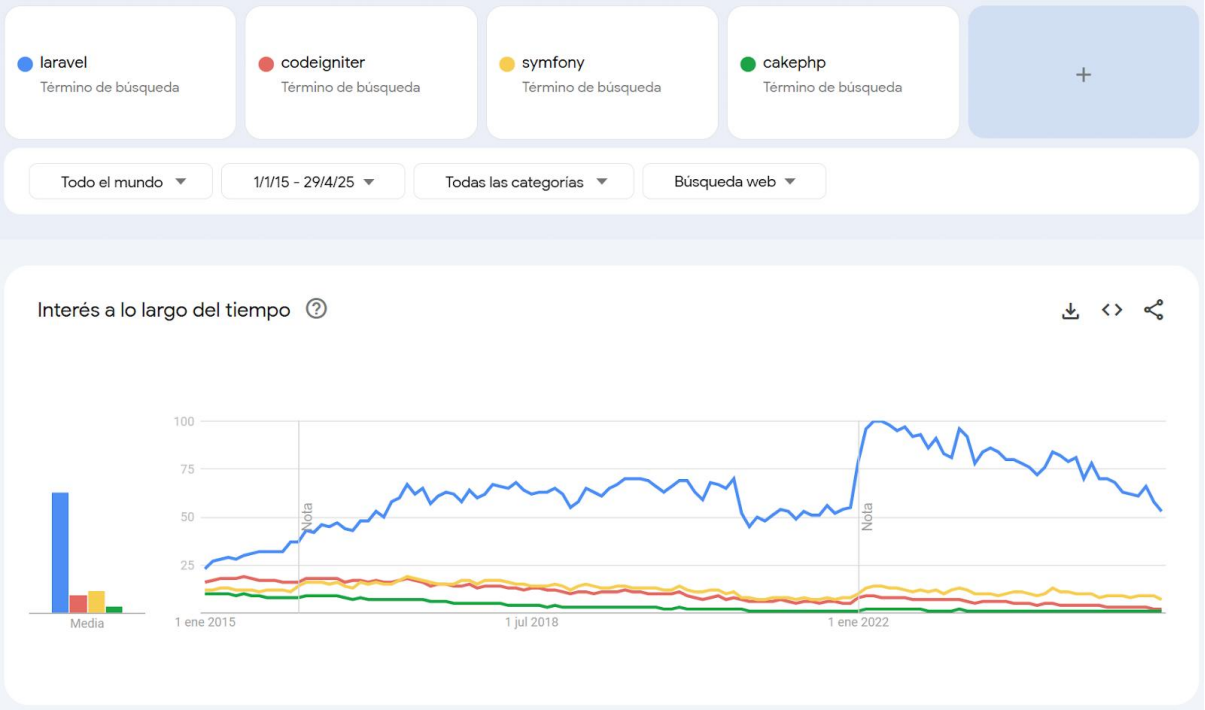

Figura 4-6. Gráfico comparativo respecto al interés despertado por frameworks de PHP.

Teniendo en cuenta el diseño inicial planteado, que prioriza tanto un dominio sencillo y rápido del *framework* como el valor aportado a la aplicación, se han identificado una serie de atributos sobre los cuales valorar las distintas herramientas. La Tabla 4-1 compara el rendimiento de los *frameworks* descritos anteriormente (columnas) en relación con los atributos mencionados para el proyecto de TimeFlex (filas).

	Laravel	**CodeIgniter**	**Symfony**	**CakePHP**
Arquitectura	MVC	MVC	MVC	MVC
Curva de aprendizaje	Media	Baja	Alta	Baja
Flexibilidad	Muy alta	Media	Muy alta	Media
Paquetes y extensiones disponibles	Muy completo	Limitado	Muy completo	Medio

Documentación	Muy extensa	Extensa	Extensa	Media
Simplicidad	Media	Alta	Baja	Alta
Seguridad	Muy alta	Baja	Muy alta	Alta
Soporte para APIs	Muy completo	Limitado	Muy completo	Medio
Peso en recursos	Moderado	Muy bajo	Alto	Bajo

Tabla 4-1. Comparativa entre distintos frameworks de PHP.

Sus múltiples ventajas incluyen su gran popularidad con respecto a otros *frameworks* de PHP, lo que se traduce en un mayor número de usuarios utilizando el *framework* y una más extensa y completa documentación, lo que particularmente se ha priorizado a la hora de desarrollar TimeFlex. Además, es un *framework* relativamente simple con muchas plantillas ya construidas, lo que agiliza el desarrollo de código, y dispone de un alto nivel de seguridad.

Entrando en profundidad en este último aspecto, Laravel utiliza por defecto un algoritmo de *hashing* para contraseñas llamado Argon2 [77] (aunque también es compatible con Bcrypt [79] , una función de *hash* de contraseñas y derivación de claves basada en el cifrado Blowfish [4]), positivamente valorado por poder ajustar el tiempo necesario para generar un *hash*. Este aspecto es relevante a la hora de generar *hashes* de contraseñas, ya que cuanto más tiempo tarda un algoritmo en generar un *hash*, mayor es el tiempo necesario para generar *rainbow tables* [88] con todos los posibles valores de los *hashes* por parte de los atacantes. De esta forma, se desincentiva a potenciales atacantes de lanzar ataques de fuerza bruta contra la aplicación web.

Laravel también permite sistemas de caché, aspecto diferenciador respecto a otros *frameworks*, lo que mejora de forma significativa el rendimiento de la aplicación web. En línea con esto, el sistema de colas único de Laravel permite posponer ciertas tareas de la web como enviar correos, lo que habilita la gestión de tareas necesarias de

forma más eficiente otorgando una mayor capacidad de procesamiento. Adicionalmente, Laravel facilita la integración de aplicaciones de terceros mediante APIs, capacitando a la aplicación web para un número extenso de funcionalidades.

A continuación, se muestran comparativas detalladas entre Laravel y los distintos *frameworks* con el fin de justificar la elección realizada:

Comparación entre Laravel y CodeIgniter. Estos dos frameworks utilizan el patrón de base de datos "*active record*", que permite realizar con poco código operaciones de inserción, recuperación y modificación de información en la base de datos. La primera gran diferencia aparece en los sistemas de autenticación, mientras que CodeIgniter [11] utiliza *Shield* [59] , que requiere configuración manual, Laravel ofrece una solución más completa y automatizada con múltiples opciones como *Breeze* [32] o *Jetstream* [36]. En términos de seguimiento del patrón Modelo-Vista-Controlador, CodeIgniter es más laxo ofreciendo mayor flexibilidad mientras que Laravel [38] es más estricto, garantizando una mejor división de responsabilidades. Por otro lado, la gestión de errores es más completa, configurable y avanzada en Laravel, ya que permite personalizar páginas de error según el tipo de fallo e integrar de manera sencilla servicios de monitorización externos [41]. Además, Laravel ofrece un mejor rendimiento basándonos en las solicitudes por segundos que es capaz de gestionar, destacando frente a CodeIgniter y el resto de las alternativas como se puede observar en la Figura 4-7 [10].

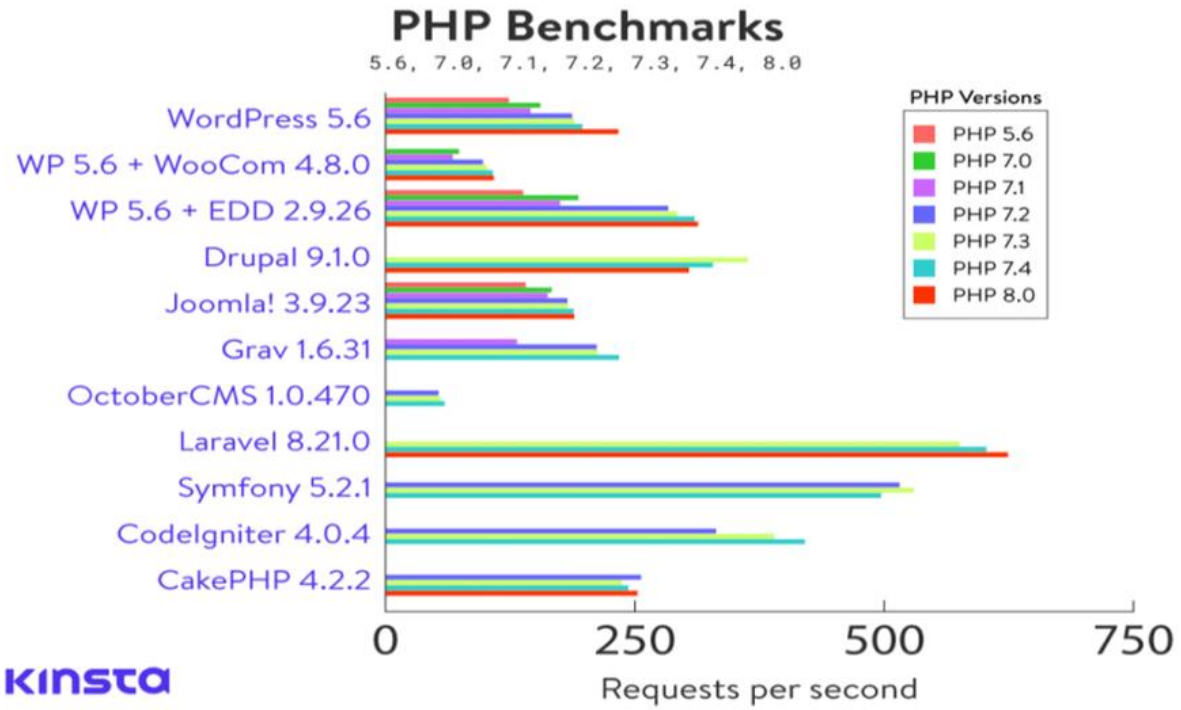

Figura 4-7. Comparación entre el rendimiento de distintos frameworks de PHP (fuente Kinsta).

Comparación entre Laravel y Symfony. Ambos *frameworks* coinciden en varios puntos como por ejemplo en el uso de un diseño basado en el patrón Modelo-Vista-Controlador. Sin embargo, existen también diferencias entre estos *frameworks*. En primer lugar, la curva de aprendizaje de Laravel [38] es más suave frente a la de Symfony [66]. Esto se debe a que Symfony dispone de una documentación más reducida y un menor número de tutoriales. Esto enlaza con la popularidad de Laravel, que dispone de una comunidad mucho más activa que Symfony. En relación con la modularidad, Symfony cuenta con ventaja gracias a sus componentes reusables que aprovechan su mayor simplicidad en organización de código, lo que es especialmente beneficioso en proyectos de gran magnitud. Finalmente, las migraciones en Symfony se automatizan con definiciones sencillas de campos, en contraposición a Laravel, donde son manuales, ofreciendo una mayor flexibilidad y control [65].

Comparación entre Laravel y CakePHP. Mientras que CakePHP [6] destaca como una opción más adecuada para proyectos de tamaño reducido debido a su mayor simplicidad, Laravel [38] se postula como una opción idónea para proyectos que

requieren mayor complejidad dada su capacidad de personalización e integración de otras aplicaciones y servicios. Esto se evidencia en sus procesos de *binding*, ya que Laravel permite la inyección automática de dependencias. Esto proporciona una mayor escalabilidad.

Sin embargo, CakePHP únicamente es capaz de utilizar un enfoque manual, que evidentemente es más limitado. La documentación es otro punto a favor de Laravel, ya que, como se ha mencionado anteriormente, cuenta con una mayor comunidad de usuarios y documentación. Finalmente, Laravel es una opción mucho más sólida para proyectos que requieren utilizar bases de datos y operaciones o elementos relacionados como copias de seguridad o inserción de datos, ya que cuenta con un gran ecosistema de paquetes diseñados para dichas acciones, por ejemplo, *Laravel Backup* [31]. CakePHP también permite gestionar datos, pero carece de tantas herramientas, y por lo tanto, sus prestaciones son menores [5].

4.1.1.1 Capa de presentación

La capa de presentación es la encargada de mostrar información al usuario y permite la interacción de este con la aplicación web. Esta interacción es fundamental y se debe garantizar que sea eficiente además de clara, rápida e intuitiva. Para lograr estos objetivos, se han utilizado distintos componentes: Blade [3], Alpine.js [1] y Tailwind CSS [67].

Blade [3] es un motor de plantillas incluido dentro del *framework* Laravel. Es una herramienta que destaca frente a otras similares gracias a la libertad que ofrece en términos de uso, ya que permite utilizar código PHP dentro de las vistas. Esto permite embeber código propio dentro del código HTML. Además, Blade compila las plantillas en código PHP y guarda en la caché dichas plantillas ya compiladas para próximas interacciones evitando ralentizar la aplicación web y garantizando rapidez en sus respuestas. Esto ha permitido generar vistas dinámicas y eficientes, combinando código PHP y HTML.

Para complementar las estructuras HTML generadas por Laravel y Blade, se ha utilizado Alpine.js para mejorar la interacción de los usuarios con la aplicación TimeFlex.

Alpine.js [1] es un *framework* de JavaScript que permite añadir interacción con los usuarios dentro de cualquier aplicación web y sobresale frente a *frameworks* más completos para esta funcionalidad como Vue [74] o React [58] por ser considerablemente más simple y ligero. Alpine.js está compuesto únicamente por 15 atributos, 6 propiedades y 2 métodos, a través de los cuales ofrece toda su funcionalidad. Esto la hace más ligera en cuanto a uso de recursos frente a sus competidores. Además, su curva de aprendizaje es mucho menor a pesar de ofrecer un número considerable de oportunidades interactivas. En definitiva, Alpine.js ha sido indispensable para manejar acciones habituales en aplicaciones web como mostrar u ocultar secciones, abrir o cerrar menús y pop ups o gestionar el uso de *toggles*, casillas de verificación y otros elementos interactivos.

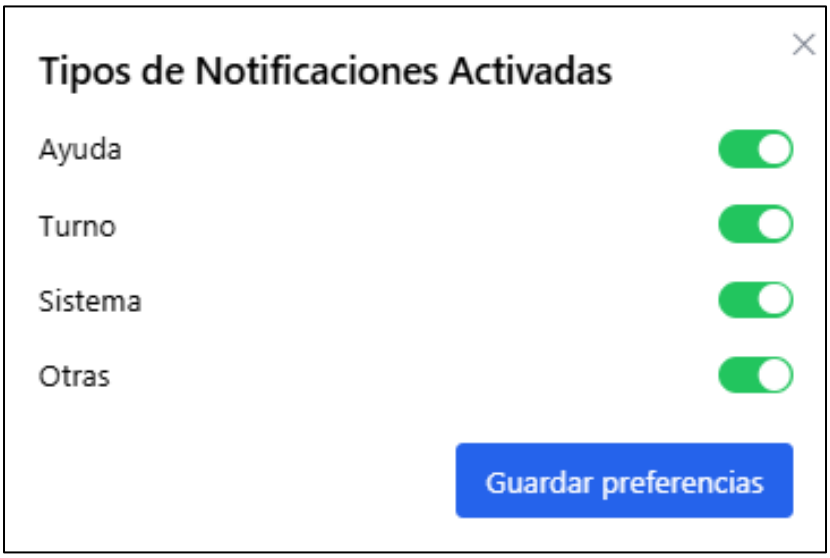

Figura 4-8. Toggles interactivos para las preferencias de notificaciones.

Por ejemplo, Alpine.js ha permitido implementar el panel de configuración de notificaciones mostrado en la Figura 4-8, dentro de TimeFlex.

Otro elemento esencial de cualquier aplicación web es disponer de una buena estética y coherencia visual. Para ello, se ha utilizado Tailwind CSS [67], un potente framework de CSS [14] que facilita la creación de estilos y diseños web. Tradicionalmente, el diseño de aplicaciones web con CSS se basaba en desarrollar clases personalizadas en hojas de estilo separadas para personalizar el aspecto visual.

No obstante, esta práctica conlleva la creación de extensos archivos CSS con poca trazabilidad y difícil mantenimiento. Tailwind CSS soluciona este problema mediante la creación de clases aplicables directamente sobre los elementos HTML, evitando la aglomeración de código de estilo en archivos CSS a la vez que se agiliza el desarrollo y se facilita el mantenimiento. Tailwind CSS se ha utilizado en numerosas estructuras visuales dentro de TimeFlex, ya que se han adaptado sus diseños predefinidos. Ejemplos de ello son la página de preguntas frecuentes (ver Figura 4-9) o la página "Sobre nosotros".

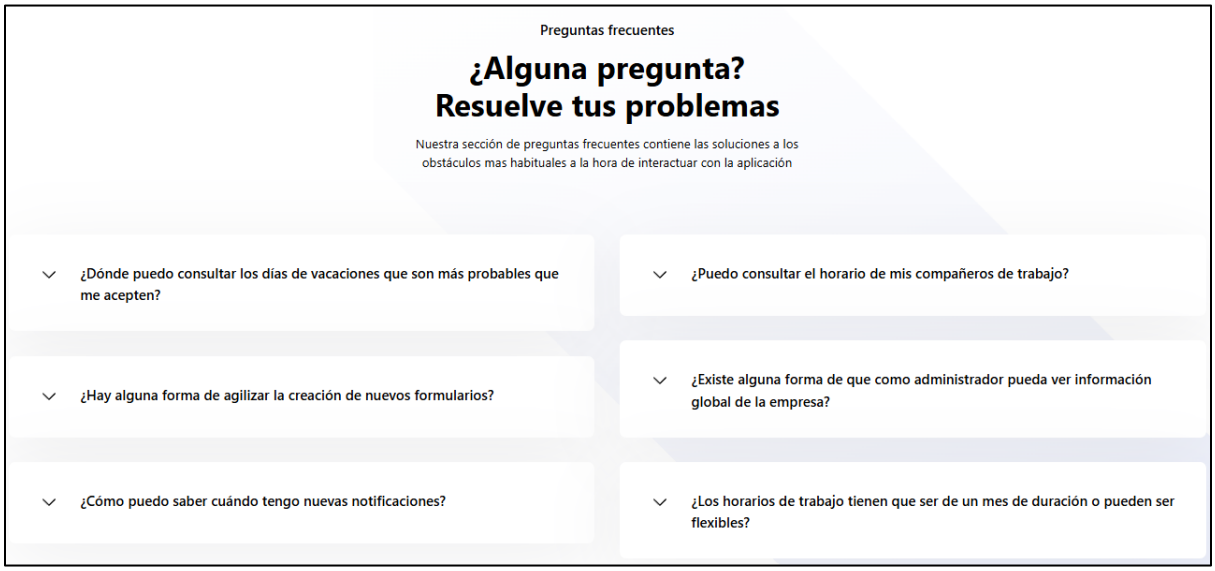

Figura 4-9. Página de preguntas frecuentes utilizando un diseño adaptado de Tailwind CSS.

Con respecto al módulo de estadísticas de la aplicación, surge la necesidad de definir una herramienta para presentar y mostrar estadísticas a los usuarios. Teniendo esto en cuenta, se decidió llevar a cabo una investigación exhaustiva comparando las ventajas e inconvenientes de una serie de posibles candidatas. Las herramientas analizadas fueron:

- **Highcharts** [26]: Biblioteca de JavaScript enfocada a mostrar datos y estadísticas de forma interactiva con SVG [86] que goza de gran

popularidad y es ampliamente utilizada por empresas líderes de sus sectores como Netflix, SAP, Visa o Google.

- **Matplotlib** [40]: Biblioteca de Python orientada a la visualización de datos y creación de gráficos.
- **CanvasJS** [8]: Biblioteca de JavaScript [68] [95] que permite crear gráficos interactivos mediante el uso del elemento Canvas [29].
- **Chart.js** [15]: Biblioteca de JavaScript que permite crear gráficos utilizando también el elemento Canvas.

La Tabla 4-2 compara el rendimiento de las de las herramientas descritas anteriormente (filas) en relación con una serie de atributos relevantes para el proyecto de TimeFlex (columnas).

	Highcharts	**Matplotlib**	**CanvasJS**	**Chart.js**
Gráficos interactivos	Muy alta	Compleja y tediosa implementación	Alta	Alta
Datos dinámicos	Alta	Requiere complejidad adicional	Alta	Alta
Variedad de gráficos	Muy alta	Moderada	Alta	Alta
Personalización	Muy alta	Alta	Alta	Moderada
Integración	Alta	Baja	Alta	Muy alta
Facilidad de uso	Alta	Moderada	Alta	Alta
Documentación	Muy extensa	Muy extensa	Extensa	Extensa

Lenguaje	JavaScript	Python	JavaScript	JavaScript
TypeScript	Sí	No	Sí	Sí

Tabla 4-2. Comparativa entre distintas herramientas de visualización de estadísticas.

Debido a la comparativa descrita en la Tabla 4-2, Highcharts [26] resultó elegida como la opción más adecuada para el proyecto de TimeFlex. Esto fue debido a su destacado rendimiento en aspectos prioritarios para el equipo: excelente experiencia interactiva por parte del usuario y sencilla integración de datos dinámicos en los gráficos.

Además, Highcharts ofrece la posibilidad de utilizar TypeScript [70], extensión de JavaScript que añade sintaxis adicional a este lenguaje permitiendo una mejor integración y detección de errores. Todo el código elaborado en TypeScript se transpila a JavaScript [95], por lo que es adaptable a la multitud de plataformas y servicios que soportan JavaScript. Su sistema de tipado estático reduce significativamente el número de bugs. Esto se consigue mediante la declaración de variables de forma explícita, puesto que los errores se detectan en tiempo de compilación en lugar de en tiempo de ejecución. También supone una herramienta ideal para proyectos de gran extensión como TimeFlex, puesto que su escalabilidad facilita la refactorización y mantenimiento de código.

Todos los factores estudiados han llevado a elegir Highcharts por delante del resto de alternativas para elaborar enriquecedores gráficos que muestren información valiosa y diversa a los usuarios permitiéndoles tomar decisiones informadas. A modo de resumen, se presentan las principales ventajas de Highcharts, herramienta elegida para la visualización gráfica de las estadísticas:

- Alto nivel de interactividad que permite al usuario personalizar en cada momento qué partes del gráfico quiere ver y consultar los valores de un elemento particular de la serie.

- Enorme variedad de gráficos: circulares, sectoriales, lineales, áreas, *donuts*, *heatmaps* e incluso *dashboards*.
- Integración sencilla de datos dinámicos mediante *endpoints* con el objetivo de mostrar siempre al usuario datos actualizados.
- Adaptable a cualquier base de datos o servidor.
- Diseño responsive ideal para una adecuada visualización en cualquier dispositivo.
- Herramienta gratuita para proyectos académicos.

A continuación, se muestran comparativas detalladas entre Highcharts y el resto de las alternativas con el fin de justificar la elección realizada.

Comparativa entre Highcharts y Chart.js. Highcharts [26] dispone de una enorme variedad de gráficos, mientras que Chart.js a pesar de disponer de un catálogo también considerable, no dispone de ciertos tipos de gráficos (3D, *Gantt, SVG, Bullet*). Adicionalmente, Highcharts ofrece la posibilidad de exportar y combinar varios gráficos, permitiendo una visualización más flexible que Chart.js. La documentación de Highcharts es otro de sus puntos fuertes, ya que ofrecen a disposición del usuario ejemplos de código y una extensa documentación gracias a una comunidad ciertamente grande en relación con su sector. Por otro lado, Chart.js [15] ofrece una mejor y más ligera compatibilidad con *frameworks* modernos con respecto a Highcharts, que suele requerir un mayor esfuerzo para llevar a cabo estas integraciones [27].

Comparativa entre Highcharts y Matplotlib. Matplotlib [40] es una biblioteca de Python que permite una visualización de datos personalizada, pero requiere actualizaciones manuales complejas para manejar datos dinámicos. Highcharts [26], en cambio, permite la integración de datos dinámicos de forma sencilla mediante endpoints, lo cual supone una ventaja significativa. Por otro lado, Highcharts puede ser usado en cualquier navegador mientras que Matplotlib, que es una biblioteca de Python, está más limitado a aplicaciones basadas en servidor y de escritorio. Matplotlib requiere de un mayor esfuerzo y codificación para obtener resultados similares, y además no ofrece una gama tan extensa de gráficos en comparación con Highcharts. Este último además de ofrecer un alto nivel de personalización también facilita la

implementación de elementos relevantes para el proyecto de TimeFlex como *dashboards* [28].

Comparativa entre Highcharts y CanvasJS. CanvasJS [8] dispone de cierta ventaja en términos de rendimiento con respecto a Highcharts [26] debido al uso de Canvas [29] frente a SVG [86], ya que Canvas es más rápido y eficiente. Sin embargo, Highcharts ofrece un mayor catálogo de gráficos y posibilidades de personalización más extensas y avanzadas. En términos de documentación, Highcharts es también una opción más competente puesto que su comunidad es más grande y, por lo tanto, dispone de más información acerca de sus bibliotecas y funcionalidades [7].

4.1.1.2 Capa de lógica de negocio

La capa de lógica de negocio es la encargada de gestionar la lógica y las funcionalidades principales de la aplicación. Esta interactúa con la base de datos y la capa de presentación, transmitiendo la información necesaria en los casos adecuados según las reglas de negocio establecidas.

Como se ha mencionado en la Sección 4.1.1, Laravel es el *framework* elegido para implementar la capa de negocio, aprovechando las múltiples ventajas que presenta frente a otros *frameworks* de PHP.

En términos de seguridad, Laravel proporciona mecanismos de seguridad configurables, destacando su sistema de autenticación que utiliza sesiones y cookies para aplicaciones web, y tokens para APIs. Por otro lado, para la conexión con APIs externas, el paquete Sanctum [37] permite una autenticación híbrida que mejora la seguridad al separar ambos métodos. Además, Laravel ofrece herramientas de *hashing* como Argon2 [77] para proteger las contraseñas almacenadas en la base de datos, junto con funciones adicionales como "*check*" para verificar la integridad de los hashes. En la Sección 6.3 se presenta en profundidad el sistema de seguridad de Laravel.

Además, Laravel incluye un sistema de middleware [42] configurable que actúa como filtro de seguridad para las rutas definidas. Este sistema no solo permite verificar si el usuario está autenticado antes de acceder a determinadas solicitudes, sino que también comprueba si el rol asignado al usuario habilita el acceso a funcionalidades

específicas, permitiendo gestionar vistas y accesos diferenciados según el tipo de usuario.

En la aplicación desarrollada, no solo se ha empleado el sistema de middlewares para la autenticación, sino que también se ha utilizado para almacenar un registro de los últimos enlaces visitados por el usuario permitiendo así ofrecer accesos directos a las páginas más recientes, mejorando la usabilidad de la aplicación.

Otro aspecto clave de esta capa es la interacción con la base de datos, realizada a través del modelo Eloquent de Laravel. Eloquent [19] es un ORM (*Object-Relational Mapper*), elemento que permite mapear el contenido de la base de datos del proyecto e interactuar con dicho contenido como si fuera un objeto. Esto se traduce en la generación de modelos específicos para cada tabla de la base de datos, utilizados a la hora de insertar, actualizar o eliminar elementos de la misma.

Esta funcionalidad facilita el desarrollo de código de *backend*, permitiendo un control más eficiente y productivo de las operaciones asociadas. Además, *Eloquent* brinda la oportunidad de emplear herramientas como por ejemplo: (i) *timestamps* para conocer la fecha de creación y última modificación, (ii) un constructor de consultas a la base de datos con métodos específicos para obtener información y añadir restricciones, y (iii) los métodos *fresh* y *refresh* que permiten obtener datos actualizados ya sea creando nuevas instancias del modelo (*fresh*) o actualizando la instancia actual y sus relaciones (*refresh*). Aunque solo se hayan mencionado los métodos más relevantes, existe una extensa gama de recursos adicionales ofrecida por *Eloquent*, que otorga a TimeFlex un gran manejo de datos.

La capa de negocio de TimeFlex se ha validado mediante el completo sistema de pruebas de Laravel, que incluye el uso de *seeders* para generar datos iniciales automáticamente y simular un entorno real de uso. El uso de estas herramientas se detalla en la Sección 6.4.

Por otro lado, los formularios son elementos que aparecen de forma habitual en las aplicaciones web con el fin de obtener información de los usuarios. Laravel permite el desarrollo de estos mediante la función "*form*", que permite construir cuestiones para el usuario de forma agrupada, y el método "*submit*", que devuelve un array indexado

numéricamente con las respuestas recibidas asociadas al formulario creado con "*form*". Adicionalmente, se ofrece la posibilidad de otorgar un nombre a cada cuestión, de forma que posteriormente se pueda acceder a la respuesta de cada cuestión usando como índice su nombre en vez del número. Además, permite llevar un control sobre las acciones del usuario, pudiendo restaurar versiones anteriores del mismo sin necesidad de reiniciar el formulario entero en caso de tener que corregir errores o modificar respuestas.

Para el desarrollo de la capa de negocio de TimeFlex se ha optado por utilizar SQLite [62] [83], una base de datos ligera que almacena toda la información en un único archivo, facilitando así la portabilidad y autonomía del sistema sin necesidad de un servidor separado. Aunque SQLite es adecuada para esta fase, el código está diseñado para aislar la gestión de la base de datos, permitiendo un cambio futuro a otra tecnología sin afectar al resto de la aplicación. Posteriormente, en la Sección 4.2 se hablará en detalle sobre el diseño y uso de las bases de datos en el proyecto.

4.1.2 Capa de servicios

El microservicio de optimización es una funcionalidad clave y una pieza con gran importancia dentro de la arquitectura de la aplicación. Su objetivo es generar horarios personalizados en función de las preferencias, disponibilidad horaria y restricciones definidas por los empleados y la empresa, de forma aislada del *backend* principal en forma de servicio independiente.

Inicialmente, se valoró desarrollar la lógica de optimización mediante WebAssembly [75], permitiendo así ejecutar el código directamente desde el navegador del cliente, reduciendo la carga del *backend* y aumentando la seguridad y privacidad. Sin embargo, tras un análisis más profundo, esta idea fue descartada debido a varias razones. En primer lugar, la integración de librerías supone un proceso complicado y de difícil mantenimiento. Además, trasladar el proceso al frontend, introduce sobrecarga en el dispositivo del cliente, lo cual afecta negativamente a la experiencia de usuario, y a la escalabilidad de la herramienta ya que dependerá del software donde se ejecute el frontend.

Una vez descartada la opción de ejecutarlo en el cliente, se contempló integrarlo directamente dentro del *backend* principal, simplificando así el despliegue. Sin embargo, en lugar de integrar la lógica de horarios dentro del *backend* Laravel, se optó por extraer esta parte en un microservicio independiente. Esta decisión de diseño permite la separación de responsabilidades, de modo que tanto el *backend* principal como el microservicio de optimización puedan evolucionar de manera independiente, minimizando el riesgo de afectarse mutuamente de forma negativa. Al desacoplar la lógica de optimización, cabe también la posibilidad de utilizar herramientas y lenguajes más adecuados, como Python junto al resolutor Z3 [76] [87]. De igual manera, permite que el *backend* principal se comunique con el microservicio a través de una API, pudiendo sustituir este módulo por otros en caso de ser necesario.

Por otro lado, desde un punto de vista de rendimiento y escalabilidad, aislar la lógica en un microservicio, facilita el despliegue del *backend* principal y el microservicio en distintos entornos según sus necesidades específicas, evitando que procesos intensivos del optimizador afecte al rendimiento general de la plataforma.

Para desarrollar el microservicio se han utilizado las siguientes tecnologías:

FastAPI

Para la implementación del microservicio se ha elegido FastAPI [21], un *framework* para el desarrollo de APIs en Python que permite un rendimiento excelente debido a estar construido sobre Starlette [64] . Starlatte es un *microframework* que permite la gestión de solicitudes HTTP y HTTPS de forma asíncrona. Gracias a esto, el sistema puede manejar muchas peticiones al mismo tiempo sin bloquearse, lo que mejora notablemente la eficiencia.

Además, FastAPI integra de forma nativa Pydantic [55], una librería que permite definir los datos que espera recibir el sistema, validarlos automáticamente y convertirlos al tipo correcto, mejorando la seguridad y reduciendo errores en tiempo de ejecución.

Dentro de Python existen varios frameworks para construir APIs, siendo Flask [22] y Django [17] dos de las más populares. Flask destaca por ser un framework simple y flexible mientras que Django es un framework menos ligero, más orientado al desarrollo de aplicaciones web completas, con muchas funcionalidades que pueden ser innecesarias para un microservicio.

La Tabla 4-3 muestra de forma gráfica las diferencias existentes entre FastAPI, Flask, y Django.

	FastAPI [21]	Flask [22]	Django [17]
Asincronía nativa	Sí	No	Sí
Validación de datos automática nativa	Sí	No	No
Generación automática de documentación	Sí	No	No
Complejidad	Baja	Baja	Alta

Tabla 4-3. Comparativa entre distintos frameworks de desarrollo de APIs.

Teniendo en cuenta estas diferencias, FastAPI se presentó como la opción más adecuada para el microservicio.

Z3

El núcleo del microservicio está basado en Z3 [76] [87], un "theorem prover" diseñado por Microsoft Research que permite resolver problemas de satisfacción y restricciones, también llamados CSP (Constraint Satisfaction Problems).

En el proceso, los datos que introducen tanto los empleados como los administradores, respuestas de formularios, asignaciones de turnos, etc, se convierten en variables y restricciones lógicas. Cada turno, preferencia o limitación queda modelada de manera que el motor de optimización de Z3, pueda trabajar con ellas y buscar una asignación de valores a las variables que satisfaga todas las condiciones de la mejor forma posible, intentando optimizar el reparto de turnos respetando las preferencias y limitaciones planteadas.

Entre las principales ventajas de Z3 destaca su alta expresividad, permitiendo definir restricciones muy complejas de manera natural y compacta. Otra ventaja es su alto rendimiento a pesar de la complejidad del problema (cientos de turnos, trabajadores, preferencias, etc), es capaz de encontrar soluciones optimizadas en tiempos muy reducidos. Y finalmente, otra gran ventaja es que Z3 dispone de una API oficial para Python. Esta compatibilidad ha permitido integrarlo de forma directa en el microservicio desarrollado con FastAPI.

4.2 Estructura de la base de datos

La base de datos de este proyecto se ha construido alrededor de la herramienta de software libre SQLite [62] [83] . Esto permite almacenar la información requerida para el correcto funcionamiento de la web implementando el lenguaje SQL [97] para realizar todo tipo de consultas. Además, la base de datos completa se encuentra almacenada en un solo archivo.

SQLite es un tipo de base de datos relacional, las cuales se basan en un conjunto de tablas. Las tablas contienen la información almacenada en filas, columnas y relaciones. Estas relaciones definidas conectan los datos entre diferentes tablas. Por otro lado, las bases de datos no relacionales se caracterizan por no depender de un esquema fijo como las bases de datos relacionales, pudiendo almacenar datos de múltiples formas (documentos, pares clave-valor, grafos o columnas).

Se ha decidido implementar bases de datos relacionales porque se garantiza una consistencia e integridad de los datos y una estructura clara y definida, a diferencia que las no relacionales. Estas últimas están pensadas para aplicaciones que manejan una gran cantidad de datos, mientras que la integridad de los datos es más exigente debido al esquema variable o la falta de estandarización de lenguajes debido a la amplia cantidad de distintas soluciones que existen como por ejemplo XML [96] , MongoDB [43], o Datalog [90].

Dentro de las bases de datos relacionales, se ha decidido utilizar SQLite [62] [83] en lugar de otras herramientas como MySQL [44] o PostgreSQL [53]. SQLite requiere poco rendimiento y memoria del servidor, es autónomo y portátil y está incluido por

defecto en todas las instalaciones de PHP [52]. Por el contrario, MySQL [44] requiere mucha memoria y requiere un servidor para funcionar.

Dentro del proyecto, las migraciones de las tablas están organizadas de forma independiente del resto del código. Cada tabla cuenta con su propio archivo de migración, donde se define su estructura, las relaciones con otras tablas y las reglas para su eliminación. En caso de que una tabla principal sea eliminada, las tablas relacionadas también se suprimen, lo que permite gestionar de manera controlada los posibles errores derivados de estas dependencias.

Además, todas las consultas se gestionan de forma estructurada y desacoplada de la lógica de presentación. Para ello, se utiliza un *Query Builder*, que no solo permite realizar las consultas de manera aislada, facilitando el mantenimiento de código, sino que también facilita en un futuro sustituir la tecnología subyacente sin afectar al resto del sistema.

Todo esto permite poder adaptar el proyecto de forma sencilla a las necesidades del negocio, ya que gracias a tener centralizadas tanto las migraciones como las consultas a la base de datos, se podrá realizar un cambio de tipo de base de datos sin que conlleve un desarrollo largo y costoso.

La estructura de tablas de la base de datos es la siguiente:

- **Users:** La Tabla 4-4 contiene las tablas relacionadas con las secciones. Almacena los atributos principales de los usuarios, destacando el correo electrónico, que también se utiliza como identificador único para el inicio de sesión. No obstante, este puede modificarse gracias al uso de un identificador único (id) no editable.

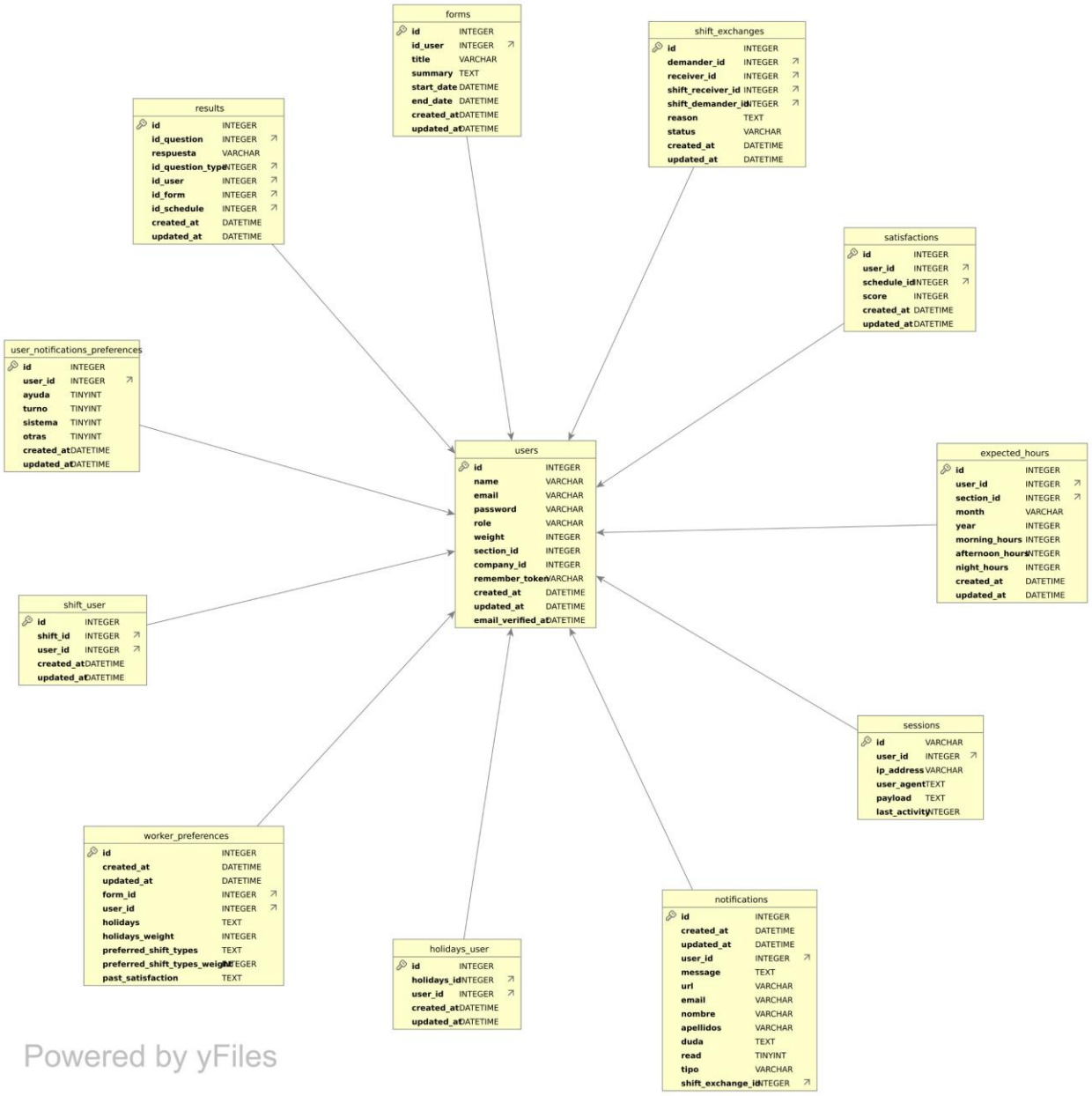

Tabla 4-4. Tabla users y sus relaciones.

- **Sections:** La Tabla 4-5 contiene las tablas relacionadas con las secciones. Contiene los valores de mínimo/máximo de horas y mínimo/máximo de turnos que se utilizarán en la API de generación de turnos.

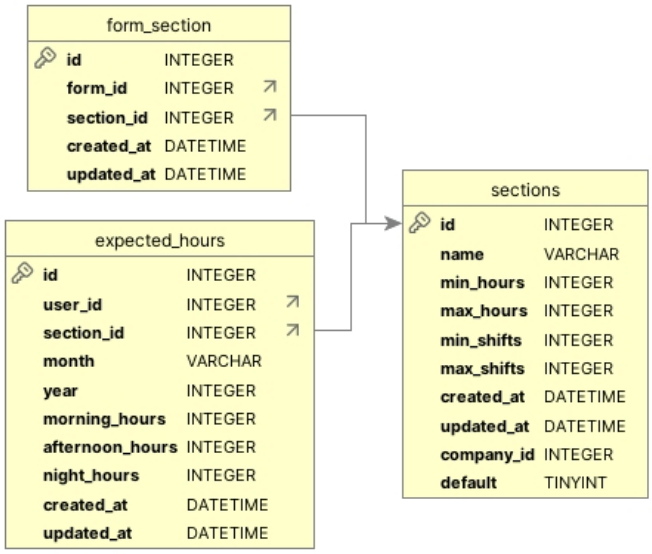

Tabla 4-5. Tabla sections y sus relaciones.

- **Forms:** La Tabla 4-6 muestra las tablas relacionadas con los formularios. Guarda la información de los formularios creados para que los empleados de las distintas secciones de la empresa contesten a estos. Se relaciona con las tablas de preguntas y respuestas, pudiendo así acceder a cualquier valor desde la tabla principal *forms*.

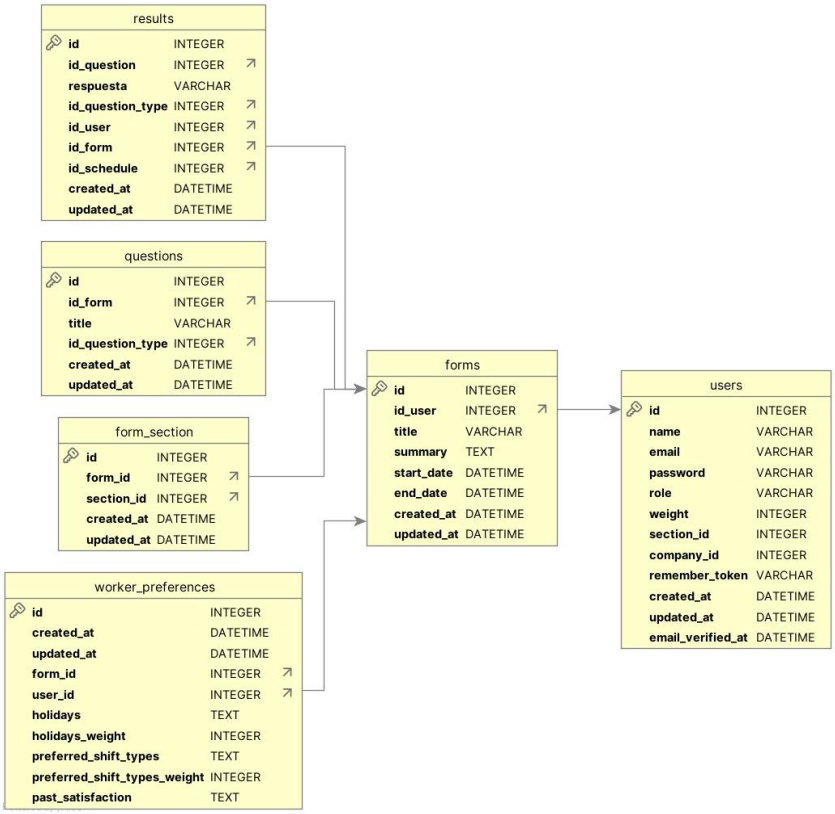

Tabla 4-6. Tabla forms y sus relaciones.

• **Questions:** La Tabla 4-7 muestra la tabla *questions*, relacionada con la tabla *forms*, guarda todas las preguntas que se crean en el proceso de alta de un nuevo formulario, distinguiéndose según el tipo de cada una. Si la pregunta es de tipo vacaciones o turnos, los valores adicionales relacionados con estas se almacenan en la tabla *weights*, donde se registran los pesos o valores asignados a dichas preguntas. Por otro lado, si la pregunta es de tipo opción simple u opción múltiple, las opciones correspondientes se guardan en la tabla *options*, permitiendo una configuración dinámica de las posibles respuestas

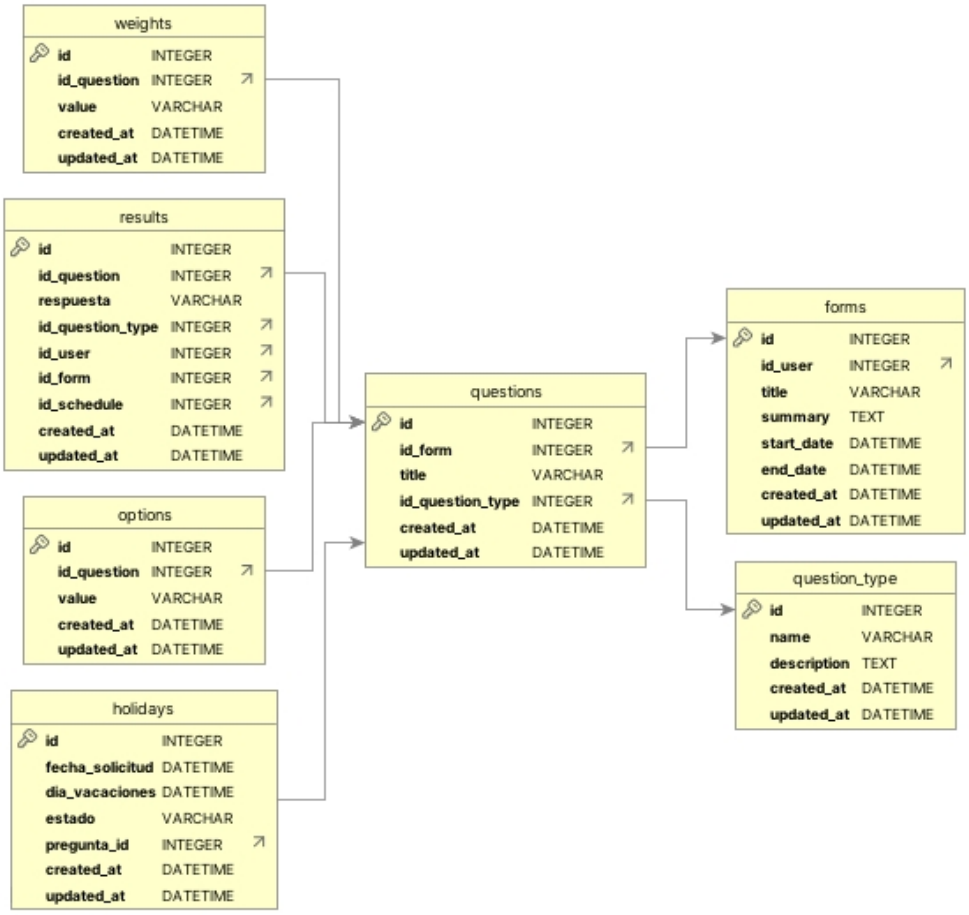

Tabla 4-7. Tabla questions y sus relaciones.

- **Schedules:** La Tabla 4-8 muestra cómo la tabla *schedule* se encarga de almacenar los valores relacionados a los horarios propios de cada sección, donde destaca la relación con los turnos que muestra los diferentes intervalos de tiempo disponibles para cada día. A su vez se almacena el grado de satisfacción de los turnos de cada empleado en relación con las preferencias introducidas mediante formularios.

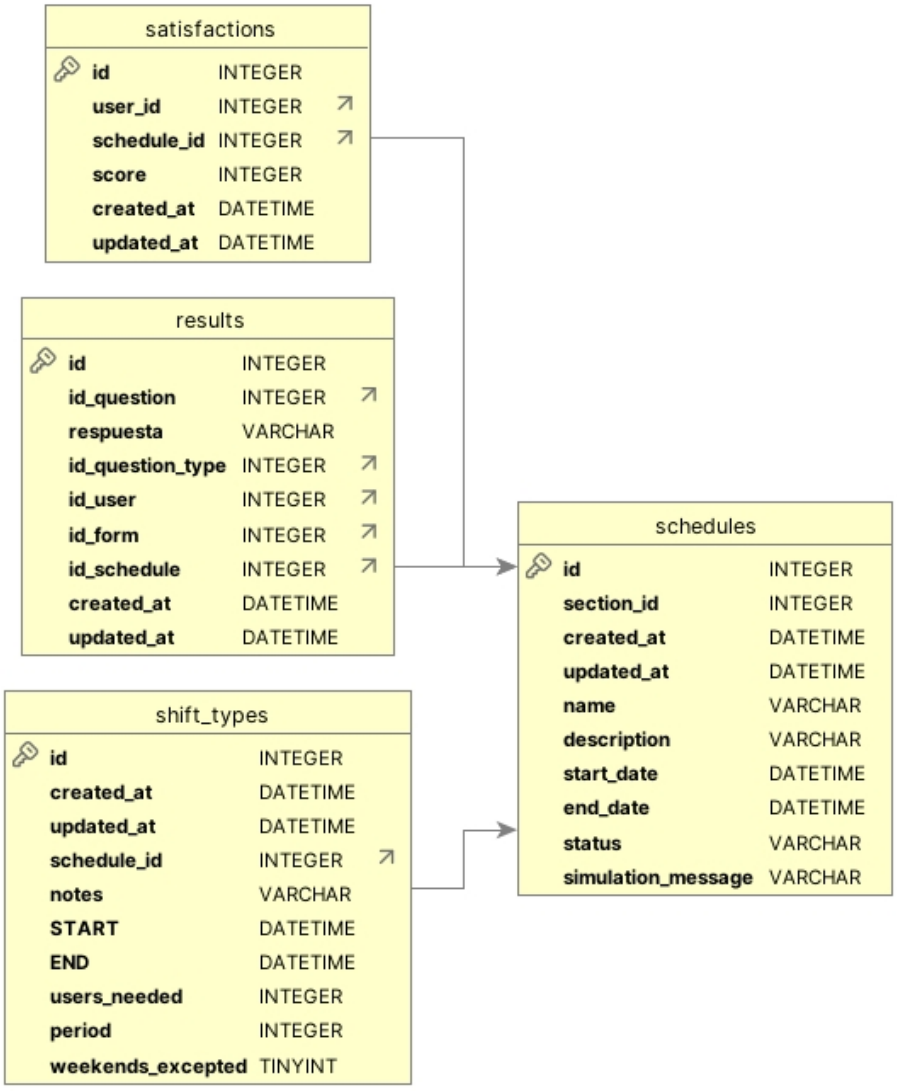

Tabla 4-8. Tabla schedules y sus relaciones.

- **Shifts:** La Tabla 4-9 muestra el conjunto de tablas que representa el contenido de cada turno que contiene un horario, cuya particularidad es la petición de cambio al administrador mediante notificaciones.

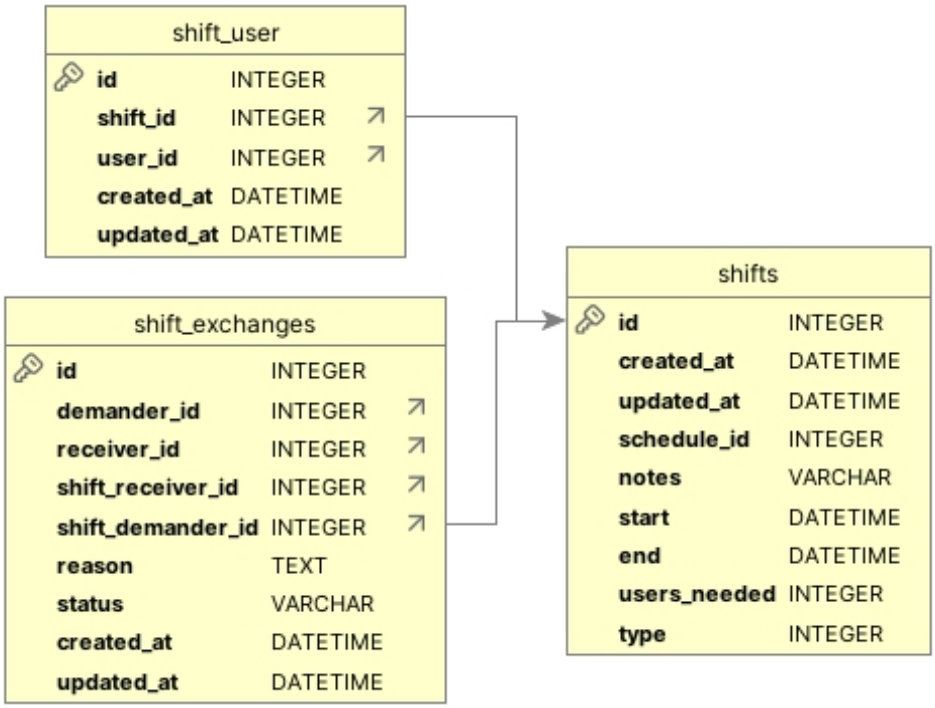

Tabla 4-9. Tabla shifts y sus relaciones.

La Figura 4-10 muestra un diagrama completo de la estructura de la base de datos y sus distintas relaciones.

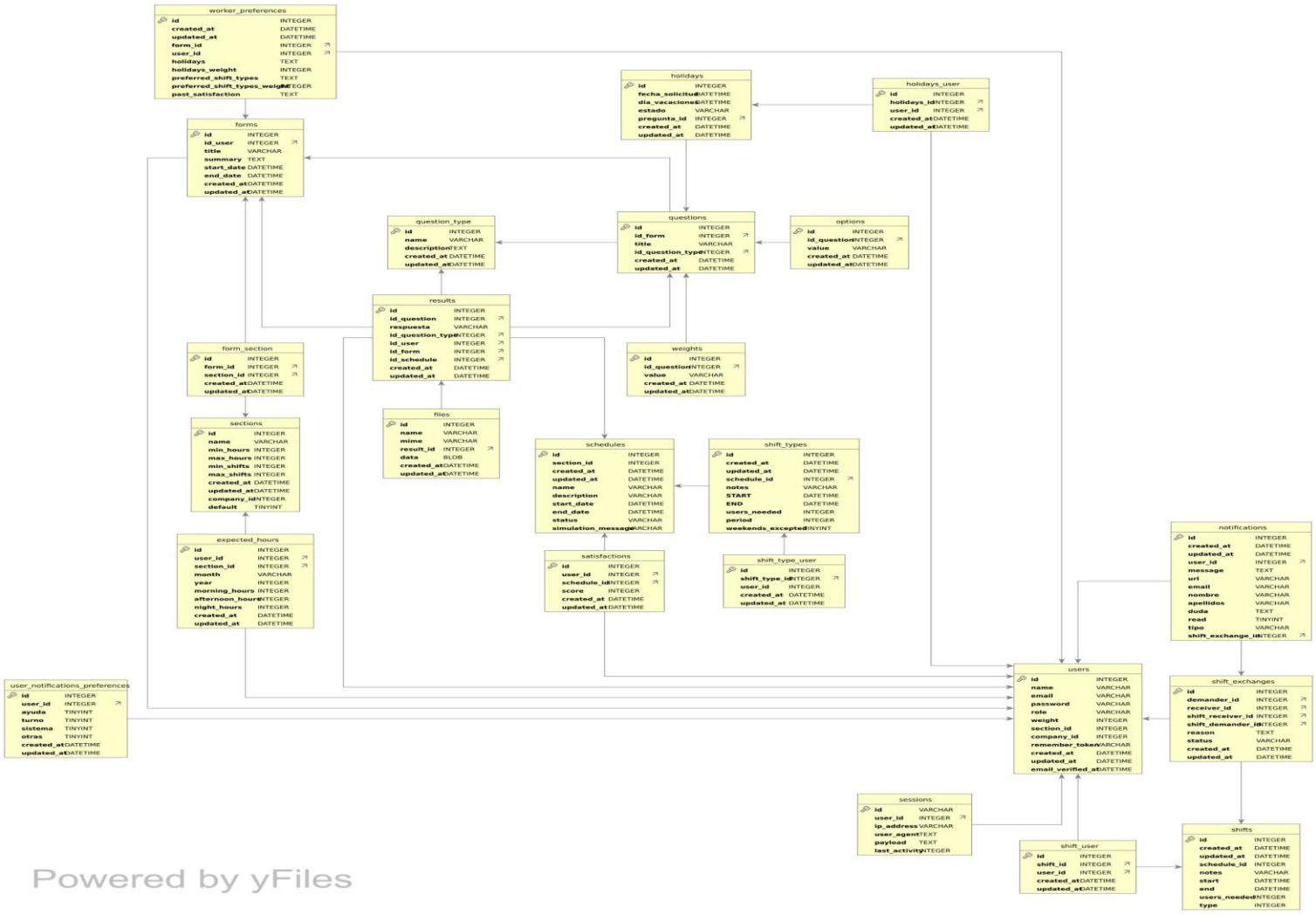

Figura 4-10. Diagrama completo de la base de datos.

Capítulo 5 - Descripción funcional de los módulos

Las funcionalidades de las que dispone la plataforma exponen las principales capacidades de la aplicación orientadas a la optimización de la gestión de turnos y horarios. Estas funcionalidades han sido diseñadas con el propósito de atender las necesidades específicas tanto de los empleados como de los personales de RRHH, proporcionando una experiencia de uso intuitiva y eficiente. Entre sus características más destacadas se encuentran la generación automática de horarios personalizados, la administración de solicitudes de cambio de turno, la elaboración de estadísticas detalladas y la configuración de notificaciones ajustables a las preferencias del usuario. La Sección 5.1 presenta el módulo de calendarios y horarios con sus diferentes acciones, la Sección 5.2 describe el módulo de formularios comprendiendo la creación y cumplimentación de los mismos, la Sección 5.3 detalla la personalización y visualización de estadísticas, y la Sección 5.4 comprende las funcionalidades asociadas al módulo de notificaciones.

5.1 Calendarios

La funcionalidad de gestión de calendarios constituye el núcleo de la aplicación, donde se detalla cómo se gestionan los horarios, desde su creación hasta su visualización y modificación, abarcando tanto las necesidades de los empleados como los encargados de la administración de la plataforma.

Generación de horarios

La generación de horarios está diseñada para optimizar la asignación de turnos de los empleados registrados en la base de datos de la aplicación. Los empleados tienen la opción de introducir sus preferencias mediante formularios, lo que permite personalizar los resultados de la optimización. Además, el sistema evalúa la satisfacción de los empleados con los horarios generados, tomando en cuenta la información proporcionada en los formularios. Para facilitar la identificación de posibles problemas durante la generación, se incluye el modo "análisis", que permite analizar en detalle qué restricciones están impidiendo la creación de un horario válido.

Este modo funciona asociando un identificador a cada restricción del sistema, de modo que cuando no existe una solución posible, el sistema es capaz de detectar cuáles de esas restricciones no pueden cumplirse simultáneamente y proporciona esa información al administrador. Gracias a este enfoque, es posible localizar con rapidez las incompatibilidades entre disponibilidades, preferencias o límites establecidos por la organización, reduciendo significativamente el tiempo de diagnóstico y ajuste, a la vez que mejora la experiencia de administración.

Visualización de horarios

La visualización de horarios permite a los usuarios consultar los turnos asignados de manera gráfica, clara y estructurada, facilitando la planificación y organización tanto para los empleados como para los responsables de RRHH. Esta funcionalidad se ha diseñado teniendo en cuenta la usabilidad, la flexibilidad y la personalización, asegurando que los usuarios puedan acceder a la información que necesitan de forma rápida y eficiente.

El calendario ofrece diferentes vistas adaptadas a las necesidades de cada tipo de usuario. Entre ellas se incluyen:

1. **Vista global de horarios**

   Permite a los administradores visualizar todos los turnos en un calendario general. Los turnos se muestran con códigos de colores para diferenciar los tipos de turno (mañana, tarde, noche) y resaltar posibles conflictos o vacantes.

2. **Vista individual de horarios**

   Cada empleado puede acceder a su calendario personal, donde se muestran únicamente los turnos que le han sido asignados. Los empleados pueden consultar sus días libres, vacaciones aprobadas y solicitudes pendientes.

3. **Vista de turnos específicos**

   Los administradores pueden seleccionar un turno concreto para ver qué empleados están asignados a él. Esta vista es útil para identificar rápidamente si un turno tiene suficientes trabajadores asignados o si requiere ajustes.

La visualización de horarios no se limita a mostrar información estática, sino que incluye elementos interactivos que mejoran la experiencia del usuario, como la interacción con los turnos, donde los usuarios pueden obtener más detalles o realizar acciones, como solicitar un cambio de turno o añadirlo a Google Calendar [16] o Outlook [51].

Modificación de horarios

La modificación de horarios permite editar las características principales de estos, como son el título, la descripción, las fechas de inicio, fecha de fin y además soporta la gestión de los turnos de cada horario.

En relación a los turnos, se encuentran disponibles las funcionalidades de añadir turnos, la cual es la encargada de realizar el alta de un nuevo turno tras indicar el inicio y fin del turno, su descripción, el número de trabajadores necesarios, el periodo de repetición a lo largo del tiempo, opciones para evitar los fines de semana y la asignación opcional de trabajadores al turno. Además, una vez dado de alta el horario permite editarlo y modificarlo.

Por último, también permite regenerar los turnos, cuya finalidad es aplicar los cambios realizados en los turnos en el horario correspondiente al que afecte y deshace todas las asignaciones de turnos realizadas por el optimizador.

Cambios de turno

Los cambios de turno permiten a los usuarios intercambiar franjas horarias entre sí, facilitando la organización de los horarios de acuerdo con sus preferencias.

Este proceso se realizará en una vista que contiene un calendario que muestra los turnos que tiene el usuario asignado en el horario, y un formulario con los campos a rellenar del turno a cambiar, turno nuevo y motivo de cambio. Una vez introducidos y pulsado el botón de 'Registrar cambio de turno', se le enviará una notificación al administrador de la petición de cambio, pudiendo este aceptarla o rechazarla.

5.2 Formularios

Uno de los aspectos más importantes de TimeFlex es la capacidad de crear y administrar formularios de manera eficiente. En este módulo, se permite recopilar datos estructurados de los usuarios, personalizar preguntas e integrar las respuestas con la generación automática de calendarios para que los turnos asignados a los empleados sean lo más parecidos posibles a las preferencias de los empleados introducidas en los formularios.

Creación de formularios

Permite a los empleados de recursos humanos diseñar formularios desde cero con una interfaz intuitiva y herramientas dinámicas. Estos usuarios pueden configurar el título, un breve resumen descriptivo, las fechas de inicio y finalización y la posibilidad de seleccionar secciones específicas de la empresa para que sólo los integrantes de estas puedan contestar. El módulo tiene nueve tipos de preguntas que pueden utilizarse para personalizar los formularios mediante la creación de preguntas de forma dinámica:

1. **Calendario**

   Muestra un calendario donde se pide que se seleccione un intervalo de fechas.

2. **Selector**

   Pregunta donde se ofrecen un número de opciones para elegir una de ellas.

3. **Gradual**

   Muestra una barra de control deslizante para seleccionar un valor entre 1 y 100.

4. **Turnos**

   Muestra los turnos disponibles del calendario con el que el formulario está relacionado. Las respuestas de esta pregunta se utilizan para la generación automática de turnos.

5. **Vacaciones**

   Muestra un calendario con un mapa de calor integrado que indica los días con más frecuencia de petición donde se seleccionan los días de vacaciones que el usuario quiera solicitar.

6. **Texto libre**

   Permite introducir una respuesta abierta a la pregunta propuesta.

7. **Opción múltiple**

   Pregunta donde se ofrecen un número de opciones para elegir múltiples respuestas.

8. **Numérica**

   El formato de la respuesta únicamente puede ser numérico.

9. **Carga de archivo**

   Permite que el usuario pueda subir un archivo de su dispositivo, almacenándolo en la base de datos.

Modificación de formularios

La funcionalidad de editar formularios permite editar formularios existentes, asegurándose que se puedan adaptar a cambios en los requisitos de creación de formularios o corrección de errores a la hora de dar el alta.

Además, dispone de la opción de publicar un formulario, realizándose una copia de este con los mismos atributos, modificando únicamente la fecha de inicio y la fecha de finalización. Esto se lleva a cabo a través de una ventana modal previa a la generación del duplicado.

Responder formulario

Es la funcionalidad principal de este módulo ya que es fundamental para la interacción entre los usuarios de la aplicación y los formularios creados. Los empleados que pertenezcan a las secciones que fueron asignadas durante la creación del formulario y se encuentren dentro de la franja de fechas de inicio y finalización del tiempo de respuesta pueden acceder y contestar las múltiples preguntas contenidas en el formulario.

Una vez completado el formulario, las respuestas se guardan en la base de datos, utilizándose para recopilar estadísticas o preferencias de los empleados. Además, mientras se siga estando dentro del plazo de validez, se pueden editar las respuestas dadas anteriores, actualizando correctamente.

Panel de control de respuestas

Finalmente, el panel de control de respuestas es una vista exclusiva para los empleados de RRHH, donde pueden analizar y gestionar los datos recopilados. En este panel se muestran todas las respuestas de todos los formularios completados por los usuarios, permitiendo visualizar la información sin necesidad de abandonar la web ni realizar consultas directas a la base de datos.

Además, ofrece una serie de filtros avanzados de búsqueda, como el título del formulario, usuario, sección o fecha. Todo esto favorece la optimización del tiempo de consulta, permitiendo realizarla de manera más rápida y eficaz.

5.3 Estadísticas

La sección de estadísticas es una de las más extensas de la aplicación web de TimeFlex, debido en gran parte a la existente necesidad por visualizar de forma intuitiva las principales métricas del área de negocio de la aplicación.

Como se ha mencionado a lo largo de este documento, TimeFlex se especializa en la generación de horarios y turnos, por lo que establecer un control exhaustivo de la asignación de los mismos es un requisito imprescindible para asegurar la excelencia en la prestación del servicio. Mediante la implementación de estadísticas que evalúen el desempeño y evolución de la satisfacción de los empleados o el número de turnos y horas trabajadas, se consigue una visión global del negocio y del rendimiento de la aplicación.

Hay múltiples vistas dentro de la aplicación dedicadas a la sección de estadísticas. Una de ellas es la vista llamada "Estadísticas", que ofrece una serie de gráficos interactivos con información relevante asociada al empleado en cuestión. El primero de ellos muestra el número de horas trabajadas por el empleado a lo largo del año en cada mes, mientras que el segundo compara mes a mes la evolución de los cambios de turno solicitados frente a los que han sido aceptados. También existe un gráfico que permite ver la evolución de las solicitudes de vacaciones por mes, mostrando de nuevo aquellas solicitadas y aquellas aceptadas. Finalmente, se ofrece una última gráfica que muestra la satisfacción mensual respecto a la cantidad y tipo de turnos del usuario en un moderno formato de burbuja. Todos estos gráficos sirven al empleado para disponer de un amplio conocimiento de la evolución de sus solicitudes de cambio de turno y vacaciones, además de las horas y tipos de turno trabajados, y de qué manera impactan en su satisfacción.

De forma adicional, existe un apartado de estadísticas personalizadas e interactivas para cada usuario de la aplicación. En caso de ser un empleado raso, se podrán visualizar aquellas estadísticas asociadas a los compañeros de sección, mientras que si se trata del administrador, se podrá visualizar las de todos los empleados. Una vez se accede a dicha sección, podemos visualizar dos gráficos que aportan información de los turnos del empleado seleccionado: el primero se trata de un diagrama circular

de sectores que muestra la distribución de los turnos del usuario entre mañana, tarde y noche. Por otro lado, el segundo permite visualizar el número de horas trabajadas y el número de horas esperadas por cada tipo de turno. Ambos gráficos muestran información del mes actual, permitiendo un seguimiento por parte del usuario de sus estadísticas y la de sus compañeros, e incluso de toda la empresa en el caso del administrador.

Dentro de este módulo, también está disponible la visualización de cómo ha evolucionado la satisfacción del empleado actual y el de su sección, comparando ambos en esta estadística a lo largo de los meses del año. La interactividad de esta gráfica permite también visualizar estos datos por separado.

Además, existe una sección de estadísticas de horario, accesible desde la pantalla de horario de cada sección. Esta vista dispone de controladores de paginación que permiten cambiar entre un mes y otro mediante flechas, actualizándose las estadísticas de la página en función del mes seleccionado. La primera estadística mostrada es la de demanda de turnos, que muestra los días del mes con mayor y menor demanda mediante un calendario que incluye un mapa de calor, que facilita la visualización intuitiva de los días más y menos demandados mediante colores. De forma similar, se ofrece una estadística de vacaciones aceptadas en formato de calendario con mapa de calor, que permite a los usuarios intuitivamente conocer los días más y menos aceptados del mes. Conocer esta información puede ser de gran utilidad para los empleados, especialmente a la hora de realizar sus solicitudes de vacaciones. Finalmente, se dispone de una estadística que muestra los días con más solicitudes de vacaciones pendientes del mes, y los que menos. Este está diseñado para mostrar los tres o cuatro días con valores más altos o bajos en cuanto a peticiones, empleando además colores que facilitan la comprensión de la estadística. Gracias a este gráfico, los usuarios pueden conocer los días más probables en cuanto a aceptación de vacaciones, y tener en cuenta aquellos donde obtener vacaciones puede ser más complicado.

Finalmente, el administrador de la aplicación dispone de una vista exclusiva de *dashboard*, en la que puede visualizar estadísticas clave de negocio a nivel global. Dicha vista, accesible desde la barra superior, ofrece dos métricas clave: número de

empleados totales y número de horas trabajadas a lo largo del año. Adicionalmente, muestra un diagrama de sectores que permite visualizar la distribución de empleados por sección, permitiendo conocer las secciones donde se concentran un mayor número de trabajadores de forma visual. Para conocer más en detalle esta estadística, también se dispone de una gráfica de barras con el número exacto de empleados por sección. Al lado de esta última estadística, se encuentra otra que especifica el número de horas trabajadas por sección a lo largo del año. La última gráfica de este panel se trata posiblemente de la más relevante de todas, puesto que muestra la satisfacción media por sección y mes, métrica clave e indispensable dentro de la lógica de negocio de TimeFlex. Todas las estadísticas mencionadas están diseñadas para identificar las diferentes secciones con los mismos colores en las diversas gráficas, permitiendo una mejor comprensión de la información. Además, la interactividad de las mismas permite visualizar estadísticas aisladas de ciertas secciones y ofrece la posibilidad de generar numerosas vistas diferentes dentro del mismo gráfico.

Este apartado destaca por la interactividad de sus gráficos, que permiten al usuario personalizar los mismos y mejoran su experiencia con la interfaz gráfica. De esta forma, los usuarios obtienen información valiosa de las diferentes estadísticas, que se suma al valor añadido aportado por la amplia interacción existente en cada una de ellas.

5.4 Notificaciones

Uno de los requisitos más solicitados por los empleados cuestionados en el estudio desarrollado en el Capítulo 3 fue la necesidad de un sistema de notificaciones personalizable. Esto adquiere una gran relevancia en el proyecto de TimeFlex, puesto que hay múltiples actividades que requieren la interacción entre varios empleados, y la notificación se postula como la mejor forma de acelerar y garantizar el éxito de dichas acciones.

Un ejemplo claro es el sistema de cambio de turnos, que comprende solicitudes de cambio de turno por parte de los empleados. Una vez que un empleado solicita un cambio de turno a un compañero, el trabajador receptor de dicha solicitud debe aceptar o rechazar el cambio de turno, y en caso de aprobarlo, el administrador decide

si aceptar o no dicho cambio. Esto se traduce en la necesidad de mínimo tres notificaciones por cada cambio de turno, una por cada paso del proceso. Sin embargo, los beneficios del sistema de notificaciones no se reducen exclusivamente a la funcionalidad de cambios de turno.

El panel de ayuda donde los empleados pueden pedir soporte al administrador, la ventana de contacto donde los visitantes de la aplicación web de TimeFlex pueden pedir información acerca del servicio ofrecido u otro tipo de casuísticas como errores de sistema hacen uso del sistema de notificaciones. Esto refuerza la idea de que esta funcionalidad es un servicio que facilita la operación de negocio de la aplicación y mejora la experiencia de los usuarios de manera global.

Este sistema de notificaciones es indudablemente esencial para la aplicación, pero los usuarios han demostrado su fuerte interés para que este sea también personalizable. Teniendo esto en cuenta, se han etiquetado los distintos tipos de notificación en categorías, obteniendo las siguientes: (i) ayuda, (ii) turno, (iii) sistema y, (iv) otras. Estas categorías son de gran utilidad en términos de personalización, pues los usuarios desde el panel de configuración de notificaciones pueden filtrar las notificaciones de las que quieren ser avisados por categorías. De esta forma, solo reciben notificaciones de aquellas categorías que les interesan y pueden evitar recibir un número excesivo de notificaciones, mejorando su experiencia con la aplicación.

Cuando el usuario tiene alguna notificación sin leer, aparece junto a un icono de campana (asociado a las notificaciones) un círculo rojo, hecho que no ocurre cuando ha leído todas sus notificaciones. De esta manera intuitiva, el usuario puede conocer si tiene alguna notificación pendiente. Además, si se pulsa en dicho ícono de campana, aparece un pequeño desplegable con las últimas notificaciones sin leer, y un enlace al centro de notificaciones, espacio formado por un panel donde el usuario puede ver todas sus notificaciones, tanto leídas como sin leer. Adicionalmente, como en el caso de las notificaciones, ofrece la posibilidad de utilizar filtros en caso de que se quiera visualizar únicamente las notificaciones de un cierto tipo.

Capítulo 6 - Implementación y desarrollo

Este capítulo detalla el proceso de implementación técnica del sistema, comprendiendo sus módulos y servicios. Se explican en profundidad aspectos tanto de *frontend* como *backend*, así como integraciones con servicios externos o microservicios. También se encuentra en esta sección el manual de uso de la aplicación y los sistemas de seguridad implementados.

La Sección 6.1 presenta la implementación técnica de los distintos módulos de la aplicación, la Sección 6.2 detalla el desarrollo del servicio de optimización de TimeFlex, la Sección 6.3 profundiza en las medidas de seguridad implementadas en la aplicación, y por último, la Sección 6.4 contiene aspectos relevantes del entorno de ejecución utilizado.

El código del proyecto TimeFlex es abierto y se encuentra disponible en el siguiente enlace: https://github.com/andmari11/TimeFlex/. Está formado por más de 167 módulos, 21 clases y más de 39.000 líneas de código. La aplicación combina múltiples tecnologías, incluyendo PHP (Laravel) para la lógica, HTML y JavaScript (Alpine.js) para la interfaz de usuario, y Python en módulos específicos como la optimización. A continuación, se describe en detalle el desarrollo de las principales funcionalidades y servicios del sistema.

6.1 Capa de negocio y presentación

Se trata de la capa más elemental del proyecto, cuyo desarrollo se ha llevado a cabo con Laravel [38]. Esta capa contiene la mayoría de los módulos de la aplicación y orquesta la lógica de negocio, como se explicó detalladamente en la Sección 4.1.1.

6.1.1 Calendarios

La gestión de calendarios forma parte del núcleo funcional de la aplicación desarrollada. Del módulo de calendarios depende por completo la planificación de horarios y turnos, la visualización del trabajo en la empresa y la interacción entre RRHH y empleados. Este módulo abarca desde la creación y modificación de horarios hasta las

solicitudes y confirmaciones de cambios de turno, así como la integración de estos datos en herramientas como Google Calendar [16] o Outlook [51].

Generación y modificación de horarios y turnos

La creación y modificación de horarios se administra desde el *ScheduleController* encargado de todas las funcionalidades principales de los horarios. La información de cada horario se almacena en la tabla "*schedules*" de la base de datos, guardando así todos los datos necesarios para posteriormente visualizar los horarios.

La generación de turnos se apoya en una lógica orientada a facilitar tanto la creación masiva como la edición de turnos. Para ello, se ha desarrollado la clase *ShiftType* la cual tiene su propio controlador. Esta clase permite crear tipos de turno, es decir, plantillas de turnos recurrentes (diario, semanal, mensual), esto permite generar múltiples turnos de forma automatizada y coherente con las necesidades de la compañía, así como editarlos individualmente antes o después de ejecutar el optimizador.

En el controlador de turnos se ha implementado un sistema de regeneración de turnos, permitiendo que cualquier cambio aplicado a los tipos de turno pueda actualizar todos los turnos reales asociados al horario. Este mecanismo garantiza la coherencia entre la configuración del horario y los turnos que finalmente se asignan a los empleados.

Además, se incluye una interfaz intuitiva que permite a los administradores modificar características de los horarios y de los tipos de turno con total flexibilidad.

Visualización de horarios

Para la visualización de los calendarios se ha utilizado la librería Carbon [9], esta librería ayuda a formatear las fechas al idioma español y obtener distintos datos contextuales de la fecha, como por ejemplo día de la semana o mes. Esto ha posibilitado que se organice la información de los turnos en grupos mensuales y mostrarla de forma estructurada dentro de un calendario interactivo.

La interacción en la interfaz ha sido reforzada con el uso de Alpine.js [1], permitiendo incluir elementos dinámicos sin necesidad de herramientas pesadas. Gracias a esto, se han podido incluir funcionalidades como pestañas interactivas, calendarios dinámicos y formularios de edición directa sobre turnos.

Cambios de turno

La gestión de los cambios de turno se realiza desde el controlador *ShiftExchangeController* y la clase *ShiftExchange*. TimeFlex permite dos tipos de cambios de turno:

- El personal de RRHH puede reasignar turnos libremente sin necesidad de confirmación por parte del empleado. Esta acción está protegida mediante middlewares que verifican rol y permisos de usuario antes de permitir este tipo de cambio de turno, garantizando que solo el personal de RRHH pueda llevar a cabo estas modificaciones.
- Un empleado puede solicitar un cambio de turno con otro compañero o con un turno que esté libre. Una vez hecha la solicitud, el sistema genera automáticamente notificaciones desde su respectivo módulo a todas las partes implicadas, incluido al administrador.

El flujo de la solicitud es gestionado mediante estados (pendiente, aceptado, rechazado). Si el personal de RRHH acepta la solicitud, el sistema ejecuta el cambio de turno desde el controlador y se notifica a los empleados. Este flujo asegura trazabilidad completa y mejora la comunicación interna en la empresa.

Integración con herramientas externas

Se ha añadido la posibilidad de añadir los turnos al calendario personal del usuario, como Google Calendar [16] o Outlook [51]. Esta integración se ha logrado generando enlaces automáticos en formato compatible con calendarios externos, usando parámetro como la hora de inicio y fin de turno y título del evento. Esto permite que el usuario pueda guardar su turno en su calendario personal y recibir recordatorios

fuera de la aplicación de TimeFlex. La Figura 6-1 muestra el funcionamiento de este módulo.

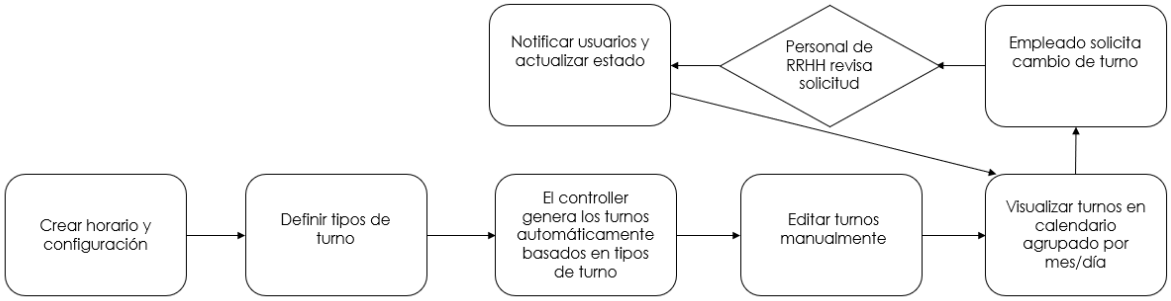

Figura 6-1. Flujo del módulo de calendarios.

6.1.2 Formularios

La funcionalidad de formularios es la encargada de recopilar las preferencias e información sobre los empleados de la empresa, utilizando sus respuestas para generar los calendarios de cada sección basándose en ellas. El menú principal de los formularios se muestra a través del siguiente *endpoint* GET /formularios, donde a través de la función *index* se muestran todos los formularios disponibles para el usuario, así como los diferentes botones que dirigen a las funcionalidades del módulo.

Creación y configuración de formularios

La lógica implementada para la gestión y configuración de los formularios se lleva a cabo en el archivo *FormsController*, el cual se encarga de centralizar todas las funcionalidades relacionadas con la creación, edición y administración.

En cuanto a la creación de formularios, como se muestra en la Figura 62, se accede a través de *GET formularios/create*, y muestra la vista de creación de formularios, donde se introducen los campos correspondientes de título, resumen, preguntas… Una vez se han rellenado los campos necesarios y se pulsa el botón para registrar el formulario en la base de datos, se llama al *endpoint POST formularios/register-form/* donde la función store se encarga de guardar en la base de datos el contenido del formulario.

Una vez creado el formulario, el administrador puede editar su configuración a través del *endpoint GET /formularios/{id}/edit*, pudiéndose modificar tanto la información del formulario como la de las propias preguntas. Una vez se realicen los cambios pertinentes y se quieran guardar los datos, *PUT /formularios/{id}* es el *endpoint* encargado de ello, donde la función *update* modifica los datos del formulario con el ID indicado.

Además, se ha desarrollado la funcionalidad de duplicar formulario única para los administradores, donde tras introducir la nueva fecha de inicio y finalización, el *endpoint POST /formularios/{id}/duplicar* se encarga de realizar una copia exacta del formulario con el ID indicado, cambiándose las fechas de inicio y finalización por las indicadas.

Visualización de la interfaz

Para la visualización de los formularios y los elementos que los componen se ha utilizado principalmente Tailwind CSS [67], que es una librería de estilos utilizada para la definición del diseño y la apariencia de diferentes elementos de la interfaz como botones, barras de control deslizante o modales de confirmación. Por otro lado, se ha utilizado la librería Flatpickr [23], la cual se utiliza en las preguntas donde se despliegan calendarios, permitiendo a los usuarios seleccionar días sueltos o intervalo de fechas de forma más gráfica e intuitiva. Además, se utiliza el método Eloquent [19] de Laravel, para mostrar los campos de cada tabla a la hora de visualizar cada diferente vista de este módulo.

Finalmente, JavaScript [68] [95] es el lenguaje de programación principal utilizado para la visualización de la interfaz, permitiendo una interacción dinámica de los usuarios con el formulario. Esto se observa en la forma de añadir preguntas y opciones al formulario a la hora de la creación, la actualización de los campos según el tipo de pregunta seleccionada a la hora de editar y la gestión de los modales.

En cuanto a la implementación, los formularios para contestar se podrán observar al acceder al *endpoint GET /formularios/{id}/show*, donde se contestarán a las preguntas del formulario con ID indicado. Una vez se responda a todas ellas, el *endpoint POST /formularios/{id}/submit*, que se encarga de almacenar las respuestas del usuario en la base de datos. Estas respuestas a su vez pueden ser modificadas mediante GET

/formularios/{id}/editar-respuestas, que muestra la vista de edición, y PUT */formularios/{id}/actualizar-respuestas*, que las actualiza en la base de datos.

Finalmente, tanto los gerentes de recursos humanos como los empleados pueden acceder a una vista donde se recopilan todas las respuestas, siendo la de Recursos Humanos *GET /formularios/respuestas*, donde se muestran todas las respuestas de todos los usuarios de la empresa, y *GET /formularios/respuestasUser* para los empleados, donde se muestran únicamente las respuestas del mismo usuario que ha iniciado sesión previamente. La Figura 6-2 muestra el funcionamiento de este módulo.

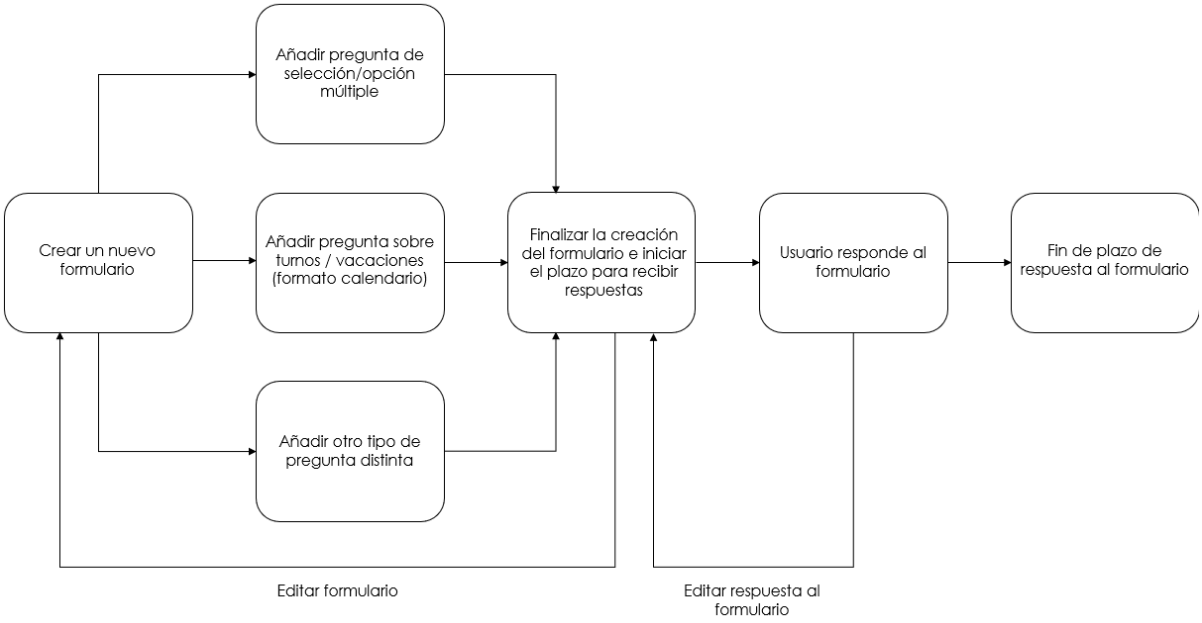

Figura 6-2. Flujo del módulo de formularios.

6.1.3 Estadísticas

En cuanto a la implementación del módulo de estadísticas, se ha empleado una API interna que ha permitido comunicar las diferentes vistas con la base de datos de TimeFlex. Mediante el uso de *endpoints*, se definen relaciones entre los datos solicitados

por la vista para generar el gráfico correspondiente y el *Controller* que recupera estos datos de la base de datos empleando el modelo Eloquent [19] de Laravel [38].

Dentro de esta funcionalidad, destaca la importancia del archivo "app.js" ya que ha sido el punto de partida de todos los gráficos de Highcharts [26] generados con JavaScript [68] [95]. En este archivo, se han cargado las dependencias y módulos necesarios para cubrir todas las necesidades visuales de este apartado (como por ejemplo *heatmap* [25] para crear mapas de calor, *timeline* [69] para crear líneas de tiempo o *highcharts-more* para crear gráficos avanzados), además de exportar Highcharts al resto de ficheros del proyecto.

En cuanto al frontend, se han utilizado varios archivos con plantillas Blade para generar las vistas de las distintas páginas de estadísticas (como por ejemplo estadisticas.blade.php o dashboard.blade.php). En cada uno de ellos, se importa app.js, y por tanto Highcharts y los módulos necesarios de forma automática gracias a Vite [73], herramienta que compila y agrupa los scripts mejorando la optimización. Además, las gráficas se generan con *scripts* que una vez se carga el DOM, realizan un *fetch* al *endpoint* definido para obtener los datos necesarios en formato JSON. El diseño de cada gráfico así como sus atributos principales tales como el tipo, título o leyenda se han personalizado también dentro de su script correspondiente.

Las peticiones *GET* realizadas desde el frontend mediante la función *fetch* a los *endpoints* se manejan desde "web.php". Las rutas de la aplicación se encuentran mapeadas, de tal forma que cuando se realiza una petición desde el frontend, existe un enlace de esa ruta con los métodos concretos de los *controllers* asociados. Además, se utiliza un mecanismo de *middleware* (ver Sección 6.3) con el fin de garantizar que solo puedan acceder usuarios autorizados a las mismas. Estos controladores disponen de números métodos que obtienen datos de diferentes tablas de la base de datos como "*satisfactions*", "*users*" o "*sections*". Estos métodos aprovechan el modelo Eloquent [19] de Laravel, que permite interactuar con cada tabla como si fuera un objeto y facilita la construcción de consultas a las mismas. Además, los datos obtenidos se guardan en arrays y otras estructuras manejables, que se incluyen en las respuestas JSON que estos métodos generan a las peticiones.

Una vez obtenida esta información por parte de la vista, se incluyen los datos como una serie dentro de la estructura Highcharts. Tras inyectar los datos, se renderiza el gráfico y se muestra por pantalla, permitiendo al usuario interaccionar con él.

A modo de resumen, la Figura 6-3. Flujo del módulo de estadísticas. muestra el flujo habitual que sucede cuando un usuario trata de ver una estadística desde la interfaz de TimeFlex:

1. El usuario interactúa en la aplicación con una página con estadísticas o pulsa en algún botón para generar un gráfico.
2. En la vista, un script Javascript solicita datos a la API interna realizando un *fetch* del *endpoint* correspondiente en función de los datos necesarios.
3. Laravel recibe la petición GET de la vista y la asocia a un método de un *Controller* gracias a las rutas definidas en el fichero "web.php", en el cual se definen estas relaciones.
4. El *Controller* correspondiente obtiene los datos solicitados de la base de datos y los transforma en un *array* u otra estructura en formato JSON.
5. El *Controller* devuelve los datos en formato JSON y la vista los recibe como respuesta al *fetch*.
6. La vista procesa los datos y los incluye en el gráfico definido de Highcharts para su renderización y posterior visualización.

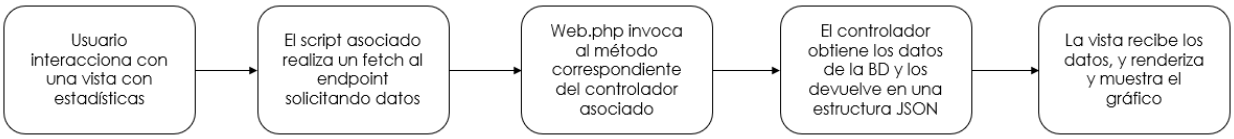

Figura 6-3. Flujo del módulo de estadísticas.

6.1.4 Notificaciones

El módulo de notificaciones basa su funcionamiento principal en la tabla de la base de datos "*notifications*". Esta almacena varios atributos relevantes para la gestión

de notificaciones (ver Sección 4.2). Entre sus atributos existen algunos creados específicamente para determinados tipos de notificación:

- Cambios de turno: "*shift_exchange_id*" identifica el cambio de turno al que se hace referencia con la notificación y "*url*" guarda el enlace directo a la vista de dicho cambio de turno con el fin de que el usuario pueda aceptar o rechazar la solicitud asociada.
- Ayuda y Contacto: "duda" contiene el mensaje escrito por el usuario.

Para garantizar que la interfaz de usuario se encuentra siempre actualizada, se ha implementado un mecanismo *backend* que cada segundo comprueba si hay nuevas notificaciones sin leer. Esta función se realiza mediante un script ubicado en "layout.php", que realiza una petición GET con la función *fetch* al *endpoint* definido en "web.php" asociado a las preferencias. De esta forma, se llama al método "*getUnreadNotifications*" de NotificationController, que devuelve las notificaciones sin leer respetando las preferencias del usuario.

La actualización constante en el *backend* sucede también en el *frontend*, gracias al icono de campana ubicado en la esquina superior derecha de la barra superior de la aplicación. Cada vez que se detecta que hay notificaciones sin leer con el mecanismo mencionado, dicho icono cambia su estado mostrando un círculo rojo como se puede apreciar en la Figura 6-4:

Figura 6-4. Icono indicando nuevas notificaciones.

De esta forma, el usuario puede ser consciente de esta situación y consultar sus nuevas notificaciones bien desde el desplegable asociado a la campana o accediendo al centro de notificaciones. Los filtros habilitados para el usuario en este último se manejan desde el método *index* de NotificationController:

Por lo tanto, el flujo regular del módulo de notificaciones, tal y como se muestra en la Figura 6-5, es el siguiente:

1. El usuario realiza un evento que genera una notificación.
2. Cada minuto, el script definido en layout.php realiza un *fetch* al *endpoint* definido para obtener las notificaciones nuevas sin leer.
3. Web.php invoca al método *getUnreadNotifications* de *NotificationsController* que se encarga de obtener los datos solicitados de aquellas notificaciones cuyo tipo sea compatible con las preferencias del usuario.
4. Se obtienen los datos y se devuelven en una estructura JSON.
5. La vista recibe los datos y actualiza el icono de campana en caso de ser necesario.

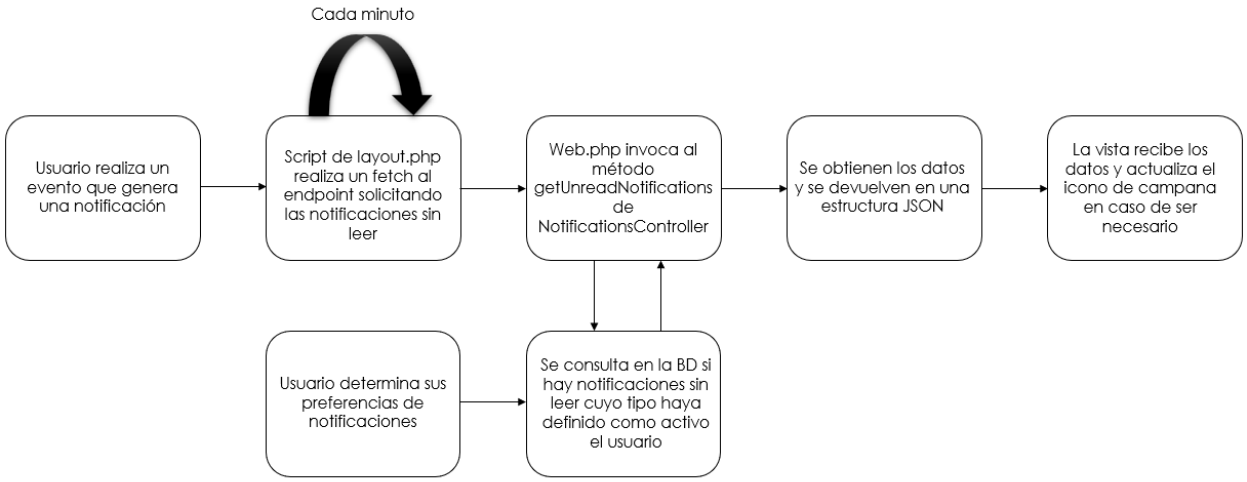

Figura 6-5. Flujo del módulo de notificaciones.

6.2 Servicio de optimización

El servicio de optimización ha sido diseñado como un componente independiente dentro de la arquitectura del sistema, permitiendo procesar solicitudes de generación de horarios sin afectar a la aplicación principal. Para interactuar con este microservicio, se ha definido un punto de entrada principal a través del endpoint /api/schedule. Este recibe una petición POST con los parámetros

necesarios para generar un horario, los turnos a repartir y los trabajadores junto a sus preferencias. Utilizando el sistema de validación automática de FastAPI, los datos se reciben como un objeto tipado (ScheduleData), que permite el proceso y validación de los campos.

La función invocada por el *endpoint* devuelve de inmediato un mensaje de confirmación al *backend* principal, no sin antes lanzar de forma asíncrona la función encargada ejecutar la optimización. Este comportamiento se visualiza en la Figura 6-6. Ejemplo de flujo de petición a microservicio., donde se muestra el flujo de petición desde la capa de negocio al microservicio: primero se realiza un proceso de autenticación que se desarrolla en detalle en la Sección 6.3. Una vez recibida esta solicitud, el microservicio confirma el inicio del proceso y ejecuta la optimización en segundo plano, sin bloquear el hilo principal.

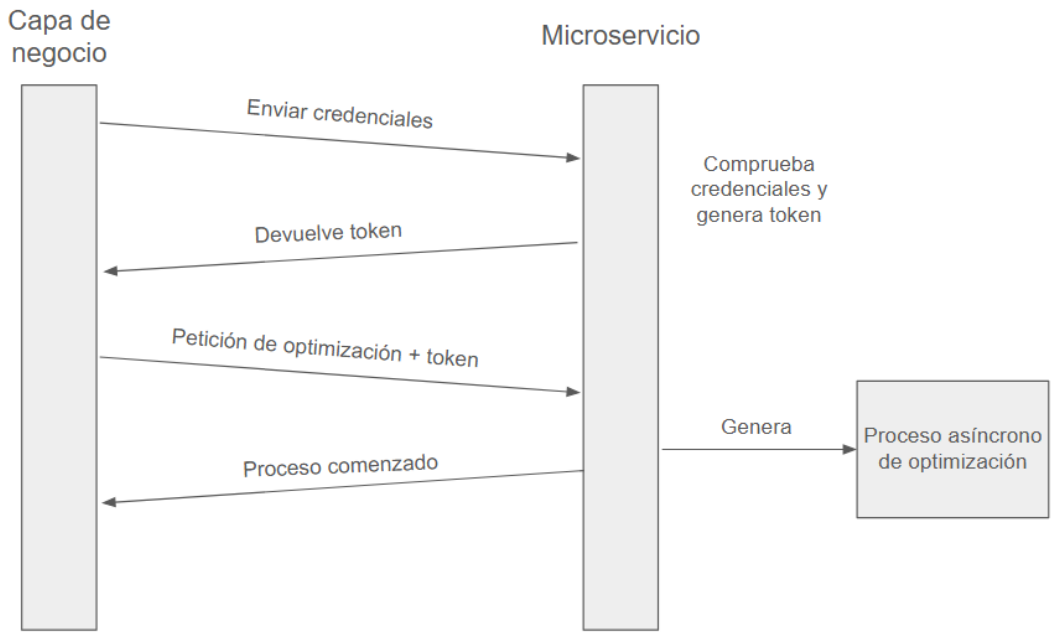

Figura 6-6. Ejemplo de flujo de petición a microservicio.

La segunda parte del flujo puede observarse en la Figura 6-7. Ejemplo de flujo de respuesta desde microservicio., que representa la respuesta del microservicio una vez finalizado el proceso asíncrono. En este punto, el microservicio recoge el resultado de la optimización y lo devuelve a la capa de negocio, repitiendo el mismo proceso de autenticación.

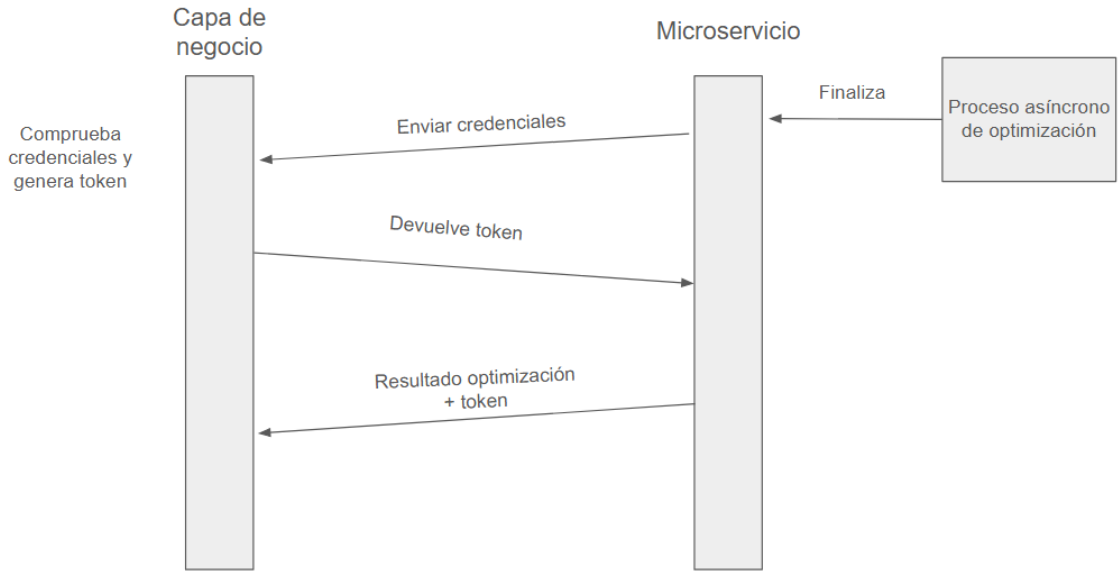

Figura 6-7. Ejemplo de flujo de respuesta desde microservicio.

Este diseño en dos fases permite desacoplar la ejecución pesada, proporcionando tanto eficiencia como una mejor escalabilidad.

Los datos recibidos se transforman en dos clases internas que representan las entidades necesarias para el proceso:

La clase Shift encapsula los datos de un turno concreto, incluyendo su identificador, duración, número de empleados necesarios, tipo de turno y usuarios asignados. El constructor asegura que las fechas son válidas y tienen formato datetime, permitiendo un tratamiento uniforme durante el modelado en Z3.

La clase WorkerPreference representa las preferencias individuales de cada trabajador. Incluye información como las vacaciones solicitadas con un peso relativo

calculado en función a la antigüedad de la petición en comparación con el resto de empleados, los tipos de turnos preferidos junto a un peso asignado por el administrador de la empresa y la satisfacción histórica para asegurar la equidad de los horarios generados en base a horarios anteriores.

Toda esta información estructurada se utiliza posteriormente por el módulo de optimización, el cual transforma estas entidades en variables y restricciones que Z3 puede procesar. El servicio de optimización ha sido implementado en Python, utilizando la API de Z3. El objetivo de Z3 es encontrar una solución (modelo) que cumpla las restricciones, es decir, una asignación para todas las variables, que cumplan las restricciones impuestas.

Denotamos las siguientes variables:

W = {w1 ,w2 ,...,wn }: conjunto de trabajadores.

S = {s1 ,s2 ,...,sm }: conjunto de turnos.

$X_{i,j} \in \{0,1\}$: variable booleana que indica si el trabajador i realiza el turno j.

n = el número de trabajadores.

m = el número de turnos.

Se pueden distinguir dos tipos de restricciones en la codificación de Z3 (la codificación completa se encuentra disponible en el repositorio de GitHub del proyecto).

Restricciones duras

Las restricciones duras, son restricciones que se deben cumplir obligatoriamente en el modelo, como por ejemplo que un trabajador no esté asignado a dos turnos simultáneamente:

1. Límite máximo de turnos por trabajador

Cada sección tiene un máximo y mínimo de turnos configurables, que limitan cuántos turnos pueden ser asignados a un trabajador con el objetivo de regular la sobrecarga de trabajo.

$$\sum_{j=1}^{m} x_{ij} \leq T_{máx} \wedge x_{ij} \geq T_{mín} \quad \forall i \in [1,n]$$

Donde:

$T_{máx}$: máximo de turnos asignables a un trabajador.

$T_{mín}$: mínimo de turnos asignables a un trabajador.

2. Límite máximo de horas por trabajador

Además del máximo y mínimo de turnos, también son configurables los límites numéricos de horas totales.

$$\sum_{j=1}^{m} x_{ij} \cdot h_j \leq H_{máx} \wedge x_{ij} \cdot h_j \geq H_{mín} \quad \forall i \in [1,n]$$

Donde:

$H_{máx}$: máximo de horas que se puede asignar a un trabajador.

$H_{mín}$: máximo de horas que se puede asignar a un trabajador.

h_j : duración del turno j.

3. No superar el máximo de turnos diarios

Cada trabajador puede como máximo trabajar un máximo de turnos por días. Por defecto este valor es 1.

$$\sum_{j=1}^{t} x_{ij} \leq D_{máx} \quad \forall i \in [1,n]$$

Donde:

$D_{máx}$: máximo de turnos diarios que se puede asignar a un trabajador.

t : número de turnos diarios.

4. Turnos con suficientes trabajadores asignados

Cada turno debe tener el número mínimo de empleados necesario configurado.

$$\sum_{i=1}^{n} x_{ij} = R_j \quad \forall j \in [1, n]$$

Donde:

$H_{máx}$: máximo de horas que se puede asignar a un trabajador.

h_j : duración del turno j.

5. Asignación fija de turnos preasignados

Los turnos previamente asignados manualmente se asignan obligatoriamente. Es decir, x_{ij} pasa a tener valor True si el $turno_j$ tiene asignado manualmente al $trabajador_i$.

$$\forall (w_i, s_j) \in M_i, \; x_{ij} = 1$$

Donde:

M_i: conjunto de turnos asignados previamente a la optimización al trabajador.

Restricciones suaves

Las restricciones suaves por otro lado, pueden ser incumplidas pero tienen una penalización y el resolutor intentará minimizarlas. La penalización asociada a su incumplimiento es configurable por el administrador, quien puede asignar un nivel de importancia dentro de un rango de 1 a 10. Este valor actúa como un factor de ponderación dentro de la función objetivo, compitiendo con el resto de preferencias y restricciones suaves.

1. **No trabajar en días de vacaciones**

Penaliza la satisfacción, pero no imposibilita que a un trabajador se le asigne un turno que coincida con un día de vacaciones solicitado. La penalización depende de la prioridad de la petición, que es proporcional a la antigüedad de esta respecto a las demás, siguiendo una normalización comparativa. Así, si un trabajador solicitó sus vacaciones mucho antes que otros, su petición tendrá mayor peso.

2. **Preferencias de tipo de turno**

Se favorece que los empleados sean asignados a los tipos de turno que prefieren. Posteriormente, se contabilizan los casos positivos y negativos y se calcula cuánto afectan a la satisfacción del trabajador dependiendo de la configuración.

Evaluación y equidad en las asignaciones

Una vez que el solucionador Z3 encuentra una solución válida que respeta las restricciones impuestas, el sistema evalúa la satisfacción individual de cada trabajador respecto al calendario generado teniendo en cuenta los turnos asignados y las vacaciones rechazadas. Este cálculo tiene un doble objetivo: (i) por un lado, ofrecer transparencia sobre la calidad de la solución, y por otro, (ii) utilizar esta información para mejorar la equidad entre los empleados.

Teniendo en cuenta las preferencias de vacaciones y turnos asignados, el sistema calcula de la siguiente manera:

Formalmente, para cada trabajador_i, se define

$$Satisfacción_i = \alpha \cdot P_i + \beta \cdot V_i$$

Donde:

P_i: número de turnos preferidos asignados al trabajador$_i$.

V_i: número de días de vacaciones en los que el trabajador$_i$ ha sido asignado.

α, β: pesos que representan la importancia relativa de cada aspecto que han sido configuradas previamente por el administrador.

Posteriormente, para facilitar la comparación entre empleados y representar las puntuaciones en una escala de 0 a 100, se aplica una normalización lineal:

$$Satisfacción\ \ normalizada_i = 100 \cdot \frac{Satisfacción_i - min}{max - min}$$

Donde:

max, min: son los valores máximos y mínimos de la satisfacción obtenida de los trabajadores.

Para evitar que algunos trabajadores se vean constantemente perjudicados, se tiene en cuenta el historial de satisfacción de calendarios anteriores y se calcula la media histórica de satisfacción para cada trabajador$_i$

$$Satisfacción\ \ histórica_i = \frac{\sum_{k=1}^{n} S_i^{(k)} + S_i^{(actual)}}{n+1}$$

Donde:

$S_i^{(k)}$: satisfacción normalizada del trabajador$_i$ del calendario$_k$.

$S_i^{(actual)}$: satisfacción normalizada del trabajador$_i$ en el calendario que está siendo calculado actualmente.

n: número de calendarios anteriores registrados.

Finalmente, con el fin de maximizar la equidad en la asignación de turnos y la satisfacción de los trabajadores, se aplican las siguientes fórmulas:

$$\text{Maximizar la satisfacción media} : max\ (\frac{1}{n}\sum_{i=1}^{N}\overline{S_i})$$

$$\text{Minimizar la desviación a la media global} : min\ (\sum_{i=1}^{N}\left|\overline{S_i} - \overline{S}\right|)$$

Donde:

S_i: satisfacción media histórica del trabajador$_i$.

S: satisfacción media histórica global.

n: número de trabajadores.

Una vez alcanzada la solución optimizada que satisfaga todas las restricciones se genera un horario individual por cada trabajador junto a su satisfacción normalizada y un log detallado del proceso y sus métricas. Los horarios y satisfacciones se envían de vuelta en formato JSON para su almacenamiento y visualización en Laravel.

6.3 Seguridad

La seguridad en TimeFlex no es un tema menor, especialmente cuando se gestionan datos personales en una empresa. Desde el primer momento se han seguido buenas prácticas de seguridad, aprovechando al máximo las herramientas que proporciona el *framework* de Laravel. A continuación, se describen distintas prácticas de seguridad que se han seguido a lo largo del proyecto:

CSRF. Uno de los primeros muros de defensa es la protección contra CSRF (Cross-Site Request Forgery) [85] [89], una técnica que impide que alguien intente actuar en nombre de otro usuario sin su consentimiento. Por ejemplo, un atacante podría generar un enlace que envíe una solicitud maliciosa a la aplicación si un usuario tuviese sesión iniciada (modificar la configuración o eliminar empleados).

Para evitar esto, todos los formularios generados por la aplicación incluyen un token CSRF que el servidor valida en cada petición *POST, PUT, PATCH* o *DELETE*.

Verificando mediante el middleware *VerifyCsrfToken* [34] que el token enviado junto al formulario coincida con el que se generó para esa sesión. Si no coincide, la petición se considera inválida y se bloquea automáticamente.

XSS. Con la intención de evitar inyecciones de código malicioso en las distintas vistas de la aplicación, la capa de presentación protege automáticamente contra ataques de XSS(Cross-Site Scripting) [91] [92], una de las amenazas más comunes en aplicaciones web. Este tipo de ataques se produce cuando un usuario, de forma malintencionada, consigue inyectar código JavaScript u otros scripts en contenidos que luego son renderizados por otros usuarios en su navegador. Por ejemplo, un atacante podría introducir un fragmento de JavaScript como nombre de usuario, de forma que, al visualizar su perfil, dicho script se ejecute en el navegador de la víctima, permitiendo al atacante llevar a cabo acciones en nombre de la víctima

Gracias a la política de "escape por defecto" implementada en la aplicación Laravel, se transforman todas las variables impresas en una versión segura que no puede ser interpretada como código ejecutable por el navegador. Por ejemplo, una cadena como: "*<script> alert('XSS') </script>*", se transformaría automáticamente en texto plano seguro: "*<script> alert('XSS') </script>*", impidiendo su ejecución, aunque seguiría mostrando la cadena previa a la modificación.

Gestión de accesos mediante middleware. La aplicación hace uso del sistema de middlewares proporcionado por Laravel [42], para gestionar el acceso de rutas y funcionalidades según el rol del usuario. Cada ruta o grupo de rutas se protegen mediante middleware personalizados, que se encargan no solo de verificar si el usuario está autenticado, sino también si cuenta con los permisos necesarios antes de permitir el acceso. En caso contrario, se le redirige de vuelta a una vista permitida. Además del control de permisos, existe un middleware que permite registrar los accesos, lo que resulta útil para la trazabilidad, pudiendo identificar cuándo y qué usuarios han accedido a determinadas secciones del sistema.

Almacenamiento seguro de contraseñas. En TimeFlex, las contraseñas almacenadas están protegidas y cifradas de forma segura. Para ello, en vez de guardar las contraseñas en texto plano, se utiliza el algoritmo Argon2 [77], uno de los algoritmos más robustos y recomendados actualmente. Argon2 añade una sal única (*salt*) para

cada contraseña, es decir un valor aleatorio que se combina la contraseña e impide que dos contraseñas iguales generen el mismo resultado. Además, procesa múltiples veces el hash generado de forma paralela para dificultar ataques de fuerza bruta. Gracias a esto, si un atacante consigue acceder a la base de datos, no podría obtener las contraseñas reales, ya que sería computacionalmente costoso y extremadamente lento. Sin embargo, también es posible utilizar Bcrypt [79], un algoritmo seguro, pero que resulta menos resistente que Argon2.

Configuración de CORS. CORS (Cross-Origin Resource Sharing) [13] es un mecanismo de seguridad que controla las peticiones de origen cruzado en aplicaciones web como TimeFlex. Por defecto, los navegadores bloquean las solicitudes provenientes de un origen distinto al servidor de destino, persiguiendo el objetivo de prevenir ataques que puedan comprometer la seguridad de la aplicación. En este contexto, CORS permite configurar aquellos dominios autorizados para interactuar con los servicios o recursos ofrecidos por la aplicación, como por ejemplo la API. Esto refuerza la seguridad del sistema al impedir conexiones potencialmente maliciosas desde sitios externos eludiendo posibles ataques. Todo este sistema de seguridad es facilitado por Laravel, gracias al soporte nativo que ofrece a la hora de definir estas políticas mediante el middleware HandleCors [33]. Esto permite que sin necesidad de implementar lógica adicional, la aplicación cuente con validaciones CORS.

Autenticación con APIs externas. Para ofrecer una autenticación segura, se ha contemplado el uso de Laravel Sanctum [37], un sistema ligero de autenticación API basado en tokens personales. Esto permite ofrecer acceso seguro al microservicio sin necesidad de mantener sesiones tradicionales.

En la primera fase de interacción (Figura 6-6. Ejemplo de flujo de petición a microservicio.), la capa de negocio envía las credenciales, en este caso "*user_id*" y "*user_secret*", que son validadas por el microservicio y genera un token de autenticación. Una vez obtenido este token, realiza una nueva petición con el objetivo de ejecutar una optimización. Por otro lado, el microservicio una vez validado el token responde inmediatamente confirmando el comienzo del proceso y lanza en segundo plano la lógica de optimización.

La segunda fase (Figura 6-7), representa el flujo de respuesta. Una vez finalizado el proceso asíncrono, se repite el proceso previamente utilizado para entregar los datos al *backend* principal.

6.4 Gestión y verificación

El correcto funcionamiento de una aplicación tiene una gran dependencia de la gestión y verificación del entorno. Esto incluye la capacidad de registrar eventos relevantes, diagnosticar problemas y validar el comportamiento durante todo el proceso de desarrollo. En esta sección se describen las herramientas y metodologías implementadas para garantizar la trazabilidad y validación del sistema desarrollado.

6.4.1 Sistema de logs y trazabilidad

Para facilitar el mantenimiento y la monitorización del microservicio, se ha implementado un sistema de registro de eventos utilizando el módulo estándar *logging* de Python. Esto permite registrar información sobre las solicitudes recibidas, errores o eventos claves en distintos niveles de registro dependiendo de la criticidad de los mensajes (*DEBUG, INFO, WARNING, ERROR, CRITICAL*). Además en un futuro, el uso de logs de forma estructurada permite su integración en sistemas como Stack [63], servicio para almacenamiento, búsqueda y visualización de logs, permitiendo su observación desde la nube.

En cuanto a la capa de negocio implementada en Laravel [38], se ha configurado también un sistema de *logging* que utiliza el componente integrado de Laravel. Este sistema registra eventos relevantes del *backend*, como accesos a recursos, errores en las operaciones de base de datos, o alertas del sistema de forma ordenada, facilitando el diagnóstico de errores.

6.4.2 Datos de pruebas

Para asegurar la fiabilidad del sistema a lo largo del desarrollo de cada funcionalidad, se optó por la generación automatizada de datos de prueba tanto en Laravel como en FastAPI [21].

En Laravel, se utilizaron *seeders*, que son scripts encargados de poblar la base de datos con datos iniciales o ficticios, utilizando la librería Faker [35], para generar datos con información realista y variada.

Paralelamente, en FastAPI, se realizaron pruebas utilizando tanto los datos generados automáticamente en Laravel como escenarios variados, incluyendo casos realistas y algunos diseñados manualmente para cubrir situaciones menos comunes, con el fin de facilitar una validación exhaustiva del microservicio.

En conjunto, esta estrategia de pruebas entre los distintos componentes ha sido esencial para garantizar la consistencia, fiabilidad e integridad del sistema, permitiendo detectar y corregir errores, tanto en funcionalidades nuevas como en otras ya desarrolladas. De esta forma se ha asegurado un desarrollo ágil y seguro.

Capítulo 7 - Instalación y manual de uso

Este capítulo ofrece los pasos a seguir para instalar el proyecto TimeFlex y contiene los manuales de uso de la aplicación. La Sección 7.1 describe el proceso de instalación y la Sección 7.2 detalla los manuales de uso.

7.1 Instalación

A continuación, se describen los pasos necesarios referentes a la instalación y configuración de elementos para el correcto funcionamiento de la aplicación web TimeFlex, tanto en entorno de desarrollo como de pruebas locales. Se requiere tener instalados previamente PHP, npm [46] [47], Python [56], así como un gestor de base de datos (en nuestro caso SQLite [62] [83]).

1. Se debe clonar el proyecto de Laravel desde GitHub [18] utilizando el siguiente comando:

   ```
   git clone https://github.com/andmari11/TimeFlex.git
   ```

2. Una vez clonado, entrar en el directorio del proyecto:

   ```
   cd TimeFlex/
   ```

3. Dentro del directorio del proyecto, instalar todas las dependencias necesarias, definidas en el archivo *composer.json*, ejecutando el siguiente comando:

   ```
   composer install
   ```

4. Laravel utiliza un archivo .env que define las variables de entorno. En caso de no existir, se puede generar a partir del archivo de ejemplo:

   ```
   cp .env.example .env
   ```

Posteriormente, es necesario editar el archivo .env con el fin de configurar el entorno: definir la conexión a la base de datos, el puerto y el dominio.

5. Generar la clave de aplicación en caso de no estar ya establecida en el archivo .env con el comando:

```
php artisan key:generate
```

6. Para crear las tablas necesarias en la base de datos se deben ejecutar las migraciones con el comando:

```
php artisan migrate
```

7. Si se desea poblar la base de datos con datos ficticios (usuarios, empresas, turnos, secciones, etc.), se puede ejecutar:

```
php artisan db:seed
```

8. Se ha de ejecutar la API de Python necesaria para las optimizaciones, e iniciar el servidor que implementa el micro-framework de FastApi [21] mediante los siguientes comandos:

```
pip install uvicorn
pip install fastapi
pip install httpx
cd fastapi; python -m uvicorn main:app --host 127.0.0.1 --port 8001
```

9. Para que los estilos y scripts funcionen correctamente, es necesario compilar los archivos con Vite [73]:

```
npm run dev
```

10. Verificar la configuración correcta del servidor web para lanzar el proyecto de Laravel. En caso de usar el servidor web integrado de Laravel [38] para pruebas, iniciarlo con el siguiente comando:

```
php artisan serve
```

Esto va a abrir la aplicación de TimeFlex en http://localhost:8000.

7.2 Manual de uso

TimeFlex se ha desarrollado con el objetivo de simplificar la administración del personal, la programación de los horarios laborales y las peticiones de días libres o vacaciones en una compañía. Gracias a su diseño amigable y unificado, cada usuario puede encontrar los datos y funciones clave para mejorar la comunicación y la organización interna de los grupos de trabajo.

La aplicación contempla dos roles principales, gerente de RRHH (administradores) y empleados de la empresa. Para facilitar la usabilidad y el entendimiento de la plataforma desarrollada, se proporcionan los siguientes manuales de uso para cada rol.

7.2.1 Manual de uso para Gerente de RRHH

Desde el perfil de gerente de RRHH, la aplicación proporciona un conjunto completo de herramientas orientadas a la gestión eficiente del personal, la creación de formularios, la generación de horarios optimizados y la visualización de estadísticas relevantes. A continuación, se detallan las principales funcionalidades disponibles para este perfil.

7.2.1.1 Inicio de sesión

Para acceder a la aplicación, el gerente de RRHH debe dirigirse a la URL de TimeFlex (descrita anteriormente) y clicar sobre el botón de Iniciar Sesión, donde se deben introducir las credenciales de email y contraseña (ver Figura 7-1).

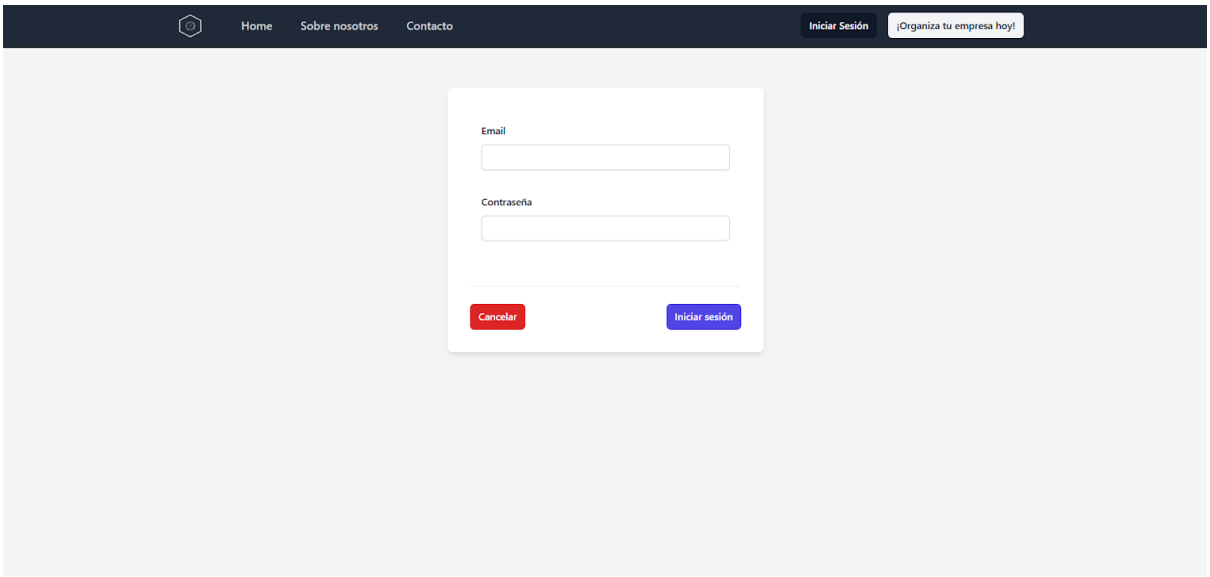

Figura 7-1. Pantalla de inicio de sesión.

7.2.1.2 Panel de administración

Una vez que el usuario con rol de gerente de RRHH inicia sesión, es redirigido a su área de usuario (ver Figura 7-2). En este caso, el gerente de RRHH tiene disponible todas las funcionalidades de administración.

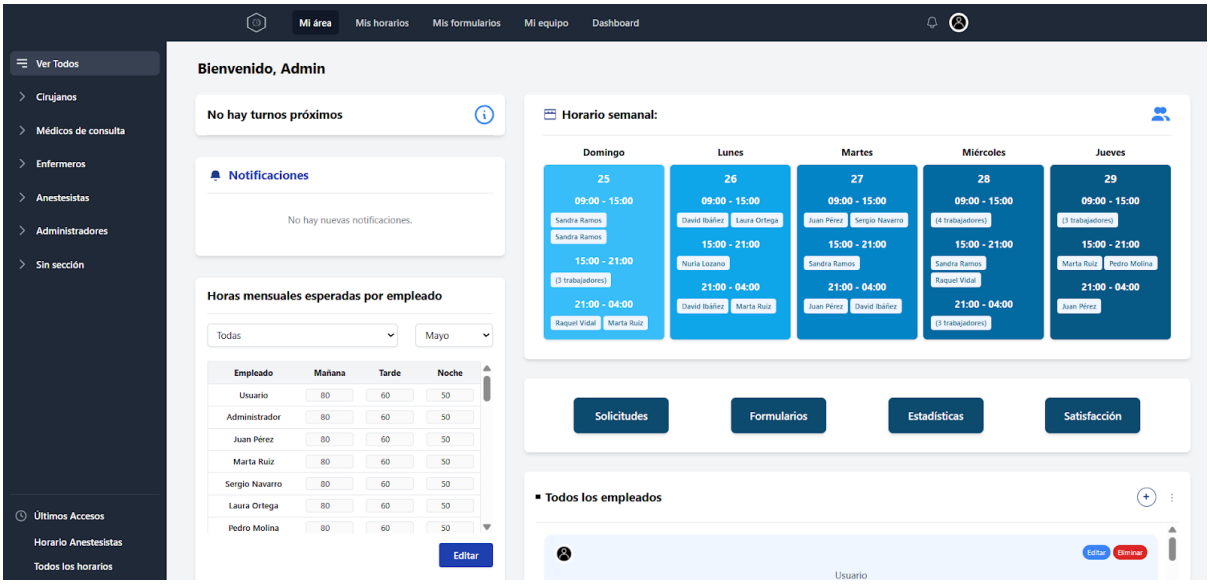

Figura 7-2. Panel de administración.

- En el encabezado de la página se pueden encontrar accesos directos a Mi área (ver Figura 7-2), Mis horarios (todos los horarios de la empresa y opciones para la administración de estos), Mis formularios (todos los formularios de la empresa y administración de estos), Mi equipo (todos los trabajadores de la empresa y administración de los empleados) y *Dashboard* (visualización de estadísticas). Además, en la parte derecha se encuentra el icono (una campana) para el acceso a notificaciones y foto de perfil, permitiendo que el propio usuario edite su perfil, acceda a los ajustes o cierre sesión.
- La barra lateral izquierda se divide en dos partes, en la parte superior, tenemos la opción de ver todas las secciones de la compañía. Por defecto se seleccionan todas las secciones, pero existe la posibilidad de clicar sobre una sección en concreto, con el objetivo de que el panel de administración muestre el horario semanal y los empleados de esa sección en específico. En la parte inferior se encuentra un historial de los últimos accesos.
- En el cuerpo de la página de administración aparecen distintos apartados divididos claramente. A primera vista se encuentra el próximo turno a trabajar. A la derecha de este se encuentra el horario semanal, con posibilidad de filtrar por la sección seleccionada en la barra lateral izquierda. Bajo el horario

semanal están disponibles accesos rápidos a solicitudes, formularios, estadísticas y satisfacción de los empleados, mientras que a la izquierda se muestran las últimas notificaciones. En la parte inferior izquierda se encuentra el apartado de horas esperadas de los empleados, con la posibilidad de filtrar por sección y mes del año, así como también la funcionalidad de edición de horas esperadas. Por último, en la parte inferior derecha, se observa el apartado de empleados con opción de filtrar por sección desde la barra lateral izquierda. En este último apartado como gerente de RRHH se puede hacer uso de funcionalidades para gestionar (añadir modificar o eliminar) tanto empleados como secciones.

7.2.1.3 Gestión de usuarios

Desde el panel de administración se tiene acceso a todas las funcionalidades de gestión de los empleados de la compañía.

Para la creación de nuevos empleados y su registro en el sistema hay que pulsar en el botón +, localizado en la parte superior derecha del apartado de empleados, en el panel de administración. Se redirige a un formulario de registro de empleado (ver Figura 7-3) que hay que rellenar con los siguientes campos obligatorios: Nombre, email, sección (desplegable con todas las secciones de la empresa), peso del usuario, contraseña y confirmación de contraseña. El peso del usuario representa como de favorecido se verá el empleado por el algoritmo de optimización para el reparto de turnos. Como último paso, se pulsa en el botón de registrar empleado.

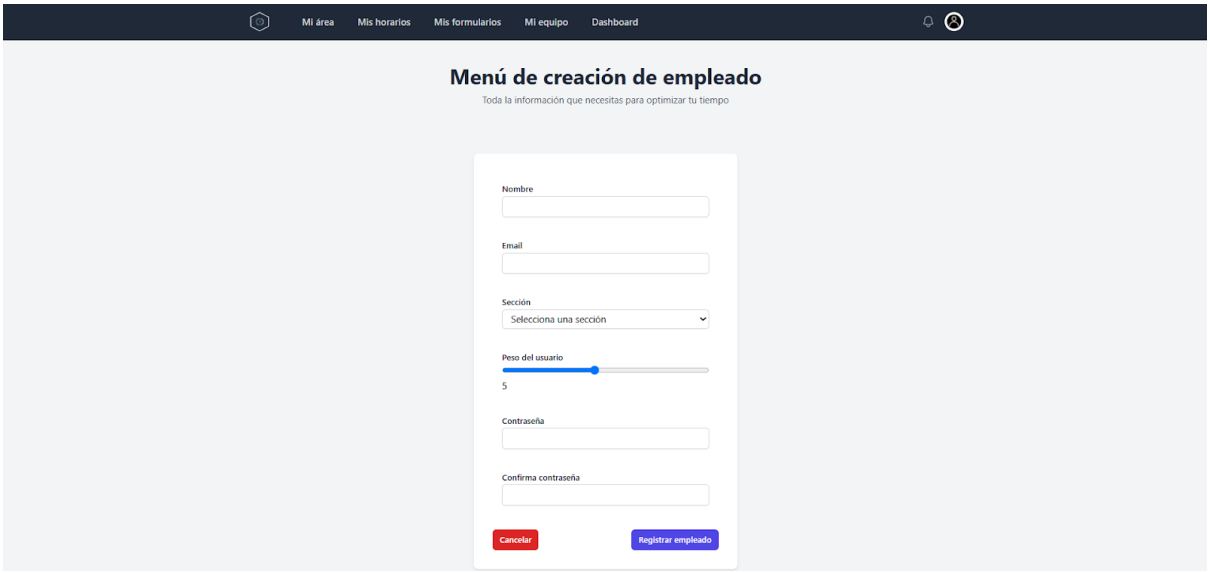

Figura 7-3. Registro de empleado

Para editar la información de los empleados, hay que ir al listado de usuarios en el apartado de empleados y clicar sobre el botón "Editar" del empleado en específico. Esto redirige a un formulario de edición (ver Figura 7-4) con la posibilidad de modificar cualquier campo del empleado. Aparecerán los valores de los campos antiguos por defecto con la posibilidad de cambiarlos. Una vez editado el usuario se pulsa el botón de "Actualizar información".

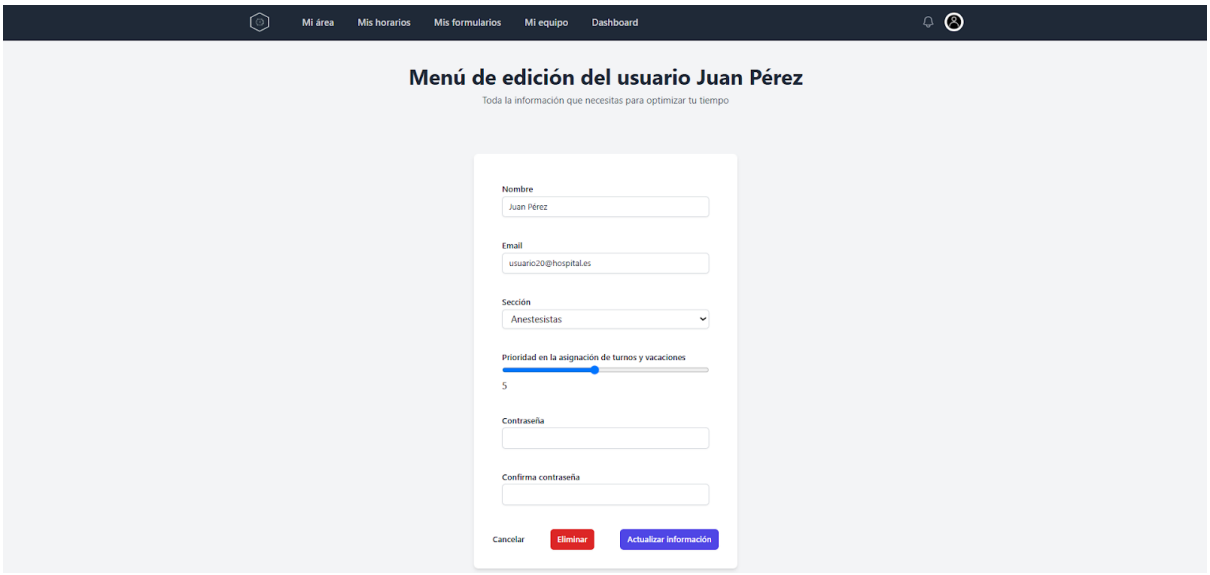

Figura 7-4. Edición de empleado.

Si el objetivo del gerente de RRHH es suprimir un usuario, únicamente hay que ir al listado de usuarios y pulsar sobre el botón "Eliminar" del usuario en cuestión, disponible junto al botón "Editar". Además, existe la posibilidad de eliminar un usuario desde el formulario de edición de usuario, pulsando el botón "Eliminar" en la parte inferior del formulario. Antes de borrar el usuario aparece un pop-up de confirmación para evitar errores.

Por otro lado, si se pulsa sobre un usuario en el listado de empleados, nos aparece en la parte derecha de la pantalla un resumen sobre el empleado (ver Figura 7-5). Este resumen contiene la opción de editar y eliminar al usuario, posibilidad de visualizar su horario y un resumen de sus estadísticas.

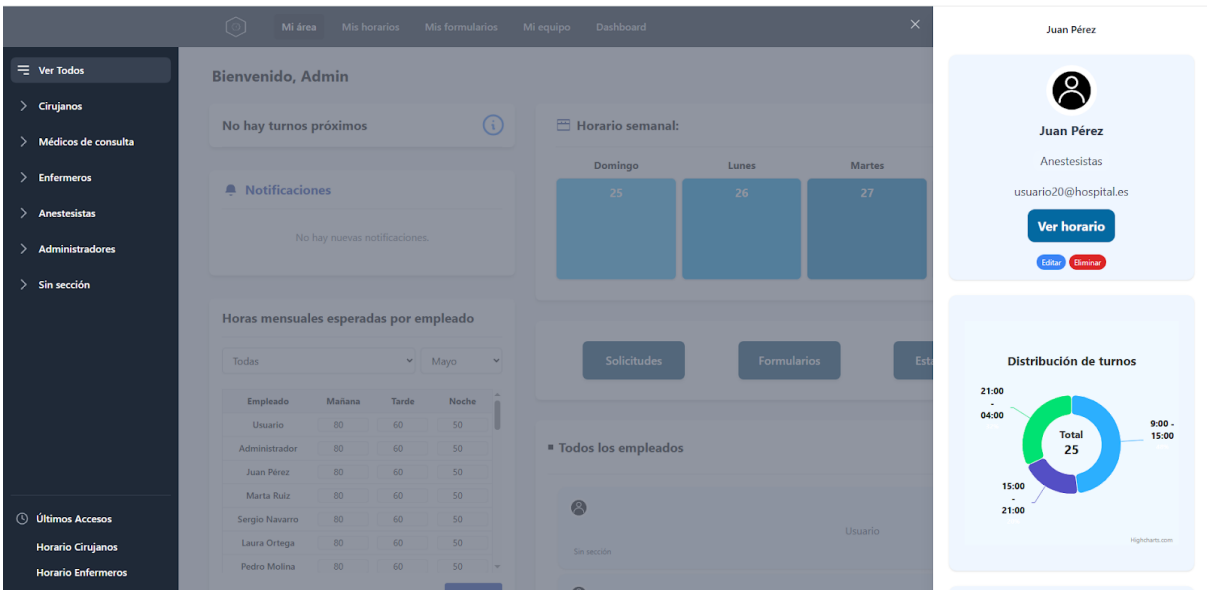

Figura 7-5. Información y estadísticas de empleado.

Finalmente, se pueden gestionar usuarios desde el acceso a *Mi equipo* en el encabezado de la vista. Al ser gerente de RRHH se pueden visualizar todos los usuarios del sistema (ver Figura 7-6). En este caso, se cuenta con una barra de búsqueda para filtrar a un empleado en concreto. Debajo de la barra de búsqueda se encuentra un botón para descargar un CSV con toda la información de los empleados. Desde esta vista también se cuenta con todas las funcionalidades de administración de usuarios descritas anteriormente.

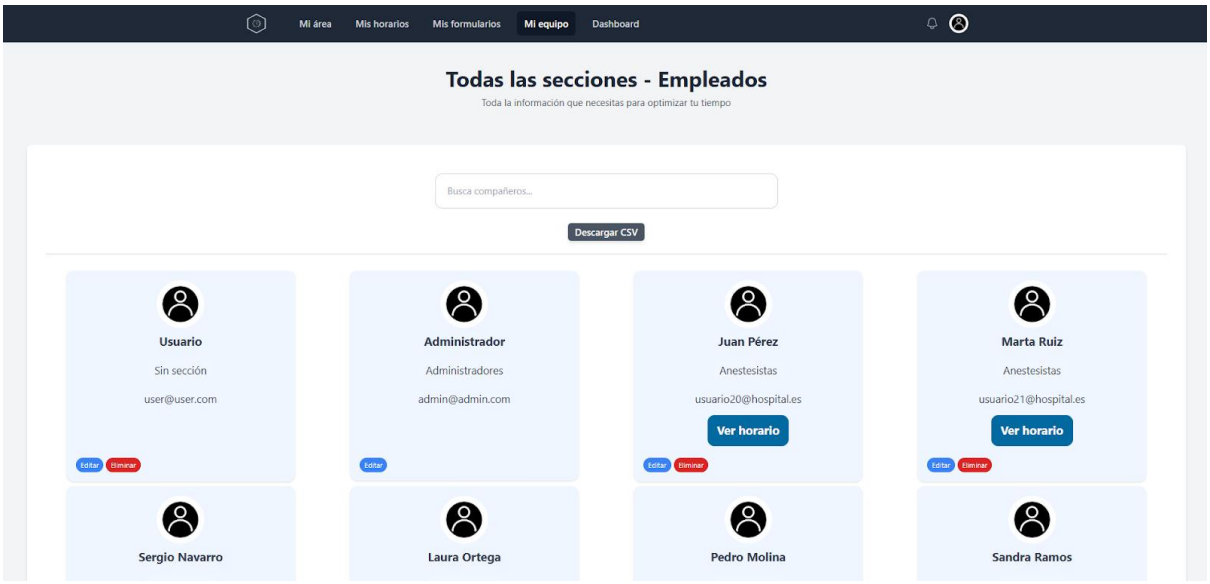

Figura 7-6. Vista de empleados.

7.2.1.4 Gestión de secciones

Todo lo relacionado con gestión de secciones se realiza desde el área de administración. En el apartado de empleados, justamente al lado del botón de creación de usuarios, se encuentran tres puntos (ver Figura 7-7) que incluyen las tres principales funcionalidades de gestión de secciones: añadir, editar y eliminar.

Figura 7-7. Opciones para secciones.

Para añadir una nueva sección, se pulsa sobre el botón vertical de menú (tres puntos) y se clica sobre "Añadir sección". Tras ello, se redirige a un formulario (ver Figura 7-8) para la creación de una nueva sección, donde hay que rellenar los campos obligatorios: nombre de la sección, horas mínimas a trabajar, horas máximas a trabajar, turnos mínimos y turnos máximos. Posteriormente, para añadir la nueva sección se pulsa sobre el botón de "Registrar sección".

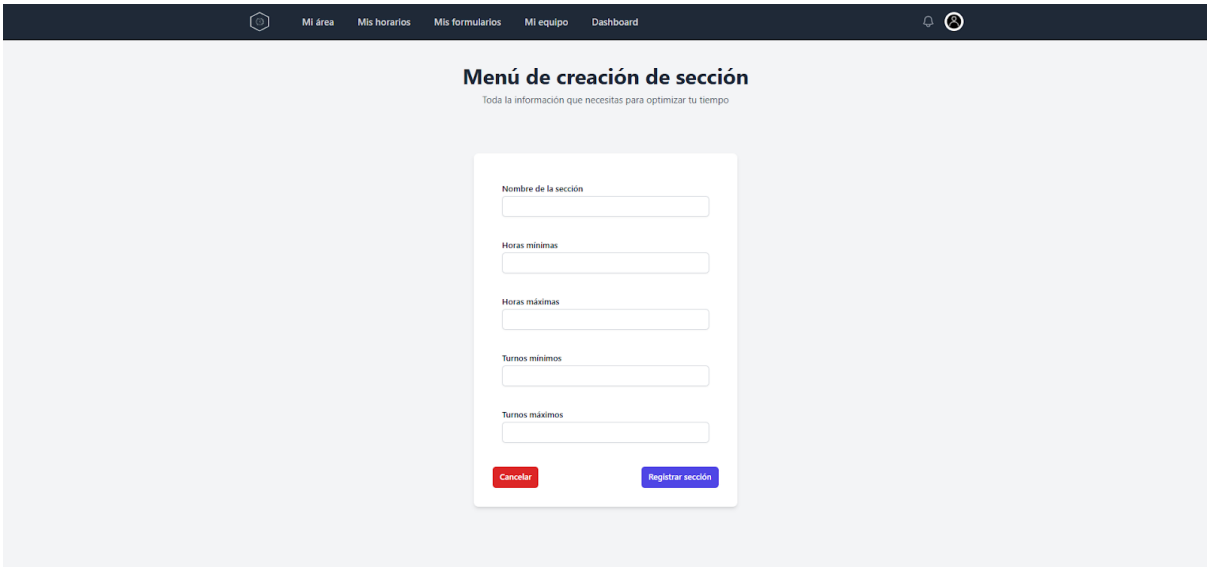

Figura 7-8. Registro de sección.

Si se quiere editar una sección, simplemente en sobre el botón vertical de menú, pulsamos sobre "*Editar sección*". En este momento aparece un pop-up para seleccionar qué sección de la compañía se quiere editar. Al seleccionar la sección se abre el formulario de edición (ver Figura 7-9). Cuando la sección está editada con los nuevos campos se pulsa sobre "Actualizar información".

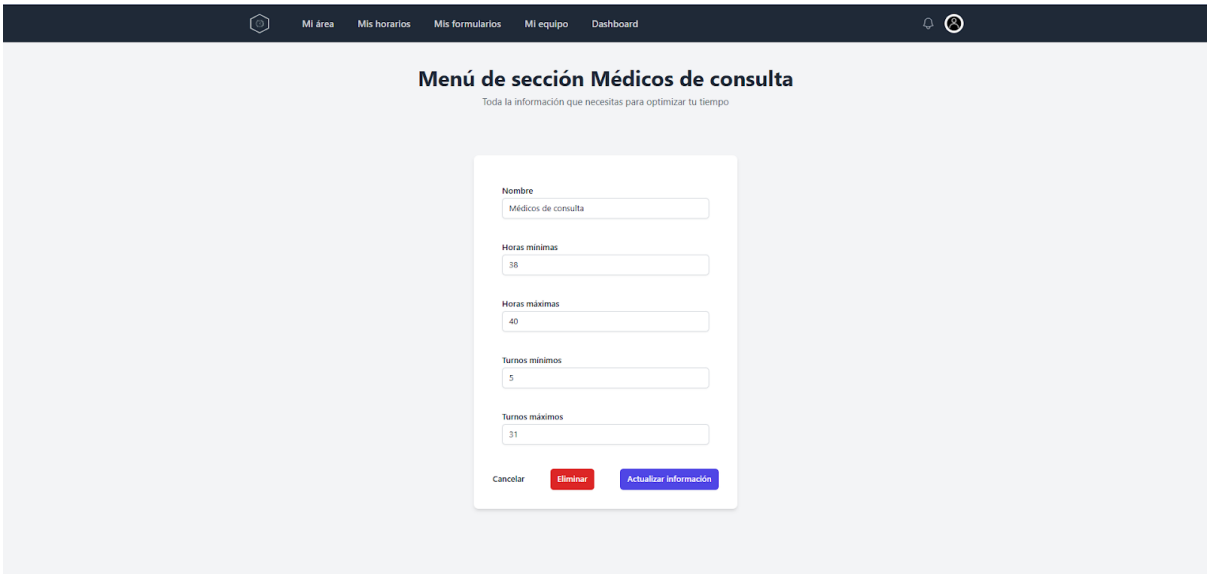

Figura 7-9. Edición de sección.

Por último, para eliminar una sección basta con volver al botón vertical de menú y clicar sobre "Eliminar sección". Para indicar que sección se elimina en específico se abre un pop-up para seleccionar la sección en cuestión. Con el objetivo de evitar errores se cuenta con un mensaje de confirmación para borrar la sección.

7.2.1.5 Gestión de horarios y turnos

Si vamos al acceso de Mis horarios en el encabezado de la página, se accede a todas las funcionalidades de administración de horarios, turnos y calendarios.

En la página principal de Mis horarios (ver Figura 7-10) se observan todos los horarios existentes con su título y el estado del horario. Se tiene la opción de acceder a distintas opciones pulsando sobre el botón de menú vertical (los tres puntos) de la parte superior derecha de cada tarjeta de horario, y desde el botón de Ver detalles se podrá ver más información sobre el horario en específico.

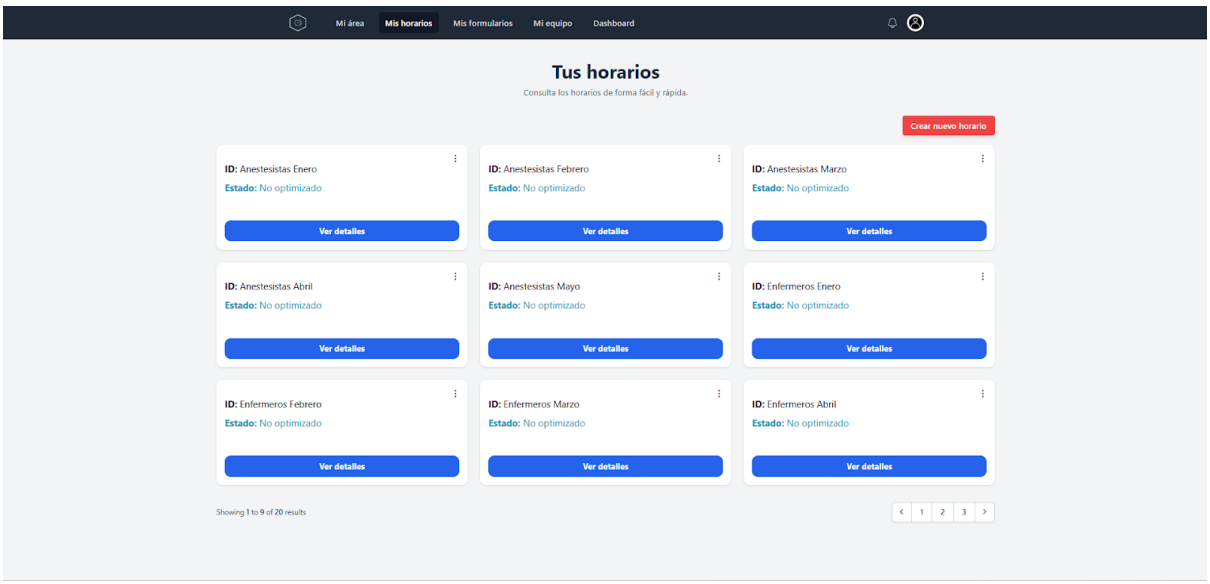

Figura 7-10. Vista principal de administración de horarios.

Al pulsar sobre el botón de "Crear nuevo horario" en la parte superior derecha de la página, podremos registrar un nuevo horario. Para ello se abrirá un formulario (ver

Figura 7-11) en el que habrá que rellenar los campos título de horario, descripción, sección a la que va a afectar este nuevo horario, fecha de inicio y fecha de fin. Para terminar con la creación del nuevo horario se clica sobre "Registrar horario".

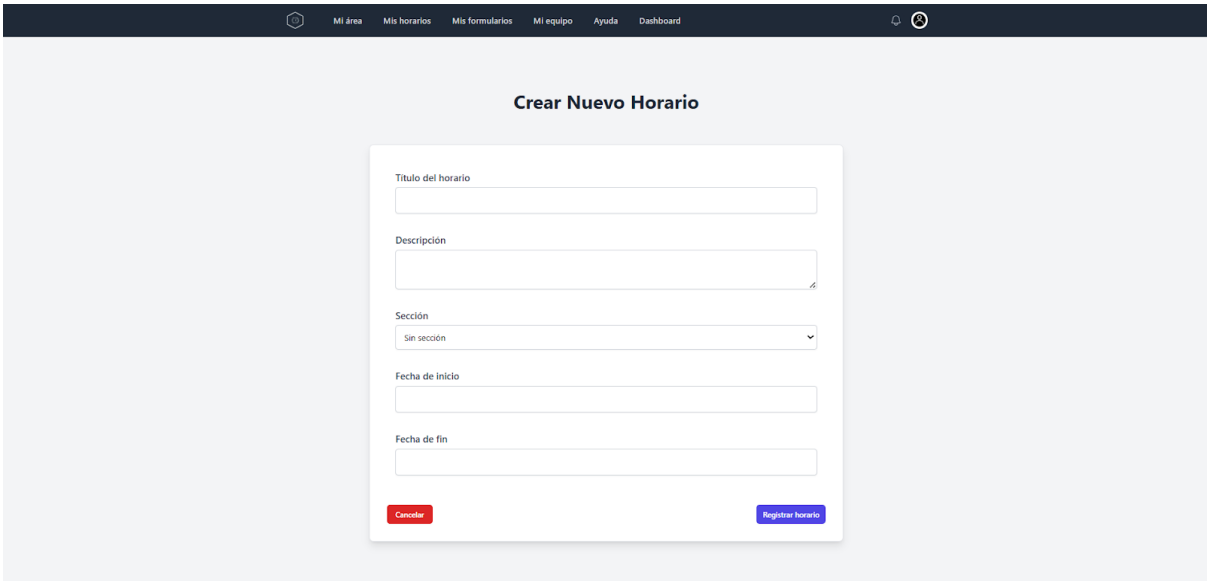

Figura 7-11. Registro de horario.

Al pulsar en "Editar" dentro de las opciones de gestión de un horario, se redirigirá al formulario de edición de horario (ver Figura 7-12). Desde este formulario se podrán editar todos los campos del horario. Además, en el formulario tenemos la posibilidad de añadir tipos de turnos al horario que estamos editando. Los tipos de turno aparecen en la parte inferior del formulario.

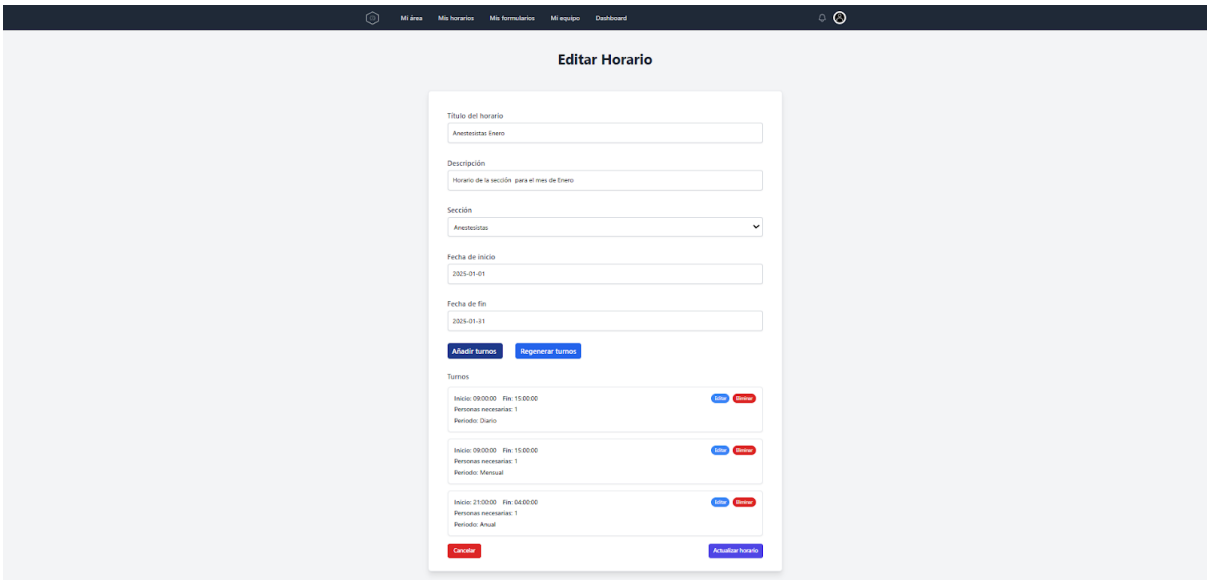

Figura 7-12. Edición de horario

Para añadir tipos de turno se pulsa sobre el botón de "Añadir turnos", tras lo cual nos aparecerá un formulario (ver Figura 7-13) que habrá que rellenar con los siguientes campos: día y hora de inicio del tipo de turno, día y hora de fin del tipo de turno, descripción del tipo de turno, trabajadores necesarios en el tipo de turno, periodo del tipo de turno (una sola vez, diario, semanal, mensual, o anual), y un *checkbox* para que el tipo de turno evite fines de semana o no. Si los tipos de turno se van a asignar a los trabajadores manualmente, desde este formulario se pueden indicar los empleados asignados para cada tipo de turno. En cambio, si los turnos se van a asignar con el algoritmo de optimización basado en las preferencias de los usuarios, no es necesario asignar trabajadores al tipo de turno. Clicando en el botón de "Registrar turno" se vuelve al formulario de edición de horario y se añade el tipo de turno al horario.

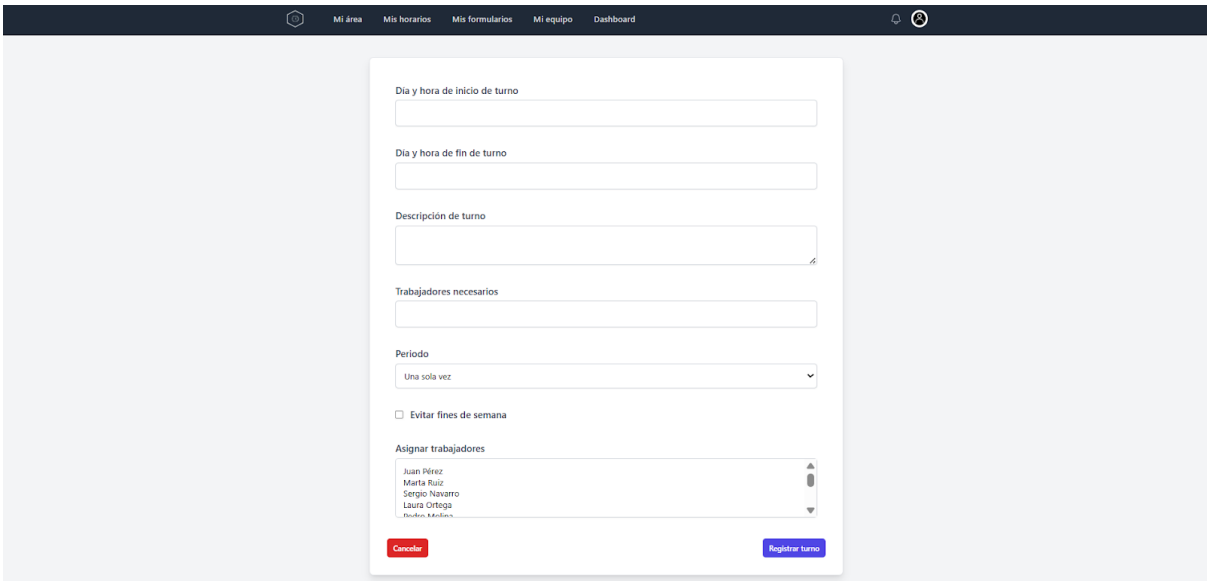

Figura 7-13. Registro de tipos de turno.

En el formulario de edición de horario, cada tipo de turno asociado al horario cuenta con botones de editar y eliminar.

En el caso de edición del tipo de turno se abrirá un nuevo formulario para editar los campos correspondientes. Una vez editado un tipo de turno se puede pulsar sobre el botón de "Regenerar turnos" para que los cambios de tipos de turno se vean reflejados en el horario y se actualice el calendario correspondiente.

Para eliminar un tipo de turno basta con pulsar el botón "Eliminar" del tipo de turno, habrá que confirmar el borrado de este registro. También se pueden regenerar turnos después de eliminar un tipo de turno.

En el momento de haber editado los campos del horario, haber añadido tipos de turno o ambas, se pulsa sobre el botón de "Actualizar horario" en el formulario del horario en el que se han hecho los cambios.

Otra de las opciones del horario consiste en eliminar el horario por completo, para ello se abre el desplegable de opciones y se pulsa sobre "Eliminar". En ese instante aparece un pop-up de confirmación para que no haya equivocaciones.

Si dentro de los tipos de turno asociados a un horario no se han asignado trabajadores manualmente por parte del gerente de RRHH, existe la opción de optimizar el horario para repartir los turnos con el algoritmo diseñado teniendo en cuenta las prioridades de los usuarios. Estas prioridades se recolectan con formularios específicos de los empleados. Para lograr esto, se abren las opciones de horario y se pulsa sobre "Optimizar". Tras esto, la aplicación ejecutará el microservicio de optimización para conseguir una asignación de turnos teniendo en cuenta la preferencia de los usuarios. El estado del horario pasará a optimizado. Además, se puede optimizar con la opción "Optimizar modo análisis", permitiendo ver si ha habido algún fallo durante la asignación.

Como última opción de horario están los cambios de turno. Al pulsar sobre "Cambio de turno" se abre una página con un calendario (ver Figura 7-14). Para cambiar un turno hay que seleccionar al trabajador con el que se quiere cambiar el turno, los dos turnos que se quieren intercambiar e indicar el motivo del cambio.

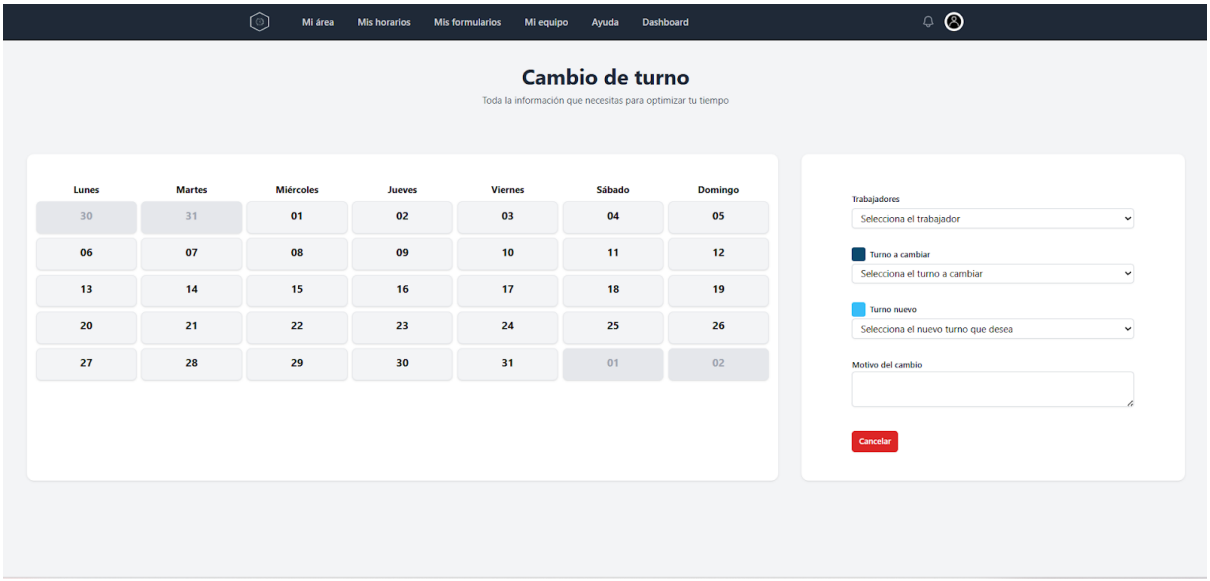

Figura 7-14. Cambio de turno desde administración.

Para cada horario, en cada tarjeta se tiene el botón de "Ver detalles". Al pulsar en este botón se accede al calendario de equipo de la sección específica del horario

(ver Figura 7-15). En cada día se pueden ver los turnos, y clicando sobre uno de ellos se observa información detallada sobre el turno como los trabajadores asignados al turno, horas de inicio y fin.

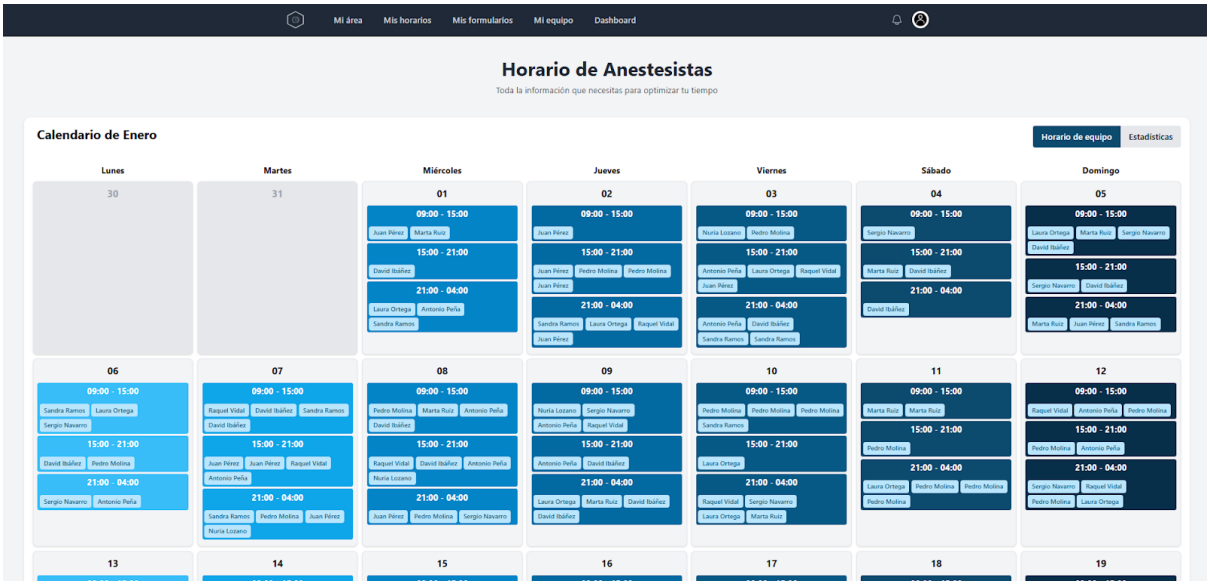

Figura 7-15. Horario específico de sección.

Además, si se pulsa sobre un turno en el calendario de equipo y ese turno todavía no tiene asignados los trabajadores necesarios para cubrir el turno, el gerente de RRHH puede asignar el turno manualmente desde aquí. Al abrir un turno (ver Figura 7-16), se pulsa sobre "Asignar turno", y seguidamente se abre la vista para poder asignar este turno. Este turno aparecerá en el calendario de la vista en cuestión. Se tiene que indicar a qué empleado se asigna el turno y una breve descripción para registrar la asignación del turno específico.

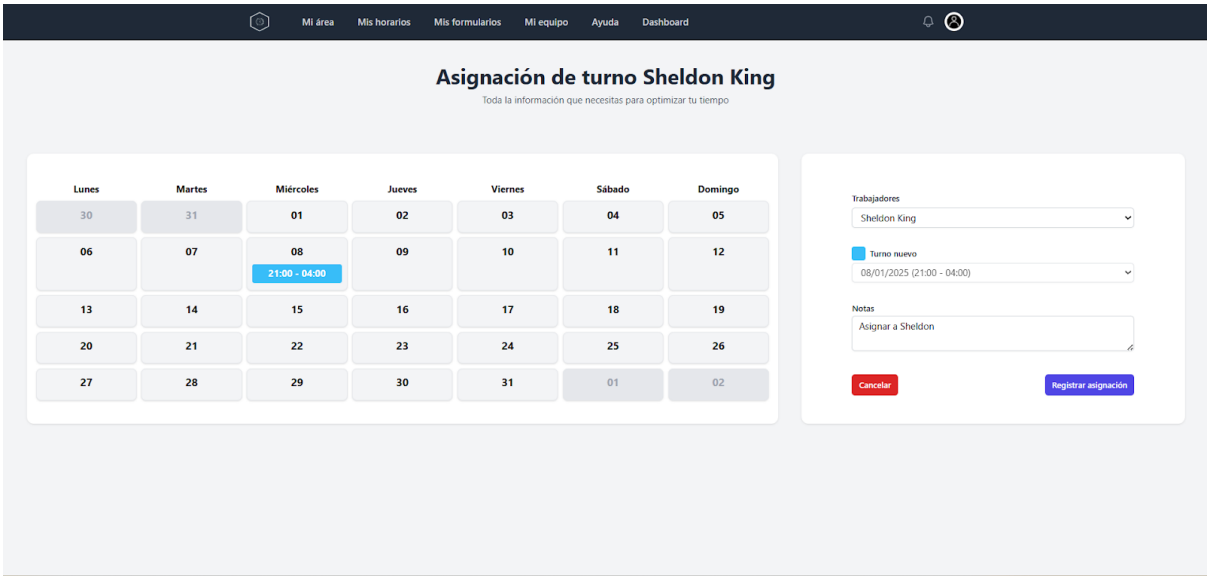

Figura 7-16. Asignación de turnos manual.

En la parte superior derecha del calendario se encuentra el acceso a las estadísticas, a través del botón con el mismo nombre. Si se pulsa sobre "Estadísticas", se visualizan las estadísticas del horario de la sección asociada (ver Figura 7-17), en la que se miden distintas métricas (como por ejemplo los días con mayor demanda de turnos o con mayor número de solicitudes de vacaciones aceptadas). En la página de estadísticas del horario de la sección se puede filtrar por meses.

Figura 7-17. Estadísticas de horario.

7.2.1.6 Gestión de formularios de preferencia

En el encabezado de la página encontramos el acceso a Mis formularios. A través de este acceso se tienen disponible todas las funcionalidades de gestión de formularios para preferencias de usuarios.

La página principal de formularios consiste en una serie de tarjetas (ver Figura 7-18) para cada formulario, con su respectivo título, rango de fechas para el formulario, secciones a las que afecta y funcionalidades de administración para la gestión de cada formulario.

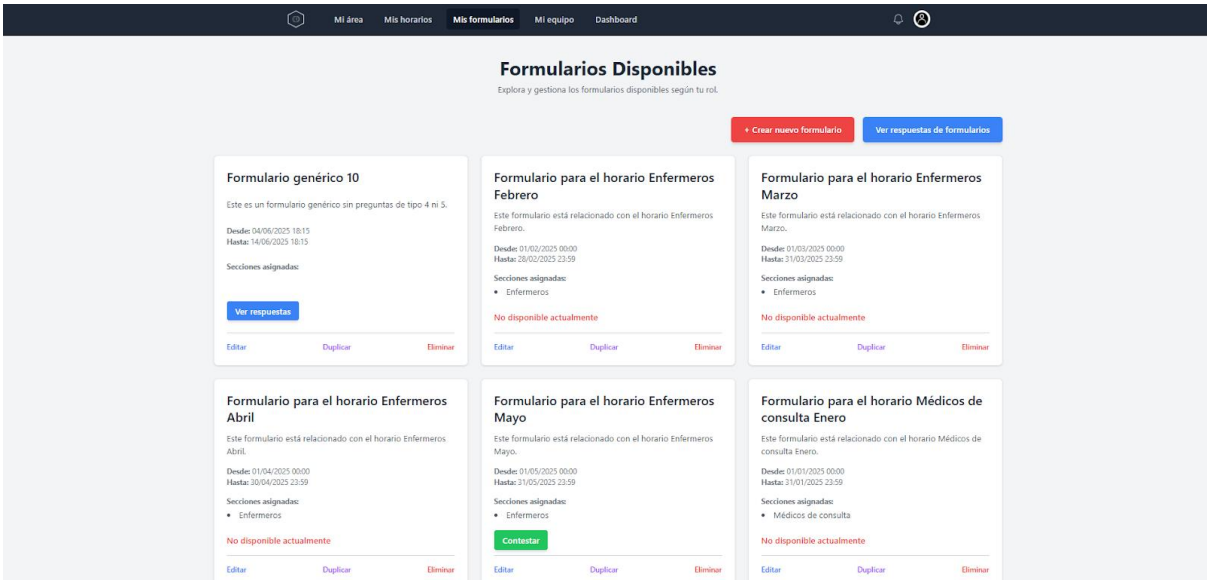

Figura 7-18. Vista de administración de formularios de preferencia.

En el caso de que el gerente de RRHH quiera crear un nuevo formulario de preferencia se pulsa sobre el botón "Crear nuevo formulario", localizado en la parte superior derecha de la pantalla. El formulario que se abre automáticamente es el que hay que rellenar para poder registrar un nuevo formulario de preferencia (ver Figura 7-19). Se tiene que indicar el título del formulario, resumen del formulario, comienzo y fin del plazo de respuesta, secciones a las que afecta el formulario y preguntas del formulario. Para agregar una nueva pregunta al formulario de preferencia, habrá que indicar el título de pregunta y el tipo de respuesta (calendario, selector, gradual, turnos, vacaciones, texto libre, opción múltiple, numérica o carga de archivo). Al pulsar sobre el botón de "Agregar pregunta", esta cuestión se añadirá al formulario de preferencia. Por último, una vez estén todos los campos requeridos se clica sobre "Crear formulario".

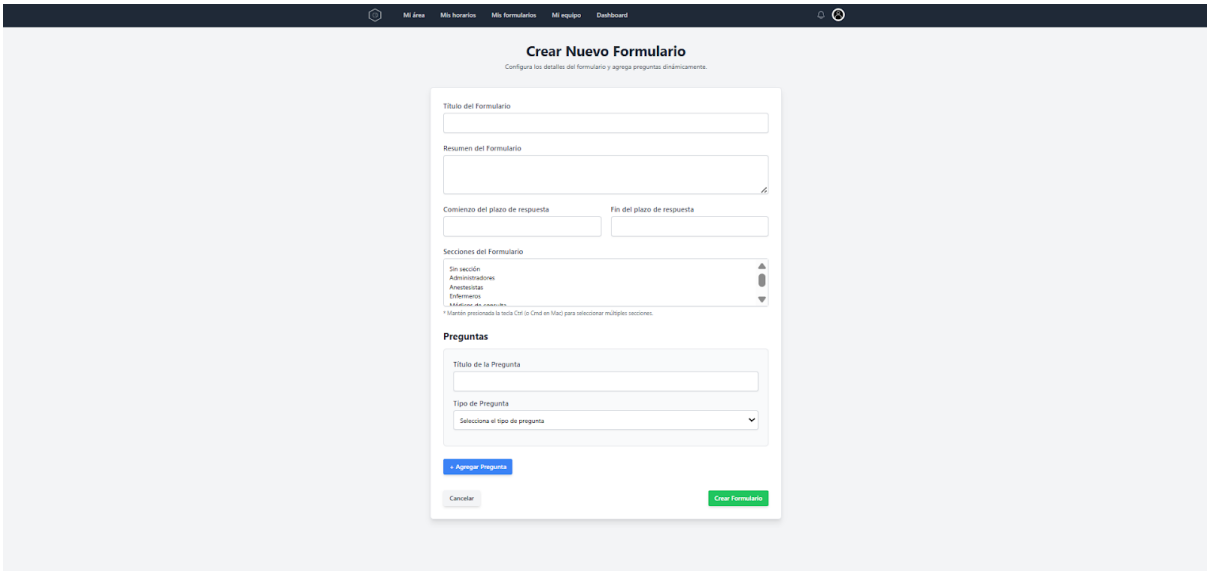

Figura 7-19. Registro de formulario de preferencia.

Si se desea editar un formulario, desde la vista principal de formularios se pulsa sobre el botón de "Editar" en la zona inferior izquierda de la tarjeta del formulario de preferencia en cuestión. Se abre el formulario de edición (ver Figura 7-20) para editar cualquiera de los campos. Los cambios se van a confirmar en el momento de pulsar el botón de "Actualizar formulario".

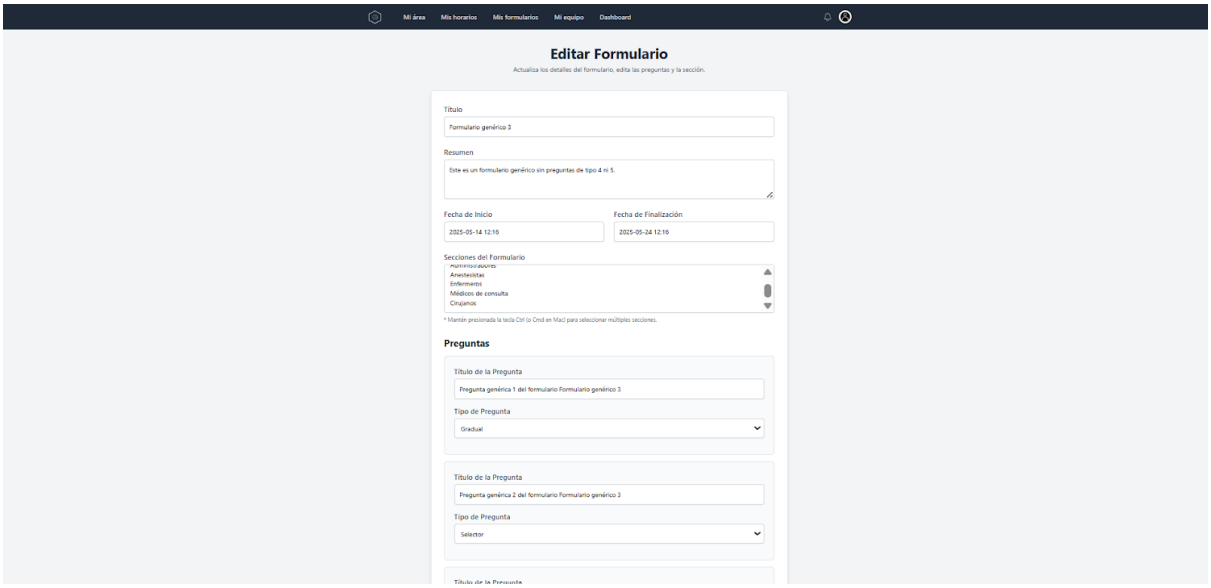

Figura 7-20. Edición de formulario de preferencia.

Con el objetivo de eliminar un formulario de preferencia, se clica sobre el botón de "Eliminar", localizado en la parte inferior derecha de la tarjeta del formulario de preferencia. Una vez pulsado hay que confirmar el borrado del formulario.

Otra funcionalidad disponible para formularios de preferencia es la posibilidad de duplicar un formulario, ayudando a una creación de formularios más eficiente. El formulario duplicado se puede editar y personalizar a gusto del gerente de RRHH. Para aprovechar esta funcionalidad hay que pulsar sobre "Duplicar" en la zona inferior de la tarjeta del formulario de preferencia que se quiere replicar.

En la vista principal de formularios, en la parte superior derecha, se encuentra el botón de "Ver respuestas de formularios". Pulsando sobre este botón se abre una pantalla para ver todas las respuestas a todos los formularios de preferencia de la plataforma, ayudando a la gestión de horarios y turnos (ver Figura 7-21). Para ser específicos con las respuestas que interesan al gerente de RRHH, existe la opción de filtrar las respuestas que interesan, indicando título del formulario, rango de fechas, usuario en concreto, sección e indicando si el formulario sigue activo o no. Se pueden ver las respuestas de un formulario en específico desde la propia tarjeta del formulario en cuestión.

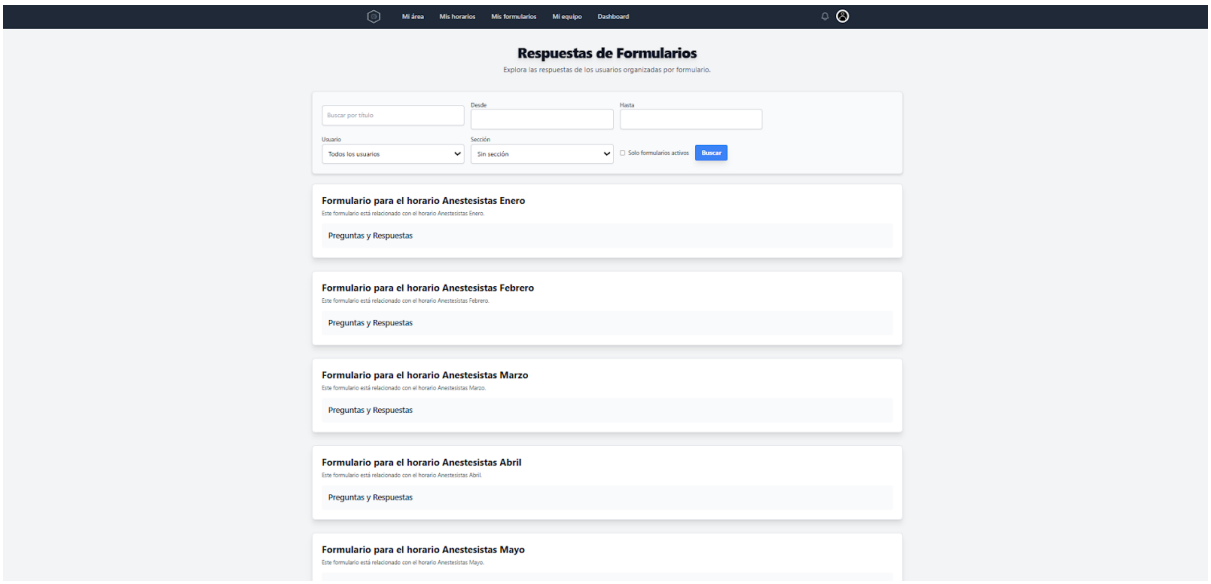

Figura 7-21. Respuestas de formularios de preferencia.

7.2.1.7 Estadísticas de empresa

Al pulsar sobre el acceso de *Dashboard* en la cabecera, nos aparecerá un cuadro de mandos (ver Figura 7-22) en el que se evalúan distintas métricas. En este *dashboard* interactivo se cuenta con distintos Indicadores Claves de Desempeño (KPIs por sus siglas en inglés, *Key Performance Indicator*) y distintas gráficas las cuales miden la situación de la compañía respecto a empleados, horarios, turnos y satisfacción.

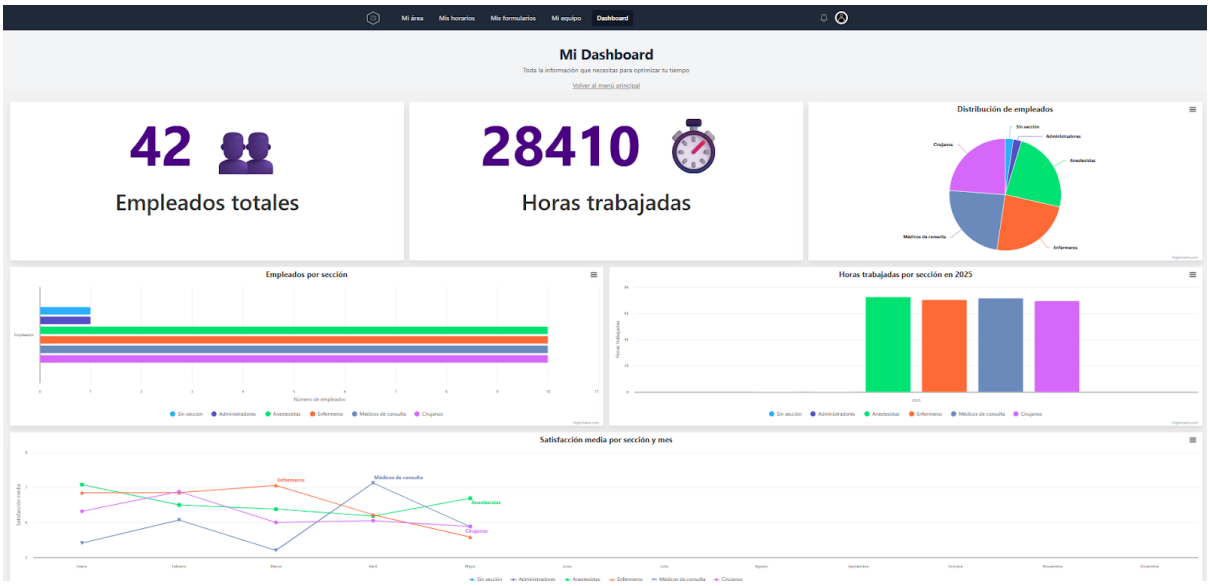

Figura 7-22. Estadísticas generales de la empresa.

Si se pulsa sobre el menú de hamburguesa (tres rayas) localizadas en la parte superior derecha de cada gráfica está habilitada la opción de ver gráfica en pantalla completa, imprimir gráfica y descargar en formato JPEG, PNG y SVG.

7.2.2 Manual de uso para Empleado

Desde el perfil de empleado, la aplicación está diseñada para ofrecer una experiencia sencilla e intuitiva, centrada en la respuesta de formularios, la recepción de notificaciones y la visualización de turnos asignados. A continuación, se detallan las principales funcionalidades accesibles para los trabajadores.

7.2.2.1 Inicio de sesión

El inicio de sesión para empleados es análogo al del rol de gerente de RRHH. Se introduce email y contraseña para poder acceder (ver Sección 7.2.1.1).

7.2.2.2 Área de usuario

Iniciando sesión, el usuario se encuentra con la vista de la Figura 7-23. Esta vista es muy similar a la explicada para el rol de gerente de RRHH, sin las opciones de edición.

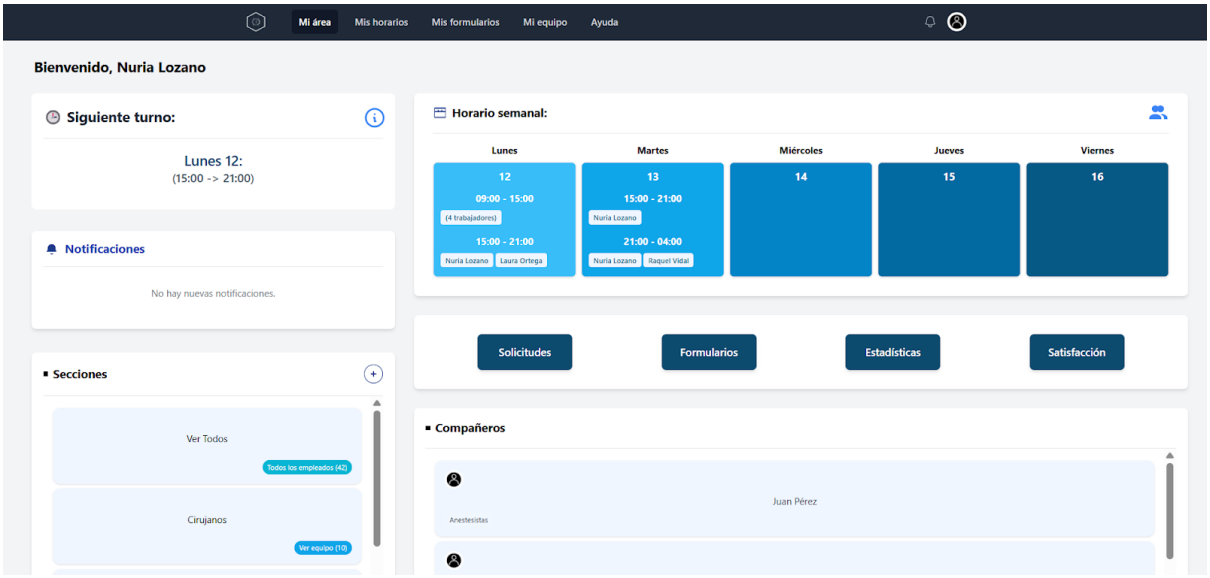

Figura 7-23. Vista de panel de usuario.

- En el encabezado de la página se encuentran distintos accesos para el usuario. Mi área (pantalla actual), Mis horarios (horarios de todos los meses), Mis formularios (formularios de preferencia respondidos y por responder), Mi equipo (empleados de la sección del empleado), Ayuda (preguntas frecuentes y contacto). En la zona derecha se encuentra una campana para acceso a notificaciones y foto de perfil, con acceso a edición de perfil propio, ajustes y cierre de sesión.
- Al igual que en el panel de administración para gerentes de RRHH, el cuerpo de la página está dividido en varios apartados. En la zona superior se encuentra el siguiente turno a realizar, con día y rango de hora, y se observa también el horario semanal con los días que trabaja el empleado. Si se pulsa sobre un turno en el horario semanal se abrirá un calendario mensual con un resumen del turno.

La parte media del cuerpo en la vista tiene un apartado de notificaciones donde se encuentran las más recientes, y otros accesos rápidos como "Solicitudes" (histórico de solicitudes), "Formularios" (acceso a mis formularios), "Estadísticas" (*dashboard* con estadísticas propias del usuario, ver Figura 7-24).

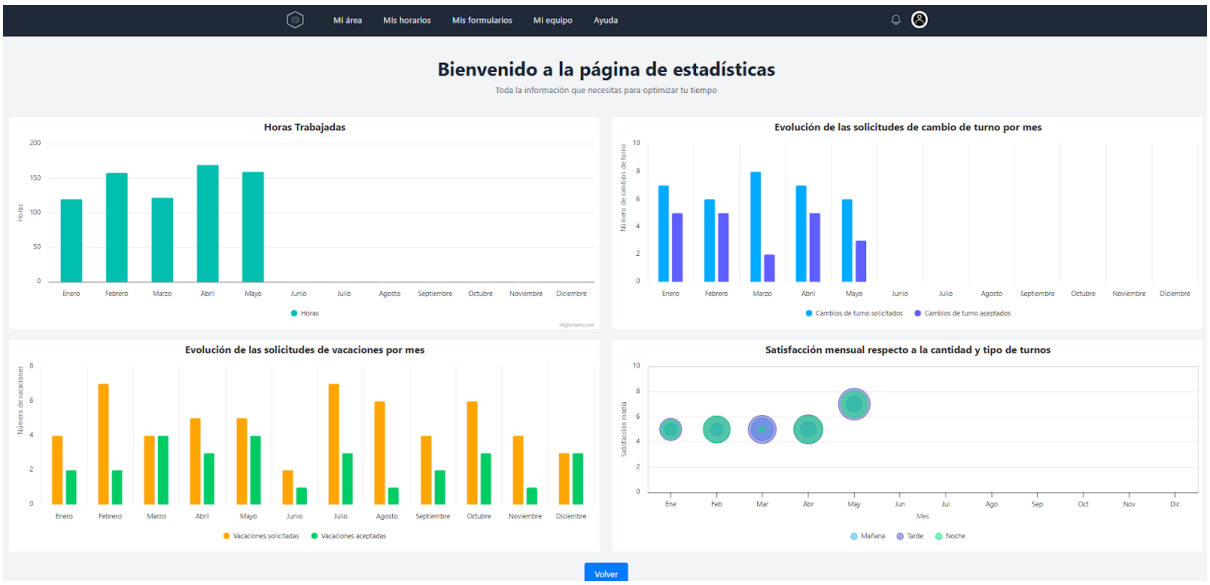

Figura 7-24. Estadísticas personales y de equipo.

"Satisfacción" (pop-up con satisfacción personal y de la sección, ver Figura 7-25).

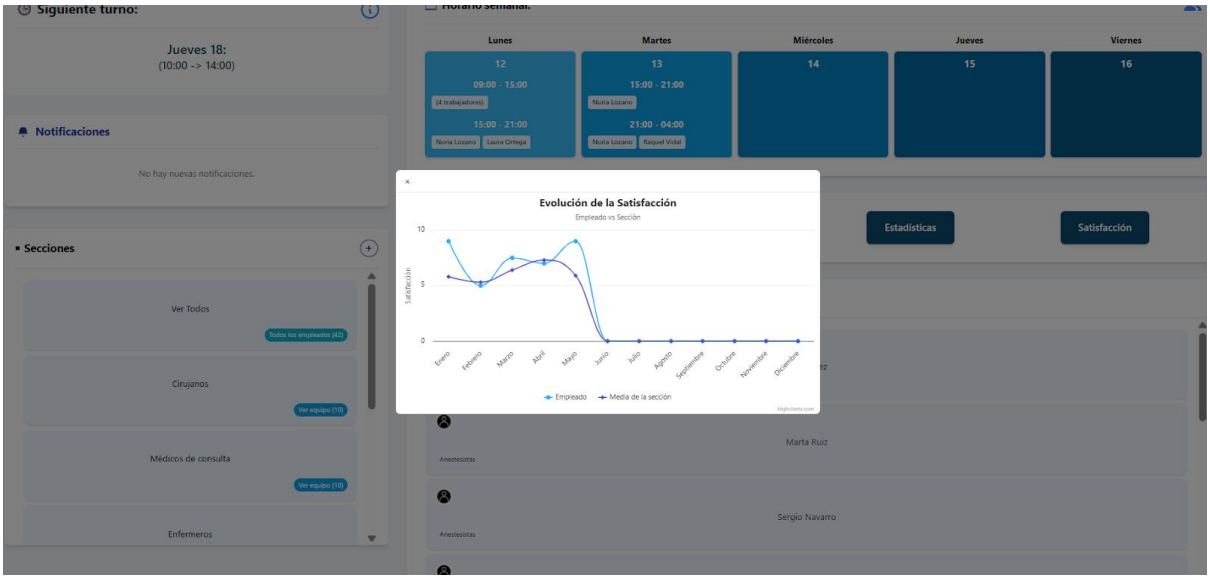

Figura 7-25. Evolución de satisfacción personal y de equipo.

La zona inferior del cuerpo consta del apartado "Secciones" de la empresa y "Compañeros". Secciones muestra la lista de las diferentes secciones definidas en la aplicación. Si se pulsa sobre el botón de *Ver equipo,* localizado en la tarjeta de una sección, se abrirá una página con información básica de todos los empleados de esa sección. Compañeros contiene una lista de los compañeros de la misma sección que el usuario. Al pulsar sobre cualquiera de ellos, se abre una barra lateral en la parte derecha con información y estadísticas sobre el compañero (Ver Figura 7-26).

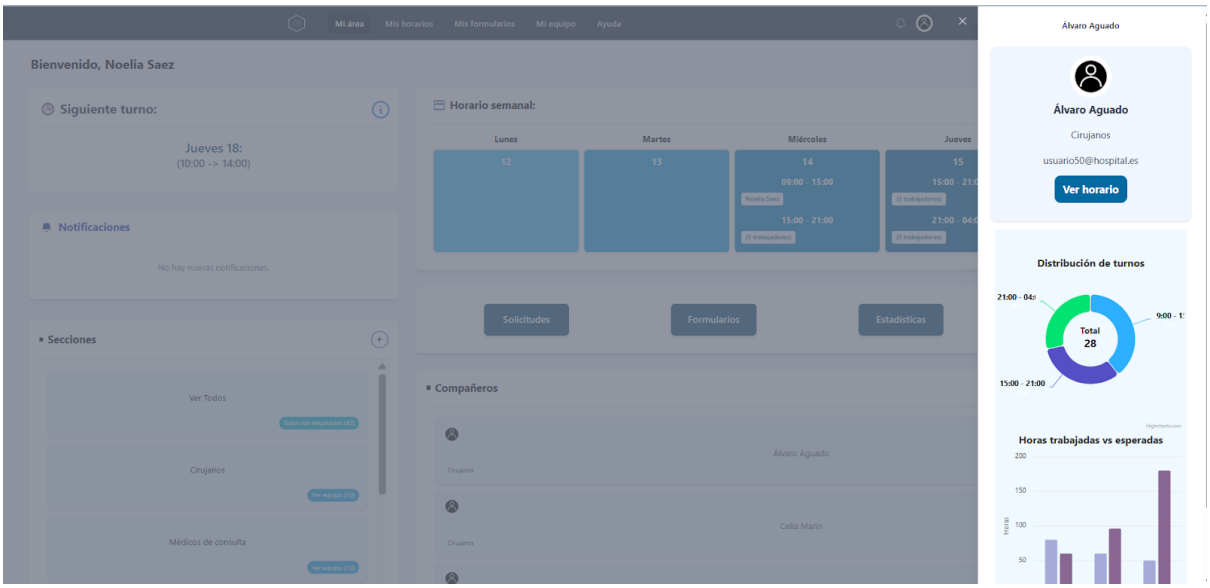

Figura 7-26. Información sobre compañeros.

7.2.2.3 Horarios

Desde la cabecera se puede acceder a la página de horarios del empleado. En esta vista aparecen los distintos horarios de los distintos meses que estén asignados, con la posibilidad de acceder a cualquiera de ellos mediante el botón de "Ver detalles" (Ver Figura 7-27).

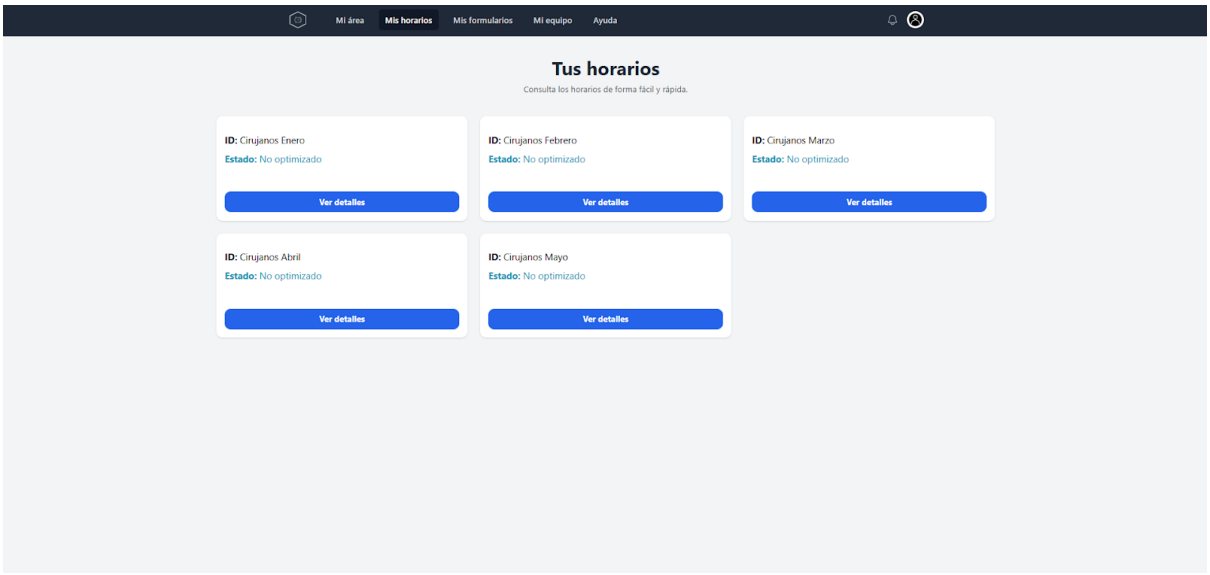

Figura 7-27. Vista principal de horarios.

Una vez dentro de un horario, se pueden ver 3 opciones: horario de equipo (ver Figura 7-28), horario personal (ver Figura 7-29) y estadísticas (ver Figura 7-30) sobre ese horario. Dentro de los horarios se puede seleccionar los turnos y trabajadores para ver información sobre estos.

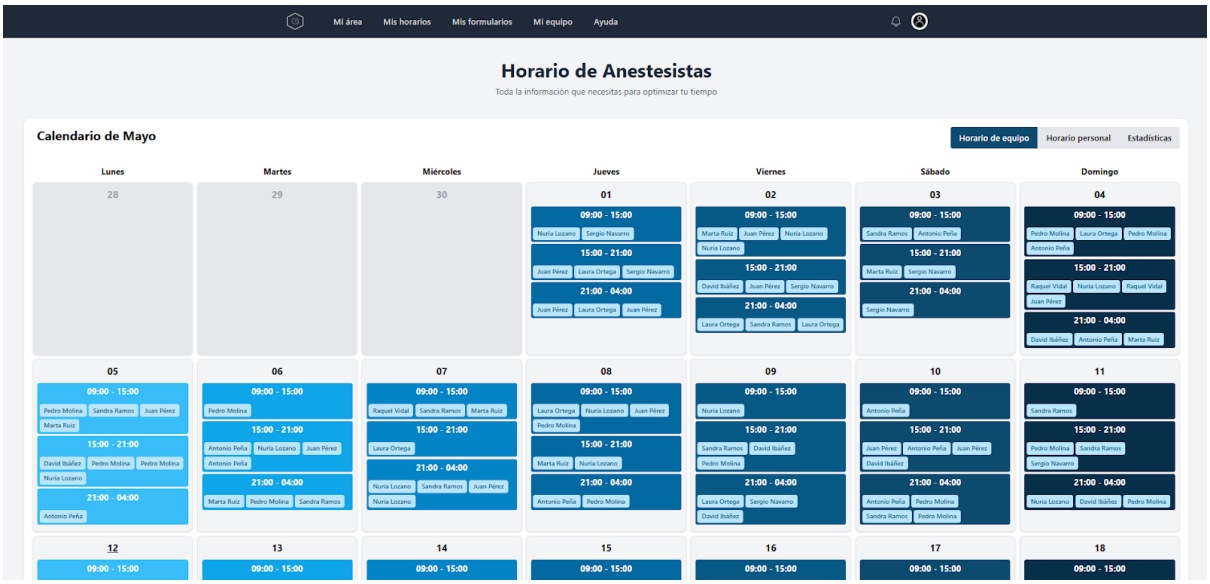

Figura 7-28. Horario de equipo.

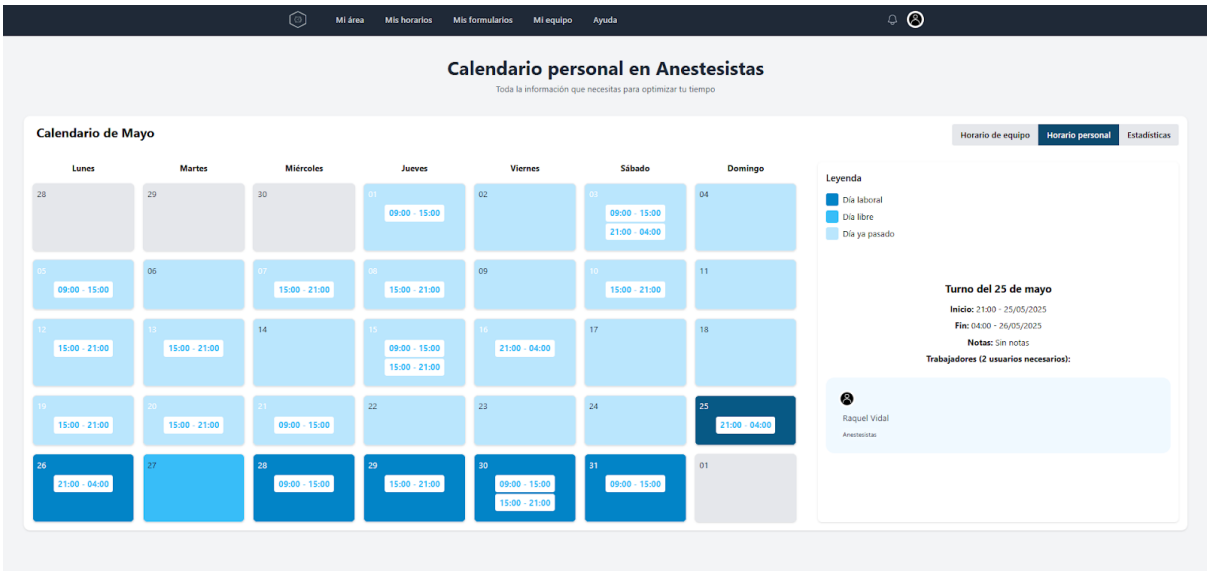

Figura 7-29. Horario personal.

Figura 7-30. Estadísticas del horario de equipo.

En el calendario de equipo, si se accede a un turno específico y se pulsa sobre el botón vertical de menú de la parte superior derecha en la descripción del turno, existe la opción de añadir el turno a Google Calendar [16] o Outlook [51]. Además se tiene la

posibilidad de solicitar un cambio de turno. Para esto se pulsa sobre la opción "Solicitar cambio de turno". Seguidamente se abre una página (ver Figura 7-31) en la que aparece un calendario con el turno que se desea solicitar. Como en el caso de lo descrito para el rol de gerente de RRHH, hay que indicar el turno antiguo que se quiere cambiar y el motivo de cambio. Esto no implica que el cambio esté autorizado, ya que este es gestionado por el personal de RRHH.

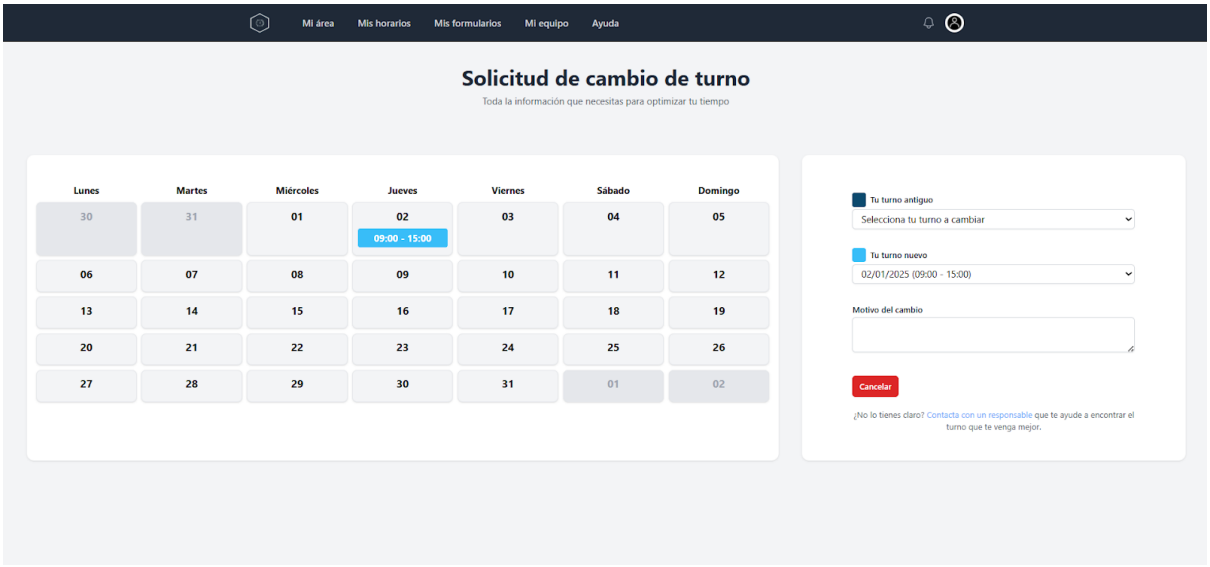

Figura 7-31. Solicitud para cambio de turno con un compañero.

7.2.2.4 Formularios

En el apartado de formularios del encabezado, se encuentran los formularios de preferencia completados y sin completar. Si un formulario no se ha rellenado tendrá un botón "Contestar" (ver Figura 7-32).

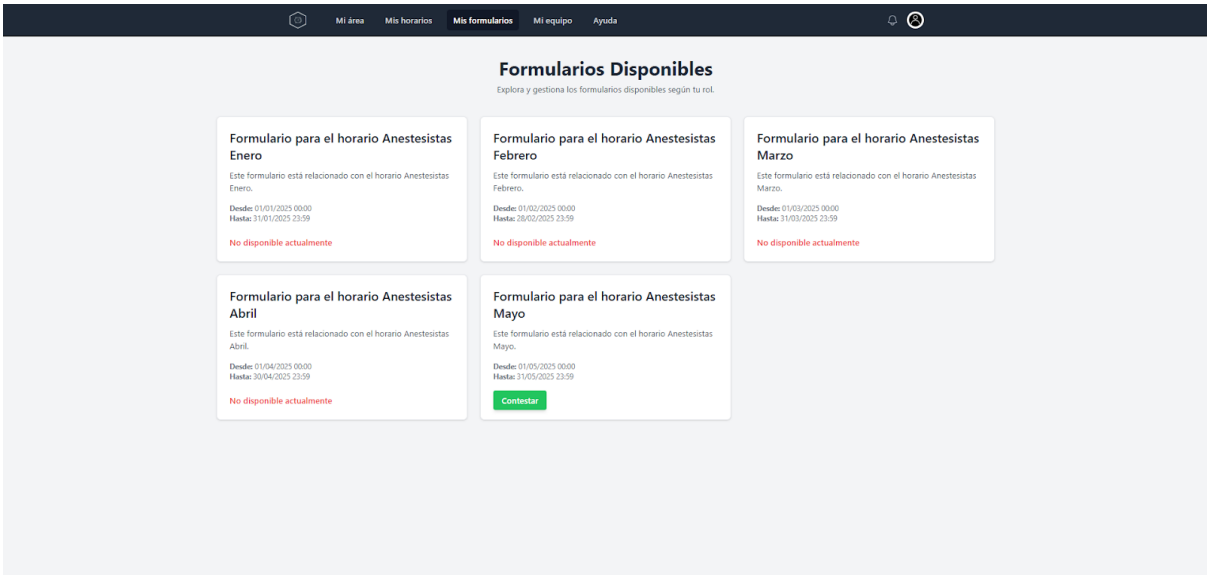

Figura 7-32. Vista principal de formularios de preferencia.

Una vez accedido al formulario de preferencia se contestan a las preguntas (ver Figura 7-33) del mismo y se pulsa sobre el botón "Enviar".

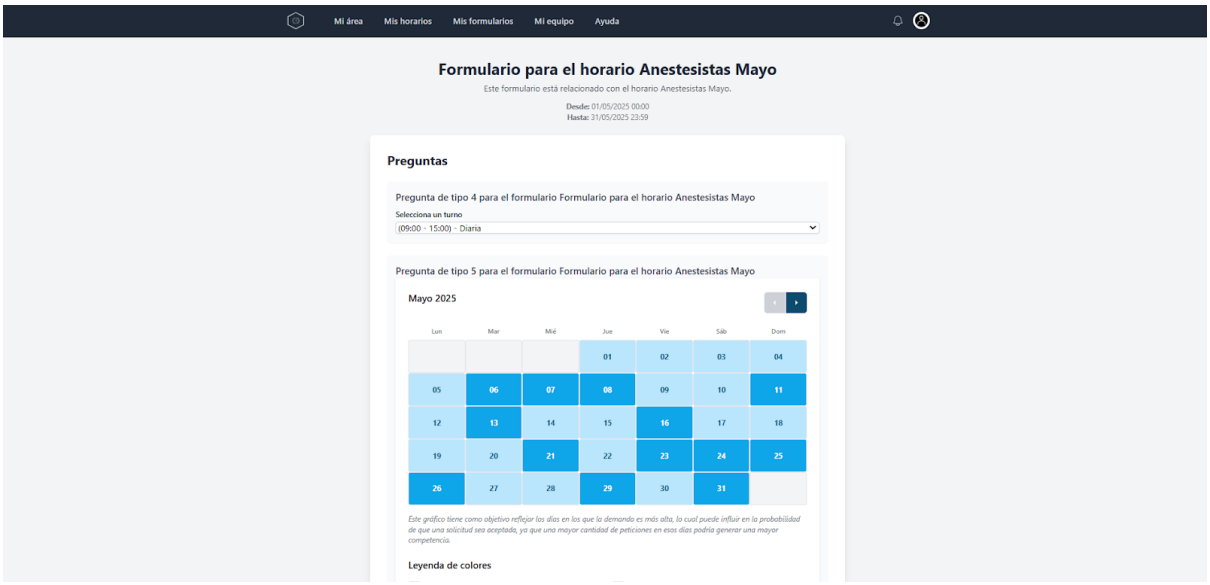

Figura 7-33. Formulario de preferencia a responder.

7.2.2.5 Equipo

Desde el encabezado de la página se puede acceder a la información del equipo/sección del usuario. Al pulsar sobre *Mi equipo* se abre la vista de equipo con una tarjeta informativa por cada compañero de sección (ver Figura 7-34). Dentro de cada tarjeta se pueden ver los datos principales de un empleado y se puede acceder al horario de este desde el botón "Ver horario". Además, los usuarios disponen de una barra de búsqueda para poder encontrar la información de un compañero.

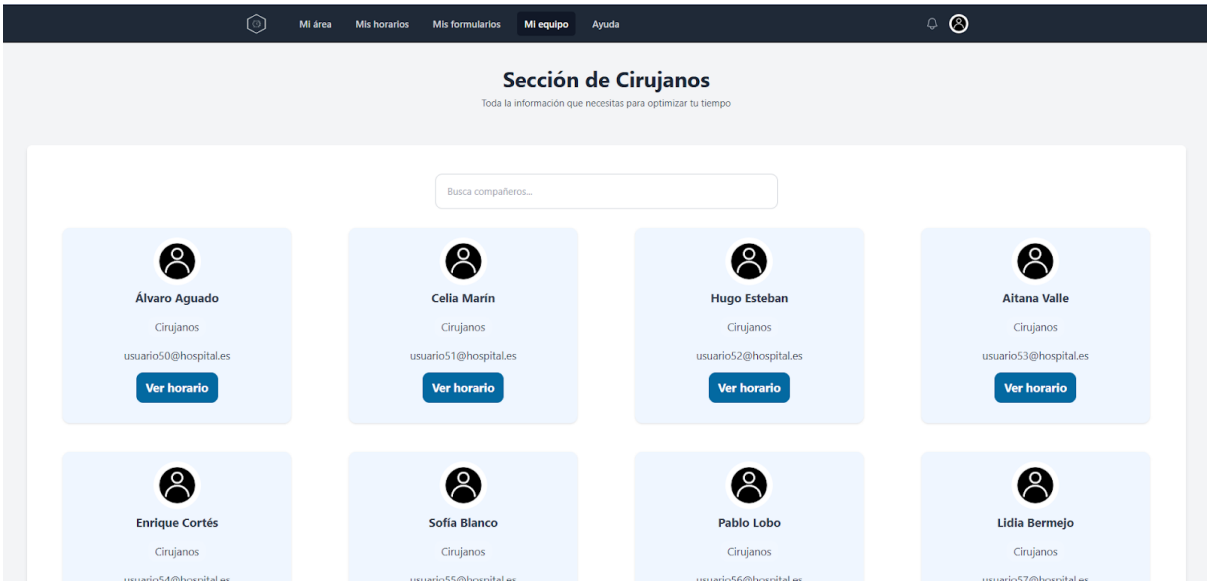

Figura 7-34. Vista de equipo.

7.2.2.6 Ayuda

En el caso de tener algún problema como usuario, se cuenta con la página de ayuda, a la que se accede por el acceso directo de Ayuda en el encabezado. En esta vista se encuentran distintas preguntas frecuentes que surgen a los empleados, además de un formulario de contacto para resolver dudas sobre funcionalidades más específicas (ver Figura 7-35).

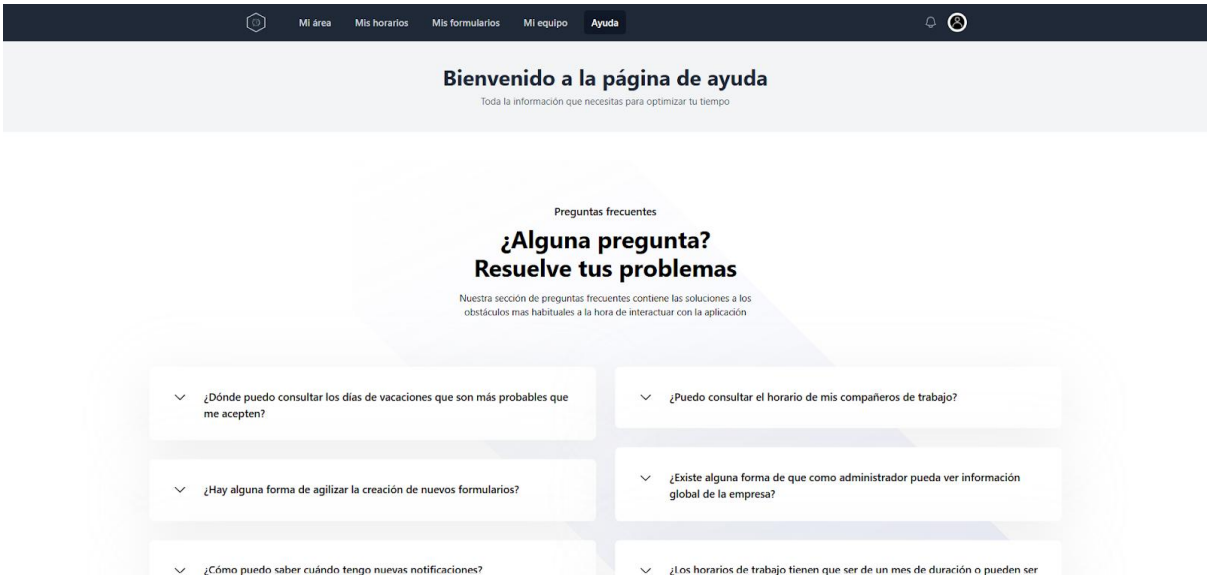

Figura 7-35. Vista de ayuda y preguntas frecuentes para usuarios.

Capítulo 8 - Conclusiones

El desarrollo de esta aplicación web ha implicado una fase de diseño, investigación, implementación, optimización y validación.

Uno de los principales retos del desarrollo ha sido la integración de tecnologías heterogéneas como Python [56] y PHP [52]. Ante la posibilidad de necesitar modificar librerías o incluso cambiar de lenguaje durante el proceso, se ha optado por una arquitectura basada en microservicios. Esta decisión permitió desacoplar las distintas partes del sistema, lo que facilitó el desarrollo en paralelo de funcionalidades independientes, agilizando así el proceso de implementación.

Además, esta arquitectura favorece el aprovechamiento de las fortalezas específicas de cada entorno: Laravel [38], orientado a la gestión de la lógica de negocio (usuarios, roles, formularios, etc.), y Python, que ofrece acceso a librerías especializadas como el resolutor Z3 [76] [87].

Por otro lado, la arquitectura basada en microservicios también prepara el sistema para una escalabilidad futura, permitiendo la integración de nuevos módulos o servicios externos sin comprometer la estabilidad ni el núcleo funcional de la aplicación.

El desarrollo en paralelo de las distintas funcionalidades, llegando incluso a estar en distintos entornos tecnológicos, hizo que la generación de datos de prueba fuese esencial. Estos datos no solo facilitaron el desarrollo de nuevas funcionalidades, sino también verificar que las funcionalidades ya implementadas continuaban funcionando correctamente tras cada nueva integración.

En un entorno distribuido como el de este proyecto, contar con un conjunto coherente y automatizado de datos de prueba ha sido clave para garantizar consistencia, fiabilidad e integridad durante todo el ciclo de desarrollo.

Por otra parte, la aplicación web basada en microservicios conlleva riesgos adicionales en términos de seguridad. Al dividir el sistema en múltiples componentes con distintas tecnologías, aumenta la superficie de exposición y por ende, el número de posibles ataques. Este riesgo exige una correcta configuración de los canales de comunicación, al igual que una gestión minuciosa de controles de acceso y

autenticación. Desde el punto de vista del *framework* utilizado, Laravel ofrece una base sólida en términos de seguridad (ver Sección 6.3) que han facilitado esta tarea.

En cuanto a los datos tratados por la aplicación, se maneja principalmente información de carácter personal como nombres, direcciones de correo electrónico, turnos y preferencias de los trabajadores. Aunque los datos relacionados con turnos o formularios puedan parecer inofensivos en un primer análisis, combinados con datos personales pueden permitir la identificación indirecta de los usuarios. Esto los convierte en datos personales protegidos bajo el Reglamento General de Protección de Datos (RGPD), lo que obliga a implementar medidas específicas de protección y transparencia.

En lo que respecta a la privacidad, la aplicación incluye una política de privacidad que informa sobre el uso y almacenamiento de datos, así como sobre derechos en relación con estos. Además se han seguido prácticas de minimización de datos, evitando almacenar información innecesaria.

Uno de los aspectos clave del sistema desarrollado es su enfoque en el diseño centrado en el usuario, el cual, desde el principio del desarrollo, ha guiado tanto el diseño de la interfaz como la definición de funcionalidades. Esta orientación ha permitido adaptar la aplicación a las necesidades de los distintos perfiles implicados en la gestión de horarios.

La realización de un estudio de usabilidad e interfaces en fases tempranas del desarrollo ha facilitado la identificación de patrones de uso, necesidades y puntos de fricción. Gracias a este enfoque, se ha facilitado de gran manera el desarrollo de una interfaz intuitiva, clara y accesible para todos los roles implicados.

Además, la automatización de ciertos procesos, como la validación de restricciones, la resolución de solapamientos o la detección de jornadas desequilibradas, ayuda a minimizar errores humanos comunes en la planificación manual.

A diferencia de otras herramientas más genéricas o rígidas de gestión de horarios, la aplicación planteada destaca por ofrecer una solución con una curva de aprendizaje reducida gracias a su interfaz clara, y que destaca por ofrecer una

planificación personalizada, flexible y orientada al bienestar del trabajador, permitiendo una planificación más justa y humana.

Capítulo 9 - Futuras líneas de trabajo

Aunque la aplicación desarrollada cumple con los objetivos planteados y ofrece una solución robusta y competitiva, existen diversas áreas de mejora que podrían potenciar distintos aspectos como la experiencia de usuario o la internacionalización.

Una línea clara de mejora se encuentra en la ampliación del conjunto de restricciones que el optimizador maneja. Para lograr una planificación aún más precisa y personalizada, sería conveniente añadir nuevas restricciones, tanto opcionales como obligatorias, lo que requeriría también diseñar nuevas preguntas en la creación de formularios.

En este sentido, resulta relevante incorporar la opción de añadir restricciones que aseguren el cumplimiento estricto de la legislación laboral española, como límites en horas máximas de trabajo, descansos obligatorios, y normativas sobre turnos rotativos o nocturnos. Además, al haberlo implementado como un microservicio, es posible integrar distintas técnicas y algoritmos de resolución. En esta línea se puede estudiar la eficiencia de la técnica propuesta e implementar una solución híbrida que combine un algoritmo voraz junto al resolutor.

En paralelo, y considerando un entorno cada vez más globalizado y con equipos distribuidos geográficamente, es imprescindible que la aplicación soporte la generación y visualización de horarios para trabajadores en distintas zonas horarias. Esto requiere adaptar la lógica del optimizador y la interfaz para gestionar correctamente las diferencias horarias, garantizando que los turnos asignados respeten las horas locales de cada empleado. Para ello se podría modificar la representación de los turnos y zonas horarias de tal manera que la parte del *backend* trabaje con una representación *AoE*, mientras que la parte del *frontend* particularice las horas a la región horaria desde la que se acceda a la aplicación.

En línea con el soporte de múltiples zonas horarias, la internacionalización es una estrategia clave para aumentar la accesibilidad y el alcance comercial de la aplicación. Implementar una aplicación multilenguaje implica diseñar un sistema de

traducciones que permita adaptar la interfaz, formularios y notificaciones a distintos idiomas de forma mantenible.

Desde un punto de vista técnico, actualmente el sistema utiliza SQLite [62] [83], una base de datos ligera ideal para entornos de desarrollo. Sin embargo para un entorno de producción con altas cargas y una gran comunidad de usuarios, sería necesario migrar a un sistema de bases de datos más robusto y escalable como PostgreSQL [53] o MySQL [44]. Este cambio no conllevaría grandes esfuerzos debido a la abstracción de código de Laravel, y ayudaría a gestionar mejor la concurrencia y la confianza general de la aplicación.

Debido al creciente uso de dispositivos móviles en el entorno laboral, el desarrollo de una aplicación móvil sería una mejora imprescindible, ya que facilitaría la accesibilidad y usabilidad desde cualquier lugar. Además, una app móvil puede integrar funcionalidades adicionales como notificaciones *push* o métodos de verificación en dos pasos (2FA [94]), aumentando la seguridad y la interacción con los usuarios.

Respecto al sistema de notificaciones, la versión actual de la aplicación permite transmitir avisos desde su interfaz. Sin embargo, sería conveniente disponer de formas alternativas de hacer llegar notificaciones a los usuarios. Un método recurrente para ello es el envío de correos electrónicos, facilitado por SMTP [61], protocolo estándar para el envío de emails. Laravel incorpora un soporte nativo para este protocolo [39], ya mencionado en la Sección 4.1, lo que permitiría su integración de forma sencilla. De esta forma, cada vez que se solicitara un cambio de turno o se solicitara ayuda a Administración, los usuarios involucrados podrían recibir notificaciones tanto por la aplicación como por correo electrónico. Además, esto aportaría opcionalidad en cuanto a configuración, ya que llevaría a implementar una personalización aún mayor del sistema de notificaciones permitiendo a los usuarios elegir las vías de notificación que deseen. En cuanto a su implementación, se podrían utilizar herramientas ya presentes en Laravel como el sistema de colas (Queues [57]) o notificaciones (Notifications [45]) para enviar los correos de forma asíncrona. Esto permitiría integrar el envío de correos mediante servicios corporativos como Outlook [51], ampliamente usados por las empresas, configurando tan solo parámetros en el archivo .env. Esta

integración profesionalizaría aún más la herramienta y mejoraría el sistema de comunicación con los usuarios.

Otra línea de mejora para futuras versiones consistiría en la integración con plataformas externas ampliamente utilizadas en entornos laborales, como Microsoft Teams [41] o Slack [60]. En concreto, es posible conectarse a sus APIs públicas, permitiendo enviar notificaciones automáticas a los usuarios, crear eventos en el calendario personal del trabajador directamente desde TimeFlex, sincronizar ausencias para la planificación o incorporar *bots* que permitan interactuar con TimeFlex directamente desde el chat corporativo. Estas integraciones no solo mejorarían la experiencia de uso, al permitir que los usuarios accedan a la información sin tener que cambiar de plataforma, sino que también permitirían automatizar procesos y mejorar la comunicación interna.

Finalmente, la incorporación de *chatbots* sería una innovación significativa para la interacción con los usuarios. Un *chatbot* podría asistir en la generación y cumplimentación de formularios, responder dudas frecuentes y facilitar la consulta o modificación de horarios de forma conversacional. Esta herramienta no solo mejoraría la experiencia del usuario, sino que también agilizaría el proceso y reduciría la curva de aprendizaje, convirtiendo la aplicación desarrollada en una solución aún más completa y moderna.

Bibliografía

[1] «Alpine.js: Your new, lightweight, JavaScript framework,» [En línea]. Available: https://alpinejs.dev/. [Último acceso: 2025].

[2] «Best Software Companies Products,» G2, [En línea]. Available: https://www.g2.com/best-software-companies/top-products. [Último acceso: 2025].

[3] «Blade Templates,» Laravel, [En línea]. Available: https://laravel.com/docs/12.x/blade. [Último acceso: 2025].

[4] «Blowfish,» CNN-CERT, [En línea]. Available: https://www.ccn-cert.cni.es/publico/seriesCCN-STIC/series/400-Guias_Generales/401-glosario_abreviaturas/index.html?n=138.html. [Último acceso: 2025].

[5] «CakePHP Vs. Laravel: Choosing the Right PHP Framework for Your Project,» Orient Software, [En línea]. Available: https://www.orientsoftware.com/blog/cakephp-vs-laravel/. [Último acceso: 2025].

[6] «CakePHP,» CakePHP, [En línea]. Available: https://cakephp.org/. [Último acceso: 2025].

[7] «CanvasJS vs Highcharts,» Stackshare, [En línea]. Available: https://stackshare.io/stackups/canvasjs-vs-highcharts. [Último acceso: 2025].

[8] «CanvasJS: Beautiful JavaScript Charting Library,» CanvasJS, [En línea]. Available: https://canvasjs.com/. [Último acceso: 2025].

[9] «Carbon,» Carbon, [En línea]. Available: https://carbon.nesbot.com/docs/. [Último acceso: 2025].

[10] «CodeIgniter vs Laravel: A Detailed Side-by-Side Comparison,» Kinsta, [En línea]. Available: https://kinsta.com/blog/codeigniter-vs-laravel/. [Último acceso: 2025].

[11] «CodeIgniter: The small framework with powerful features,» CodeIgniter, [En línea]. Available: https://codeigniter.com/ . [Último acceso: 2025].

[12] «Construction heuristics,» OptaPlanner, [En línea]. Available: https://www.optaplanner.org/docs/optaplanner/latest/construction-heuristics/construction-heuristics.html. [Último acceso: 2025].

[13] «CORS,» Mozilla, [En línea]. Available: https://developer.mozilla.org/es/docs/Web/HTTP/Guides/CORS. [Último acceso: 2025].

[14] «CSS,» W3, [En línea]. Available: https://www.w3.org/Style/CSS/Overview.en.html. [Último acceso: 2025].

[15] «Chart.js,» Chart.js, [En línea]. Available: https://www.chartjs.org/. [Último acceso: 2025].

[16] «Descripción general de la API de Google Calendar,» Google, [En línea]. Available: https://developers.google.com/workspace/calendar/api/guides/overview?hl=es-419. [Último acceso: 2025].

[17] «Django: Django makes it easier to build better web apps more quickly and with less code.,» Django, [En línea]. Available: https://www.djangoproject.com/. [Último acceso: 2025].

[18] «Documentación de GitHub,» GitHub, [En línea]. Available: https://docs.github.com/es. [Último acceso: 2025].

[19] «Eloquent,» Laravel, [En línea]. Available: https://laravel.com/docs/12.x/eloquent. [Último acceso: 2025].

[20] «Facilita RGPD,» AEPD, [En línea]. Available: https://facilita.aepd.es/. [Último acceso: 2025].

[21] «FastAPI,» FastAPI, [En línea]. Available: https://fastapi.tiangolo.com/. [Último acceso: 2025].

[22] «Flask Documentation,» Flask, [En línea]. Available: https://flask.palletsprojects.com/en/stable/. [Último acceso: 2025].

[23] «Flatpickr,» Flatpickr, [En línea]. Available: https://flatpickr.js.org/. [Último acceso: 2025].

[24] «Free Timetabling Software Description,» FET, [En línea]. Available: https://lalescu.ro/liviu/fet/. [Último acceso: 2025].

[25] «Heatmap,» Highcharts, [En línea]. Available: https://www.highcharts.com/docs/chart-and-series-types/heatmap. [Último acceso: 2025].

[26] «Highcharts Documentation,» Highcharts, [En línea]. Available: https://www.highcharts.com/docs/index. [Último acceso: 2025].

[27] «Highcharts vs Chart.js,» Stackshare, [En línea]. Available: https://stackshare.io/stackups/highcharts-vs-js-chart. [Último acceso: 2025].

[28] «Highcharts vs Matplotlib,» Stackshare, [En línea]. Available: https://stackshare.io/stackups/highcharts-vs-matplotlib. [Último acceso: 2025].

[29] «HTML Canvas Graphics,» W3Schools, [En línea]. Available: https://www.w3schools.com/html/html5_canvas.asp. [Último acceso: 2025].

[30] «Hyperheuristics,» OptaPlanner, [En línea]. Available: https://www.optaplanner.org/docs/optaplanner/latest/hyperheuristics/hyperheuristic.html. [Último acceso: 2025].

[31] «Laravel Backup,» [En línea]. Available: https://spatie.be/docs/laravel-backup/v9/introduction. [Último acceso: 2025].

[32] «Laravel Breeze,» Laravel, [En línea]. Available: https://laravel.com/docs/11.x/starter-kits. [Último acceso: 2025].

[33] «Laravel CORS,» Laravel, [En línea]. Available: https://laravel.com/docs/12.x/routing#cors. [Último acceso: 2025].

[34] «Laravel CSRF,» Laravel, [En línea]. Available: https://laravel.com/docs/11.x/csrf. [Último acceso: 2025].

[35] «Laravel Faker,» Laravel, [En línea]. Available: https://laravel.com/docs/7.x/database-testing. [Último acceso: 2025].

[36] «Laravel Jetstream,» Laravel Jetstream, [En línea]. Available: https://jetstream.laravel.com/introduction.html. [Último acceso: 2025].

[37] «Laravel Sanctum,» Laravel, [En línea]. Available: https://laravel.com/docs/12.x/sanctum. [Último acceso: 2025].

[38] «Laravel,» Laravel, [En línea]. Available: https://laravel.com/docs/12.x. [Último acceso: 2025].

[39] «Mail,» Laravel, [En línea]. Available: https://laravel.com/docs/11.x/mail. [Último acceso: 2025].

[40] «Matplotlib: Visualization with Python,» Matplotlib, [En línea]. Available: https://matplotlib.org/. [Último acceso: 2025].

[41] «Microsoft Teams,» Microsoft, [En línea]. Available: https://www.microsoft.com/es-es/microsoft-teams/group-chat-software/. [Último acceso: 2025].

[42] «Middleware,» Laravel, [En línea]. Available: https://laravel.com/docs/12.x/middleware. [Último acceso: 2025].

[43] «MongoDB,» MongoDB, [En línea]. Available: https://www.mongodb.com/. [Último acceso: 2025].

[44] «MySQL,» MySQL, [En línea]. Available: https://www.mysql.com/. [Último acceso: 2025].

[45] «Notifications,» Laravel, [En línea]. Available: http://laravel.com/docs/11.x/notifications. [Último acceso: 2025].

[46] «npm Docs,» npm, [En línea]. Available: https://docs.npmjs.com/. [Último acceso: 2025].

[47] «npm,» npm, [En línea]. Available: https://www.npmjs.com/. [Último acceso: 2025].

[48] «Opcenter Scheduling SMT,» Siemens, [En línea]. Available: https://plm.sw.siemens.com/en-US/opcenter/advanced-planning-scheduling-aps/advanced-scheduling-smt/. [Último acceso: 2025].

[49] «OptaPlanner,» OptaPlanner, [En línea]. Available: https://www.optaplanner.org/docs/optaplanner/latest/planner-introduction/planner-introduction.html. [Último acceso: 2025].

[50] «Optaweb Employee Rostering User Guide,» Optaweb Employee Rostering, [En línea]. Available: https://docs.optaplanner.org/8.7.0.Final/optaweb-employee-rostering-docs/html_single/. [Último acceso: 2025].

[51] «Outlook Developer documentation,» Microsoft, [En línea]. Available: https://learn.microsoft.com/en-us/outlook/. [Último acceso: 2025].

[52] «php,» [En línea]. Available: https://www.php.net/. [Último acceso: 2025].

[53] «PostgreSQL: The World's Most Advanced Open Source Relational Database,» PostgreSQL, [En línea]. Available: https://www.postgresql.org/. [Último acceso: 2025].

[54] «Production planning and scheduling,» Siemens, [En línea]. Available: https://plm.sw.siemens.com/en-US/opcenter/production-planning-scheduling-capabilities/. [Último acceso: 2025].

[55] «Pydantic,» Pydantic, [En línea]. Available: https://docs.pydantic.dev/latest/. [Último acceso: 2025].

[56] «Python,» Python, [En línea]. Available: https://www.python.org/. [Último acceso: 2025].

[57] «Queues,» Laravel, [En línea]. Available: https://laravel.com/docs/11.x/queues. [Último acceso: 2025].

[58] «React,» React, [En línea]. Available: https://es.react.dev/. [Último acceso: 2025].

[59] «Shield Documentation,» CodeIgniter, [En línea]. Available: https://shield.codeigniter.com/. [Último acceso: 2025].

[60] «Slack,» Slack, [En línea]. Available: https://slack.com/intl/es-es. [Último acceso: 2025].

[61] «SMTP Email API Documentation,» SMTP, [En línea]. Available: https://www.smtp.com/resources/api-documentation/. [Último acceso: 2025].

[62] «SQLite Documentation,» SQLite, [En línea]. Available: https://www.sqlite.org/docs.html. [Último acceso: 2025].

[63] «Stack,» Stack, [En línea]. Available: https://www.stack.io/services/logging. [Último acceso: 2025].

[64] «Starlette,» Starlette, [En línea]. Available: https://www.starlette.io/. [Último acceso: 2025].

[65] «Symfony vs Laravel: Battle of the PHP Frameworks,» Kinsta, [En línea]. Available: https://kinsta.com/blog/symfony-vs-laravel/. [Último acceso: 2025].

[66] «Symfony,» Symfony, [En línea]. Available: https://symfony.com/. [Último acceso: 2025].

[67] «Tailwind CSS: Rapidly build modern websites without ever leaving your HTML.,» [En línea]. Available: https://tailwindcss.com/. [Último acceso: 2025].

[68] «The Modern JavaScript Tutorial,» JavaScript Info, [En línea]. Available: https://javascript.info/. [Último acceso: 2025].

[69] «Timeline Chart,» Highcharts, [En línea]. Available: https://www.highcharts.com/docs/chart-and-series-types/timeline-series. [Último acceso: 2025].

[70] «TypeScript: JavaScript with syntax for types,» TypeScriptLang, [En línea]. Available: https://www.typescriptlang.org/. [Último acceso: 2025].

[71] «UKG Ready Scheduler,» UKG, [En línea]. Available: https://www.ukg.com/resources/product-info/ukg-ready-scheduler. [Último acceso: 2025].

[72] «UniTime Documentation,» UniTime, [En línea]. Available: https://help.unitime.org/. [Último acceso: 2025].

[73] «Vite: The Build Tool for the web,» Vite, [En línea]. Available: https://vite.dev/. [Último acceso: 2025].

[74] «Vue,» Vue.js, [En línea]. Available: https://vuejs.org/. [Último acceso: 2025].

[75] «WebAssembly,» WebAssembly, [En línea]. Available: https://webassembly.org/. [Último acceso: 2025].

[76] «Z3: An efficient SMT solver,» Microsoft, [En línea]. Available: https://www.microsoft.com/en-us/research/project/z3-3/. [Último acceso: 2025].

[77] A. Biryukov, D. Dinu y D. Khovratovich, «Argon2: New Generation of Memory-Hard Functions for Password Hashing and Other Applications,» de *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, Saarbrücken, Germany, 2016.

[78] A. Boyle, «Boost Productivity with UKG Ready AI Features,» UKG, Noviembre 2024. [En línea]. Available: https://www.ukg.com/working-smarter-cafe/boost-productivity-ukg-ready-ai-features. [Último acceso: 2025].

[79] C. Skanda, B. Srivatsa y B. S. Premananda, «Secure Hashing using BCrypt for Cryptographic Applications,» de *2022 IEEE North Karnataka Subsection Flagship International Conference (NKCon)*, Vijaypur, India, 2022.

[80] D. A. Kindy y A.-S. K. Pathan, «A Survey on SQL Injection: Vulnerabilities, Attacks, and Prevention Techniques,» de *2011 IEEE 15th International Symposium on Consumer Electronics (ISCE)*, Singapore, 2011.

[81] F. Glover, «Tabu Search — Part II,» *ORSA Journal on Computing,* vol. 2, nº 1, p. 4–32, 1990.

[82] F. Glover, «Tabu Search—Part I,» *ORSA Journal on Computing,* vol. 1, nº 3, p. 190–206, 1989.

[83] G. Allen y M. Owens, The Definitive Guide to SQLite, 2nd ed., Berkeley, CA: Apress, 2010.

[84] H. B. Yacoub, «Kinsta released 2021 PHP Benchmarks for Most Popular PHP Software,» 2021. [En línea]. Available: https://phpmagazine.net/2021/02/kinsta-released-2021-php-benchmarks-for-most-popular-php-software.html. [Último acceso: 2025].

[85] J. Blatz, «CSRF: Attack and Defense,» McAfee, Santa Clara, 2011.

[86] J. Ferraiolo, «Scalable Vector Graphics (SVG) 1.0 Specification,» 2001. [En línea]. Available: https://www.w3.org/TR/2001/REC-SVG-20010904/REC-SVG-20010904.pdf. [Último acceso: 2025].

[87] L. de Moura y N. Bjørner, «Z3: An Efficient SMT Solver,» de *14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2008)*, Budapest, Hungary, 2008.

[88] O. Kara y A. Atalay, «Preimages of Hash Functions through Rainbow Tables,» de *2009 24th International Symposium on Computer and Information Sciences (ISCIS)*, Guzelyurt, Northern Cyprus, 2009.

[89] P. De Ryck, L. Desmet, W. Joosen y F. Piessens, «Automatic and Precise Client-Side Protection against CSRF Attacks,» de *Computer Security – ESORICS 2011*, Berlin, Heidelberg, Springer, 2011, p. 100–115.

[90] S. Ceri, G. Gottlob y L. Tanca, «What You Always Wanted to Know About Datalog (And Never Dared to Ask),» *IEEE Transactions on Knowledge and Data Engineering,* vol. 1, nº 1, p. 146–166, 1989.

[91] S. Fogie, J. Grossman, R. Hansen, A. Rager y P. D. Petkov, XSS Attacks: Cross Site Scripting Exploits and Defense, Burlington, MA: Syngress, 2007.

[92] S. Gupta y B. B. Gupta, «Cross-Site Scripting (XSS) attacks and defense mechanisms: classification and state-of-the-art,» *International Journal of System Assurance Engineering and Management,* vol. 8, nº Suppl 1, p. 512–530, 2017.

[93] S. Mukherjee, P. Sen, S. Bora y C. Pradhan, «SQL Injection: A Sample Review,» de *6th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Dallas-Fort Worth, TX, USA, 2015.

[94] T. JPetsas, G. Tsirantonakis, E. Athanasopoulos y S. Ioannidis, «Two-factor authentication: is the world ready? quantifying 2FA adoption,» de *Proceedings of the Eighth European Workshop on System Security*, New York, NY, USA, 2015.

[95] T. Navarrete, «El lenguaje JavaScript,» Universitat Pompeu Fabra, Barcelona, 1999.

[96] U. O. d. Catalunya, «Bases de datos XML nativas,» [En línea]. Available: https://openaccess.uoc.edu/bitstream/10609/1064/1/00789tfc.pdf. [Último acceso: 2025].

[97] U. O. d. Catalunya, «El lenguaje SQL,» UOC, 2007. [En línea]. Available: https://www.dataprix.com/files/UOC_OpenSource_El_lenguaje_SQL.pdf. [Último acceso: 2025].

Anexos

Este capítulo contiene diversos apéndices que ofrecen información adicional sobre el proyecto TimeFlex. El Apéndice A describe las contribuciones personales de cada miembro del equipo, el Apéndice B contiene el Capítulo 1, el de introducción, traducido en inglés, el Apéndice C ofrece el Capítulo 8, el de conclusiones, y el Capítulo 9, el de trabajo futuro, traducidos en inglés, el Apéndice D detalla la política de privacidad de la aplicación, y por último, el Apéndice E muestra las preguntas de las encuestas empleadas en la fase de investigación (ver Capítulo 3).

Apéndice A - Contribuciones personales

Andrés Marí Piqueras

Al comienzo del proyecto, asumí la responsabilidad de definir el *stack* tecnológico, evaluando diferentes lenguajes, frameworks y arquitecturas. Para ello, llevé a cabo pruebas funcionales con tecnologías como Laravel, FastAPI o distintos sistemas de autenticación. Este proceso permitió descartar tecnologías y asegurar una base técnica sólida, capaz de escalar y adaptarse a futuras ampliaciones.

De forma paralela, participé activamente en la elaboración de encuestas y en la realización de entrevistas a trabajadores de distintos sectores laborales, con el fin de analizar las necesidades reales de los usuarios y poder definir los requisitos funcionales del sistema.

Con los datos recopilados, contribuí en el diseño de interfaz, tanto en su versión de baja como de alta fidelidad. Pudiendo ya distinguir las funcionalidades principales de la aplicación, y permitiendo anticipar posibles necesidades a tener en cuenta en el desarrollo.

Siguiendo estos diseños, iniciamos el desarrollo de la aplicación web. Comenzando en conjunto las páginas públicas visibles por cualquier usuario sin autenticar, como la página de bienvenida. Para luego continuar con módulos básicos del sistema, incluyendo, por ejemplo, la gestión de usuarios, roles, secciones o empresas.

Durante todo el proceso mantuvimos reuniones cada dos semanas con nuestro tutor, Pablo Gordillo, quien nos orientaba y nos proporcionaba sugerencias de mejora.

Con la estructura básica del sistema ya establecida, me encargué del desarrollo del microservicio de optimización y su integración con la aplicación principal. Inicialmente, construí de forma un servicio que rellenaba los horarios con datos ficticios para comprobar la conexión con la aplicación principal. De forma paralela, implementé el algoritmo de optimización utilizando datos de prueba generados de forma independiente, para posteriormente, integrar estos dos servicios, permitiendo generar horarios personalizados a partir de las respuestas de los usuarios.

Además, me coordiné con mis compañeros Gabriel y Álex para asegurar que el optimizador considerara correctamente los datos de entrada procedentes de los formularios y que los resultados, como vacaciones aceptadas o niveles de satisfacción, se reflejaran correctamente en los informes estadísticos de la aplicación.

Una vez finalizada la funcionalidad principal de generación de horarios, colaboré en el desarrollo del módulo de cambios de turno, trabajando junto a Álvaro. También construí el esqueleto del sistema de notificaciones, que mis compañeros desarrollaron más adelante, y me encargué de funcionalidades extra como el middleware de últimos accesos, el sistema de *logging* del microservicio y el modo *debug* del optimizador.

Finalmente, participé en la redacción de esta memoria junto al resto del equipo, dividiendo las secciones de forma equitativa y realizando una revisión colaborativa supervisada por Pablo Gordillo, con la finalidad de asegurar coherencia y calidad del documento.

En resumen, este trabajo ha sido desarrollado mayoritariamente de forma colaborativa, pero con tareas individuales claramente definidas, que requirieron una gran coordinación continua, alcanzando un desarrollo coherente y fluido de los distintos componentes.

Gabriel Fernández Sacristán

Al comienzo del proyecto, participé junto con mis compañeros en la elaboración del esquema general del proyecto y en el reparto de tareas y funcionalidades a

desarrollar. Al ser el encargado de realizar la implementación del módulo de formularios, me encargué de realizar la búsqueda de diferentes modelos de implementación para encontrar aquel que fuera más sencillo para desarrollar y que a su vez tuviera un grado de escalabilidad a la hora de introducir nuevas mejoras o funciones nuevas y dinámicas relacionadas con los formularios.

En cuanto a la fase previa del desarrollo, contribuí tanto a la construcción de encuestas definiendo las preguntas orientadas a los diferentes tipos de usuarios, como a la captación de personas objetivos que pudieran aportar su punto de vista sobre los diferentes módulos del proyecto.

A continuación, se inició la fase de diseño de la interfaz y la construcción de las funcionalidades esenciales para la ejecución del proyecto, entre las que destacan la gestión de usuarios y la gestión de secciones. Mi principal responsabilidad fue el desarrollo de la gestión de secciones, implementando las funcionalidades de alta, modificación y eliminación, así como el diseño de la interfaz que soporta dichos procesos. Durante esta etapa, se llevaron a cabo reuniones de seguimiento del proyecto cada dos semanas con nuestro tutor, Pablo Gordillo, quien nos orientó y sugirió posibles mejoras

Una vez finalizada la construcción de la interfaz y las funcionalidades esenciales, procedí a implementar el módulo de formularios en la aplicación. Para ello, se llevó a cabo una etapa de análisis en la que se investigaron diferentes modelos de implementación. Posteriormente, se desarrolló la fase de diseño, en la que se crearon las tablas correspondientes en la base de datos, definiendo sus relaciones. Finalmente, se llevó a cabo la fase de desarrollo, en la que escribí el código necesario para implementar las funcionalidades del módulo de formularios. Estas funcionalidades incluyen la creación, edición, duplicación de formularios, contestación, edición de respuestas, eliminación y visualización de respuestas.

Tras concluir la implementación del módulo formulario, colaboré con mi compañero Andrés en la integración de las respuestas de los formularios con la API de generación de turnos, seleccionando los campos necesarios para su correcto funcionamiento. Para ello se desarrolló un JSON de envío para que la API pueda

procesar correctamente los datos proporcionados y así completar la generación de horarios, teniendo en cuenta las solicitudes realizadas por los propios usuarios de la aplicación. Además, he colaborado con mi compañero Alejandro en la selección de los *KPIs* más importantes de los formularios para incluirlos en diferentes gráficos del módulo de estadísticas. También se añadió a la vista de 'Mi equipo' la posibilidad para gerentes de recursos humanos de exportar la información de todos los usuarios dados de alta en formato CSV.

Por último, participé en la redacción de esta memoria junto al resto del equipo, dividiendo las secciones de forma equitativa y realizando una revisión colaborativa supervisada por Pablo Gordillo, con la finalidad de asegurar coherencia y calidad en el documento.

En resumen, este trabajo ha sido desarrollado mayoritariamente de forma colaborativa, pero con tareas individuales claramente definidas, que requirieron una gran coordinación continua, alcanzando un desarrollo coherente y fluido de los distintos componentes.

Alejandro López López de la Cova

Desde el comienzo del proyecto, colaboré con mis compañeros en la asignación de tareas de forma equitativa y equilibrada en su desarrollo. Al ser el responsable principal del módulo de estadísticas, investigué las posibles herramientas a utilizar considerando Highcharts, Matplotlib, CanvasJS y Chart.js. Mediante este proceso, se pudo decidir la más adecuada para nuestro proyecto y justificar dicha elección.

Dentro de esta fase previa al desarrollo, contribuí a la fase de investigación definiendo las encuestas empleadas tanto para empleados como para el personal de RRHH. Además, participé en el análisis de las respuestas registradas, con el fin de identificar tendencias y patrones que permitieran definir los requisitos de la aplicación TimeFlex.

Esta fase dio paso a la etapa de diseño de interfaz, donde conjuntamente se definieron los prototipos de alta y baja fidelidad. Siguiendo los diseños construidos, se comenzó a desarrollar en conjunto la aplicación web con las vistas básicas de la aplicación como la página de "Ayuda" o "Sobre nosotros", y funcionalidades esenciales

como la gestión de roles o secciones. Durante este proceso, se mantuvieron reuniones semanales con nuestro tutor, Pablo Gordillo, para que nos orientara y ofreciera sugerencias en el desarrollo.

Una vez implementadas las funciones básicas de la aplicación, mi desarrollo se centró en los módulos de estadísticas y notificaciones, aunque también participé en el desarrollo global de otras funcionalidades de la aplicación como el suministro de datos al optimizador.

En cuanto al módulo de estadísticas, implementé diversas vistas con gráficos empleando Highcharts, así como los *scripts*, *endpoints*, funciones y *Controllers* necesarios para su correcto funcionamiento. Las vistas implementadas referentes a este módulo son:

(i) *Estadísticas mensuales por empleado*: 2 gráficos con información sobre la distribución de horarios y el número de horas trabajadas y esperadas por el empleado a lo largo del mes. Además, he implementado la tabla de horas esperadas que permite visualizar, editar y eliminar sus valores para cada empleado desde la interfaz de la aplicación.

(ii) *Panel de estadísticas de empleados*: 4 gráficos con métricas clave del empleado y su sección, comparando aspectos como la satisfacción o los turnos y horas trabajadas

(iii) *Dashboard del administrador*: 6 gráficos con *KPIs* y estadísticas relevantes de las diferentes secciones de la aplicación, permitiendo comparar su rendimiento y ofreciendo una visión global de la organización

(iv) *Panel de estadísticas de horario*: 2 gráficos por cada sección con mapas de calor y formato de calendario que muestran la demanda de turnos y vacaciones, además de un gráfico en formato de línea de tiempo que muestra los días de mayor y menor demanda

Con respecto al módulo de notificaciones, me he encargado de implementar el centro de notificaciones, vista donde se pueden observar todas las notificaciones del usuario y filtrar según el tipo. También he implementado la funcionalidad de

personalización de notificaciones y el modal asociado en el frontend, que permite a los usuarios elegir sobre qué tipos de notificaciones quieren recibir información.

Adicionalmente, he implementado las tablas de la base de datos relacionadas con las funcionalidades descritas a lo largo de esta sección, incluyendo las rutas necesarias en web.php empleadas a modo de *endpoints* y los *Controllers* requeridos para implementar las funcionalidades deseadas.

También he colaborado con mi compañero Gabriel en la selección de los atributos clave a incluir en los formularios, puesto que algunos de ellos eran necesarios para la correcta visualización de ciertos gráficos del módulo de estadísticas.

Por último, participé en la redacción de esta memoria junto al resto del equipo, dividiendo las secciones de forma equitativa y realizando una revisión colaborativa supervisada por Pablo Gordillo, con la finalidad de asegurar coherencia y calidad en el documento.

En resumen, este trabajo ha sido desarrollado mayoritariamente de forma colaborativa, pero con tareas individuales claramente definidas, que requirieron una gran coordinación continua, alcanzando un desarrollo coherente y fluido de los distintos componentes.

Álvaro Juan Martín Sánchez-Montañez

Desde el primer momento, mediante consenso grupal se llegó a la decisión de hacer una aplicación para la gestión de horarios. De esta forma, todo el equipo participó en la realización de un esquema general. Además, se llevó a cabo el reparto de tareas de forma equitativa y organizada para que todos los miembros nos sintiéramos parte del proyecto.

Una vez definido el *stack* tecnológico, y sabiendo que se iba a usar Laravel para el desarrollo de TimeFlex, me familiaricé con todas las funcionalidades básicas que ofrece este *framework* con el objetivo de comenzar la implementación del proyecto. Previo a esta fase de desarrollo, participé junto a mis compañeros en la elaboración y desarrollo de encuestas enfocadas a los distintos roles de la aplicación, participando también en el reclutamiento de posibles usuarios potenciales que pudieran responder

dichas encuestas. Junto a mis compañeros, formé parte del análisis de estas encuestas, lo que nos permitió identificar las necesidades de los distintos potenciales usuarios de la aplicación. Tras ello, colaboré en la fase de investigación sobre la competencia, en la que se analizó las fortalezas y debilidades de las distintas aplicaciones de gestión de horarios que hay en el mercado, con el objetivo de hacer una comparación con la solución propuesta. El último paso antes de comenzar con la implementación fue diseñar distintos prototipos de baja y alta fidelidad de las vistas que iba a ofrecer la aplicación. A pesar de que los diseños han ido cambiando a medida que se ha ido desarrollando la aplicación, estos prototipos sirvieron para tener una idea clara de lo que queríamos.

Una vez comenzó el desarrollo, se empezó a implementar el esqueleto de la aplicación. Así participé en el desarrollo de funcionalidades básicas como creación, edición, eliminación de usuarios y secciones, tanto para el rol de usuarios como el de gerente de RRHH. Durante estos primeros meses de desarrollo, acudí a las distintas reuniones periódicas convocadas por nuestro tutor, Pablo Gordillo. En estas reuniones cada integrante actualizaba sobre los avances conseguidos y los siguientes pasos a dar, planificando de esta manera el avance del proyecto. Adicionalmente, implementé la vista de "Mi Equipo", donde diseñé las distintas tarjetas para visualizar información del usuario e hice uso de middlewares para lograr que los administradores tengan acceso a información de todos los empleados, y los usuarios a información de compañeros de sección. Además implementé la funcionalidad de búsqueda utilizando las herramientas que ofrece Laravel.

Junto a mi compañero Andrés, me encargué de parte del diseño y desarrollo *frontend* de los calendarios que ofrece la aplicación web. Tras esto me encargué de la implementación de la funcionalidad de creación de turnos, junto a mi compañero Andrés. Junto a esto, implementé la clase de tipos de turnos para una generación más esquematizada y automatizada de turnos dentro de un horario. También he formado parte del desarrollo de la funcionalidad de cambios de turno. Relacionado con esta funcionalidad, me centré en diseñar la interfaz gráfica para la gestión de los turnos.

Durante la parte final del proyecto participé en la fase de pruebas, elaborando datos de prueba junto a todos mis compañeros. De esta fase surgieron pequeños

cambios tanto en *frontend* como en *backend*. que hubo que implementar, iterando hasta llegar a la versión final del proyecto.

Por último, participé en la redacción de esta memoria junto al resto del equipo, distribuyéndonos las secciones de forma equitativa y llevando a cabo una revisión conjunta supervisada por nuestro tutor, Pablo Gordillo, con el objetivo de asegurar coherencia y calidad del documento.

En resumen, este trabajo ha sido desarrollado mayoritariamente de forma colaborativa, pero con tareas individuales claramente definidas, que requirieron una gran coordinación continua, alcanzando un desarrollo coherente y fluido de los distintos componentes.

Apéndice B - Introduction

This chapter introduces and presents the TimeFlex web application project, which aims to optimize the management of shifts, schedules and vacation assignment in companies and organizations.

Section 1.1 describes the main motivation behind this project, as well as the issue to be addressed, Section 1.2 presents the main objectives of this project, Section 1.3 contains the work plan followed during its implementation and Section 1.4 describes the organization of the document.

Motivation

Shifts and schedules are key elements in many sectors such as education, healthcare and catering. Nevertheless, there is widespread dissatisfaction among employees due to the current inefficient methods for assigning and managing these shifts. In some cases, this assignment is even done manually on paper, in stark contrast to the current wave of technology and making it difficult to use objective metrics when generating these schedules.

In addition, Human Resources (HR) workers face a long list of constraints and preferences every time they try to set up a schedule. Trying to match many of them requires a great deal of time and resources, and there is no guarantee that the solution found will be the most satisfactory for employees. This can be frustrating not only for the employees in this department, but also for the rest of the company.

Interviews conducted with hospital and healthcare workers, as well as with teachers and employees in the hospitality sector, have served to verify the above facts and add new conclusions. On the one hand, there is a need for a shift assignment system that prioritizes employee satisfaction, based on as objective criteria as possible. On the other hand, workers require a technological application with which they can interact intuitively and perform actions that improve their work experience, such as requesting shift changes or holidays.

Goals

As a solution to the problems described above, the idea of the TimeFlex Project emerges, a web application through which HR employees can generate schedules, shifts and holidays assignment. The main objective is to develop a tool that contains different modules that allow both an efficient management of shift changes and other employee requests, as well as the generation of forms to carry out the requests made by employees. Moreover, this tool will include a statistics module to visually present the result of the assignments, as well as metrics grouped by different groups and sectors. It is also intended to offer functionalities associated with the different requests from the point of view of both the HR manager and normal users, thus managing the use cases of the different roles in order to offer a complete service to organizations. Finally, it is intended to offer a useful design, with a friendly and intuitive interface for the user.

To achieve this goal, the following steps were defined:

- Provide an automatic solution for the HR department when generating schedules and assigning work shifts within organizations.
- Definition of metrics and objectives to assess employee satisfaction.
- Ensure that the schedules and shifts generated maximize the satisfaction of the employees involved.
- Design an intuitive and interactive interface that offers a good user experience.
- Implement functionalities in order to manage tasks related to work shifts and time off. These include requesting shift changes, viewing and interacting with statistics and dashboards, and generating forms for employee feedback.
- Provide a notification system to communicate changes and management with customization options available for the user.

Work plan

The work plan followed for the development of the TimeFlex Project consists of six chronologically ordered phases:

Phase 1 – Study and Analysis of the Needs

By conducting personal interviews and questionnaires with workers in the sectors of interest, the main needs and issues of the employees were identified. In this way, the goals of the application, described in the previous section, were defined.

Phase 2 – Design of the system architecture

Taking into account the requirements gathered in the previous phase, the most appropriate tools and architectures for the development of the project were studied. This included the design of the interface, the database structure and the different layers of the application so as to meet the proposed objectives. In addition, software products with similar approaches to TimeFlex were studied.

Phase 3 – Development of the frontend and backend

In this phase, the implementation of the basic functionalities of the application such as login, role management or the notification system was carried out. Moreover, the interface was designed and most of the advanced functionalities were coded, such as the creation of forms, statistics generation or shift change request, with appropriate logic for the transcription of the information to the database.

Phase 4 – Shifts optimization using API

The incorporation of a constraint solving system specialized in shift and schedule assignment was carried out in this stage. Using an API and tools such as the Z3 constraint solver [76] [87], developed by Microsoft, the assignment of both work and vacation shifts was automated.

Phase 5 – Validation and testing

Tests were carried out on the different use cases and implemented functionalities, ensuring their correct operation and guaranteeing a correct user experience.

Phase 6 - Documentation and final adjustments

The documentation associated to the web application was prepared at this stage, detailing all relevant information about the project.

Phases 3, 4 y 5 described previously followed an iterative process, in which the feedback obtained was incorporated progressively to build the different versions and prototypes of the application.

The final result obtained after following these steps is a web application that meets the initially defined requirements. A tool has been developed that efficiently automates both shift change requests and the assignment of schedules and holidays, offering customized interfaces for HR managers and normal users, adapted to their different use cases. Emphasis has been placed on making the interface intuitive and user-friendly.

Organization of the document

This document is organized as follows: Chapter 2 describes the main functionalities offered by TimeFlex, going deeper into target users and conducting a study of the competition. Chapter 3 presents the research carried out with users through interviews and surveys as well as the results obtained. It also characterizes the users as well as their use scenarios and their journey in the application. Chapter 4 details the architecture used in the Project as well as the different layers involved. In addition, it contains the structure of the database used. Chapter 5 contains detailed information about the different functionalities of each of the different modules in the application. Chapter 6 describes the technical implementation and tools used in both the application modules and the optimization service. Additionally, it includes security management and technical aspects related to the execution environment, usability and performance of the application. Chapter 7 contains the steps to follow for the correct installation of the TimeFlex Project and the user manuals. Finally, Chapter 8 contains conclusions and Chapter 9 future lines of work.

Apéndice C - Conclusions and Future Work

Conclusions

The development of this web application has involved a phase of design, research, implementation, optimization and validation.

One of the main challenges when developing has been the integration heterogeneous technologies such as Python and PHP. Taking into consideration the possibility of needing to modify libraries or even change languages during the process, a microservices-based architecture was chosen. This decision made it possible to decouple the different parts of the system, which facilitated the parallel development of independent functionalities, thus speeding up the implementation process.

In addition, this architecture takes advantage of the specific strengths of each environment: Laravel, oriented to the management of business logic (users, roles, forms, etc.), and Python, which offers access to specialized libraries such as the Z3 solver.

On the other hand, the microservices-based architecture also prepares the system for future scalability, allowing the integration of new modules or external services without compromising the stability or the functional core of the application.

The parallel development of the different functionalities, even in different technological environments, made the generation of test data essential. This data not only facilitated the development of new functionalities, but also made it easier to verify that the already implemented functionalities continued to work correctly after each new integration.

In a distributed environment such as this project, having a consistent and automated set of test data was fundamental to ensure consistency, reliability and integrity throughout the development cycle.

On the other hand, the microservices-based web application carries additional risks in terms of security. By dividing the system into multiple components with different technologies, the exposure surface and thus the number of possible attacks increases. This risk requires a correct configuration of the communication channels, as well as careful management of access and authentication controls. From the point of view of the framework used, Laravel offers a solid base in terms of security (see Section 6.3) that has facilitated this task.

As for the data handled by the application, it mainly handles personal information such as names, email addresses, shifts and preferences of workers. Although data related to shifts or forms may seem harmless in a first analysis, combined with personal data they may allow the indirect identification of users. This makes them protected personal data under the General Data Protection Regulation (GDPR), which makes it necessary to implement specific protection and transparency measures.

With regard to privacy, the app includes a privacy policy that informs about data use and storage, as well as rights in relation to this data. In addition, data minimization practices have been followed, avoiding storing unnecessary information.

One of the key aspects of the developed system is its focus on user-centered design, which, from the beginning of the development, has guided both the design of the interface and the definition of functionalities. This orientation has made it possible to adapt the application to the needs of the different profiles involved in schedules management.

A usability and interface study in early stages of development has facilitated the identification of usage patterns, needs and friction points. This approach has facilitated the development of an intuitive, clear and accessible interface for all the roles involved.

Moreover, the automation of certain processes, such as the validation of constraints, the resolution of overlaps or the detection of unbalanced days, helps to minimize common human mistakes in manual planning.

Unlike other more generic or rigid schedule management tools, the proposed application stands out not only for offering a solution with a reduced learning curve

thanks to its clear interface, but also for offering a personalized, flexible and employee welfare-oriented planning, allowing a fairer and more humane planning.

Future lines of work

Although the developed application meets the goals set and offers a robust and competitive solution, there are several areas for improvement that could enhance diverse aspects such as the user experience or internationalization.

A clear line of improvement is found in the extension of the set of constraints that the optimizer handles. In order to achieve an even more precise and personalized planning, it would be convenient to add new restrictions, both optional and mandatory, which would also require new questions in the creation of forms.

In this sense, it is relevant to incorporate the option of adding constraints that ensure strict compliance with Spanish labor legislation, such as limits on maximum working hours, mandatory breaks, and regulations covering rotating or night shifts. In addition, thanks to having implemented it as a microservice, it is possible to integrate different resolution techniques and algorithms. In this line, it is possible to study the efficiency of the proposed technique and implement a hybrid solution that combines a voracious algorithm with the solver.

In parallel, and considering an increasingly globalized environment with geographically distributed teams, it is indispensable that the application supports the generation and visualization of schedules for workers in different time zones. This requires adapting the optimizer logic and interface to correctly manage time differences, guaranteeing that the assigned shifts respect the local hours of each employee. In order to do so, the representation of turns and time zones could be modified in a manner that backend works with an *AoE* representation, while the frontend particularizes the hours to the time region from which the application is accessed.

In line with the support of multiple time zones, internationalization is a key strategy to increase the accessibility and commercial reach of the application. Implementing a multilingual application implies designing a translation system which allows adapting the interface, forms and notifications to different languages in a maintainable way.

From a technical point of view, the system currently uses SQLite [62], a lightweight database ideal for development environments. However, for a production environment with high loads and a larger user community, it would be necessary to migrate to a more robust and scalable database system such as PostgreSQL [53] or MySQL [44]. This change would not require great efforts due to Laravel's code abstraction, and would help to better manage concurrency and global trust in the application.

Due to the increasing use of mobile devices in the work environment, the development of a mobile application would be an indispensable improvement, as it would facilitate accessibility and usability from anywhere. Furthermore, a mobile app can integrate additional functionalities such as push notifications or two-step verification methods (2FA [94]), increasing security and interaction with users.

Regarding the notification system, the current version of the application allows for the transmission of notifications from its interface. Nevertheless, it would be convenient to have alternative ways to send notifications to users. A recurring method for this is sending emails, facilitated by SMTP [61] , a standard protocol for sending emails. Laravel incorporates native support for this protocol, already mentioned in Section 4.1, which would allow its integration in a simple way. In this way, every time a shift change or help were requested to Administration, the users involved could receive notifications both by the application and by email. Moreover, this would provide optionality in terms of configuration, as it would lead to implement an even greater customization of the notification system by allowing users to choose the notification channels they want. In terms of implementation, tools already present in Laravel such as the queuing system (Queues [57]) or notifications (Notifications [45]) could be used to send emails in an asynchronous way. This would also allow to integrate sending emails through corporate services such as Outlook [51], widely used by enterprises, just by configuring parameters in the .env file. This integration would further professionalize the tool and enhance the communication system with users.

Another line of improvement for future versions would consist in the integration of external platforms widely used in work environments, such as Microsoft Teams [41] or Slack [60]. Specifically, it is possible to connect to their public APIs, allowing automatic

notifications to be sent to users, creating events in the employee's personal calendar directly from TimeFlex, synchronizing absences for planning or incorporating bots that allow interaction with TimeFlex directly from the corporate chat. These integrations would not only enhance the user experience by allowing users to access information without having to change platforms, but would also allow to automate processes and improve internal communication.

Finally, the incorporation of chatbots would be a significant innovation for user interaction. A chatbot could assist in the generation and filling of forms, answer frequent questions and facilitate the view and modification of schedules in a conversational manner. This tool would not only improve the user experience, but also streamline the process and reduce the learning curve, turning the developed application into an even more complete and modern solution.

Apéndice D - Política de privacidad

TimeFlex dispone de una política de privacidad propia, que los usuarios deben leer y aceptar antes de enviar un formulario de ayuda o contacto a Administración. En dicho formulario, los usuarios incluyen datos personales para que se pueda evaluar correctamente su situación y posteriormente establecer contacto con ellos suministrando la ayuda e información que necesitan.

El documento de política de privacidad de TimeFlex se ha elaborado utilizando la herramienta "Facilita RGPD" de la AEPD (Agencia Española de Protección de Datos) [20]. Esta herramienta brinda asistencia a las empresas suministrando un formulario para establecer el riesgo de los tratamientos realizados en base a cuestiones como el sector al que pertenece la actividad de la organización o el tipo de datos tratados. Tras rellenar dicho cuestionario, se ha concluido que el riesgo para los derechos y libertades de los interesados en el tratamiento de datos de TimeFlex es escaso.

En el documento de política de privacidad de TimeFlex, se han suprimidos ciertos datos como el NIF de la persona responsable del tratamiento o el teléfono asociado a la empresa debido a que se trata de una empresa ficticia. El propósito del documento es proporcionar una política de privacidad lo más cercana posible a la realidad en caso de que la actividad comercial de TimeFlex se llevara a cabo. Este documento servirá de referencia a los usuarios para conocer sus derechos y libertades y otras cuestiones importantes como la finalidad del tratamiento.

Apéndice E - Encuestas empleadas en la fase de investigación

Encuesta para Empleados

1. ¿Qué esperas de una aplicación que gestione tus preferencias de horarios de trabajo?
    - Respuesta a desarrollar
2. ¿Cuál sería tu inquietud principal acerca de la misma?
    - Respuesta a desarrollar
3. ¿Has usado antes alguna aplicación para optimizar turnos?
    - Sí
    - No
4. En caso de haber utilizado alguna previamente, ¿qué características te gustaron y cuáles no?
    - Respuesta a desarrollar
5. ¿Cómo te gustaría introducir tus preferencias de horarios?
    - Formularios rápidos
    - Arrastrando y soltando dentro de la aplicación
    - Chatbot
    - Otra (respuesta a desarrollar)
6. ¿De qué manera prefieres recibir notificaciones acerca de nuevos horarios o cambios de turno?
    - Email
    - Notificaciones push de la aplicación en el dispositivo

- Ambas
- Otra (respuesta a desarrollar)

7. ¿Con qué frecuencia querrías recibir dichas notificaciones?
    - De forma diaria
    - 2-3 veces por semana
    - De forma semanal
    - De forma mensual
    - Otra (respuesta a desarrollar)
8. ¿Qué preferencias te gustaría que se tuvieran en cuenta?
    - Respuesta a desarrollar
9. ¿Prefieres que tus respuestas sean privadas o públicas?
    - Privadas
    - Públicas
    - No tengo preferencia
10. ¿Cómo prefieres que se presenten tus horarios?
    - Calendario
    - Lista de turnos
    - Línea de tiempo
    - Otra (respuesta a desarrollar)
11. ¿Cómo de importante es para ti poder ver todo tu horario de trabajo del mes de un vistazo?
    - Mucho
    - Poco
    - Nada
    - Otra (respuesta a desarrollar)

12. ¿Te gustaría una forma rápida de dar retroalimentación sobre los turnos que te asignan?
    - Sí
    - No
    - Tal vez
    - Otra (respuesta a desarrollar)
13. ¿Cómo te gustaría interactuar con esa funcionalidad?
    - Respuesta a desarrollar
14. ¿Cómo de importante es para ti poder comparar tu horario con el de otros compañeros de equipo?
    - Mucho
    - Poco
    - Nada
    - Otra (respuesta a desarrollar)
15. ¿Cómo te gustaría solicitar un intercambio de turnos?
    - Lista de compañeros con sus turnos
    - Solicitud automática
    - Otra (respuesta a desarrollar)
16. ¿Qué pasos considerarías ideales para gestionar las solicitudes de cambio?
    - Respuesta a desarrollar
17. ¿Cómo preferirías que fuera la aplicación?
    - Diseño minimalista
    - Diseño más detallado y completo
    - Otra (respuesta a desarrollar)

18. ¿Te gustaría poder personalizar la interfaz de usuario? Por ejemplo, poder elegir qué información ver primero o los colores del tema
    - Sí
    - No
    - Otra (respuesta a desarrollar)
19. ¿Prefieres una aplicación con menos opciones pero más fácil de usar o una más completa y con más funcionalidades?
    - Menos opciones pero más fácil de usar
    - Más completa y con más funcionalidades
    - No tengo preferencia
20. ¿Cuál crees que es el equilibrio adecuado entre simplicidad y funcionalidades avanzadas para ti?
    - Respuesta a desarrollar
21. ¿Qué tipo de asistencia o tutoriales te gustaría ver dentro de la aplicación para ayudarte a familiarizarte con sus funcionalidades?
    - Respuesta a desarrollar
22. ¿Te gustaría tener una función de chatbot que te guíe a través del proceso o que te ayude con preguntas frecuentes?
    - Sí
    - No
    - Otra (respuesta a desarrollar)
23. ¿Con qué dispositivos prefieres acceder a este tipo de aplicaciones?
    - Móvil
    - Tablet
    - Escritorio
    - Otra (respuesta a desarrollar)

24. ¿Te gustaría poder modificar tu disponibilidad o tus preferencias desde cualquier dispositivo?
    - Sí
    - No
    - Otra (respuesta a desarrollar)
25. ¿Cuál sería tu experiencia ideal en términos de facilidad y rapidez?
    - Respuesta a desarrollar

Encuesta para encargados de RRHH

1. ¿Cuáles son los mayores desafíos a los que te enfrentas actualmente al cuadrar los horarios de los empleados?
    - Respuesta a desarrollar

¿Cuánto tardas en dicho proceso?

- Respuesta a desarrollar

2. ¿Cómo sueles recopilar las preferencias de los empleados?
    - Respuesta a desarrollar
3. ¿Te gustaría que el proceso fuese más automatizado?
    - Sí
    - No
    - Tal vez
4. ¿Sientes que tienes en cuenta las preferencias de los empleados?
    - Sí
    - No
    - No lo sé
5. ¿Crees que tienes en cuenta dichas preferencias de forma equitativa?

- Sí
- No
- No lo sé

6. ¿Qué herramientas o vistas te serían más útiles para gestionar de forma eficiente los turnos?
    - Dashboards
    - Gráficos de satisfacción
    - Mapas de calor de disponibilidad
    - Otra (respuesta a desarrollar)
7. ¿Te gustaría poder simular diferentes escenarios de horarios y visualizar el impacto en la satisfacción y productividad de los empleados?
    - Sí
    - No
    - Tal vez
8. ¿Te gustaría que la aplicación te ofreciera análisis sobre cuándo podrías necesitar contratar más personal o ajustar los horarios?
    - Sí
    - No
    - Tal vez
9. ¿Qué métricas crees que serían clave para esta función?
    - Respuesta a desarrollar
10. ¿Qué preferencias te suelen pedir los trabajadores?
    - Respuesta a desarrollar
11. ¿Qué inconvenientes son los más comunes a los que te enfrentas?
    - Respuesta a desarrollar

12. ¿Cómo solucionas dichos inconvenientes?
    - Respuesta a desarrollar
13. ¿Cómo prefieres recibir reportes sobre los horarios y la satisfacción de los empleados?
    - Gráficos
    - Reportes semanales
    - Alertas
    - Otra (respuesta a desarrollar)
14. ¿Te gustaría tener una función que permita a los trabajadores de diferentes áreas colaborar en la planificación de turnos, compartiendo información sobre disponibilidad y necesidades?
    - Sí
    - No
    - Tal vez
15. ¿Cómo prefieres gestionar y priorizar las solicitudes de cambio de turno de los empleados?
    - Respuesta a desarrollar
16. ¿Te gustaría automatizar parte de este proceso?
    - Sí
    - No
    - Tal vez
17. ¿Qué tipo de informes y gráficos te gustaría que la aplicación generara automáticamente sobre la asignación de horarios?
    - Gráficos de satisfacción
    - Balance de turnos entre empleados
    - Otra (respuesta a desarrollar)

18. ¿Te gustaría que la aplicación te alertara automáticamente si hubiera empleados con turnos repetidamente desfavorables o si detecta un desequilibrio en las preferencias?
    - Sí
    - No
    - Tal vez
19. ¿Te supondría un inconveniente que las respuestas de los trabajadores sean secretas?
    - Sí
    - No
20. ¿Te parecería bien mostrar a los empleados que algunos tienen preferencia por motivos especiales (embarazadas, personas mayores, personas con depresión, ...)?
    - Sí
    - No
21. ¿Qué nivel de control te gustaría tener sobre quién puede acceder a las preferencias y horarios de los empleados?
    - Respuesta a desarrollar
22. ¿Te gustaría poder personalizar estos permisos?
    - Sí
    - No
    - Tal vez
23. ¿Cómo preferirías que fuera la aplicación?
    - Diseño minimalista
    - Diseño más detallado y completo
    - Otra (respuesta a desarrollar)

24. ¿Te gustaría poder personalizar la interfaz de usuario? Por ejemplo, poder elegir qué información ver primero o los colores del tema
    - Sí
    - No
    - Otra (respuesta a desarrollar)
25. ¿Prefieres una aplicación con menos opciones pero más fácil de usar o una más completa y con más funcionalidades?
    - Menos opciones pero más fácil de usar
    - Más completa y con más funcionalidades
    - No tengo preferencia
26. ¿Cuál crees que es el equilibrio adecuado entre simplicidad y funcionalidades avanzadas para ti?
    - Respuesta a desarrollar
27. ¿Crees que facilitaría el desarrollo de tus tareas que la aplicación te proporcionara un formulario por defecto con posibilidad de ser modificado para obtener las respuestas de los empleados o preferirías crearlos desde cero?
    - Formulario por defecto
    - Crear el formulario desde cero
    - Otra (respuesta a desarrollar)
28. ¿Qué tipo de asistencia o tutoriales te gustaría ver dentro de la aplicación para ayudarte a familiarizarte con sus funcionalidades?
    - Respuesta a desarrollar
29. ¿Te gustaría tener una función de chatbot que te guíe a través del proceso o que te ayude con preguntas frecuentes?
    - Sí
    - No

- Otra (respuesta a desarrollar)

30. ¿Con qué dispositivos prefieres acceder a este tipo de aplicaciones?
    - Móvil
    - Tablet
    - Escritorio
    - Otra (respuesta a desarrollar)
31. ¿Te gustaría poder modificar tu disponibilidad o tus preferencias desde cualquier dispositivo?
    - Sí
    - No
    - Otra (respuesta a desarrollar)
32. ¿Cuál sería tu experiencia ideal en términos de facilidad y rapidez?
    - Respuesta a desarrollar