DASHBOARD PARA JUECES EN LÍNEA
A DASHBOARD FOR ONLINE JUDGES

ALEXANDRA PÉREZ BERMEJO

GRADO EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

TRABAJO FIN DE GRADO

JUNIO 2020

DIRECTORES:
GUILLERMO JIMÉNEZ DÍAZ
PEDRO PABLO GÓMEZ MARTÍN

RESUMEN

Los paneles de control o "*Dashboards*" son un tipo de interfaz gráfica de usuario que muestran de manera visual los indicadores clave de desempeño (KPI*, key performance indicators*) relevantes a un objetivo particular o un proceso de negocio.

El presente proyecto consiste en la elaboración de un dashboard para el juez en línea *¡Acepta el reto!* tanto para sus administradores como para los profesores que hacen uso de él. El diseño se ha realizado siguiendo la metodología de Diseño Guiado por Objetivos (DGO) y se ha implementado una aplicación web modular, lo cual permite que cada gráfica pueda ser añadida en diferentes lugares del portal.

Palabras clave: DGO, dashboard, jueces en línea, base de datos, gráficos, MVC

ABSTRACT

Dashboards are a type of graphical user's interface, which shows, in a visual way, the relevant key performance indicator (KPI) to a particular purpose or a business process.

The present project consists of developing a dashboard for "*¡Acepta el reto!*" an online judge, to be used for both, administrators and teachers. It has been designed following a Goal-Directed Design process and a modular website app has been implemented, which allows each graphic to be added to different site locations.

Key words: DGO, dashboard, online judges, database, graphics, MVC

Índice de contenidos

Índice de figuras

Índice de tablas

AGRADECIMIENTOS

Quería dar las gracias en primer lugar a mis tutores: Guillermo y Pedro Pablo por tener tanta paciencia conmigo, ayudarme y darme consejos en la realización de esta memoria: "Más vale tarde que nunca".

En segundo lugar, gracias a mi familia y a mis amigos por estar ahí siempre de forma incondicional y apoyarme en todo momento a realizar este proyecto.

Y también a todos los profesores que han dedicado parte de su tiempo en responder a las entrevistas iniciales para la realización de este proyecto y las evaluaciones finales.

Capítulo 1 - Introducción

Hoy en día, los jueces en línea son muy habituales cuando se está empezando a programar ya que es una manera sencilla de resolver problemas de una forma individual siendo el propio juez el que corrige la solución del problema resuelto.

Estos jueces en línea disponen de gran cantidad de datos sobre las interacciones de los usuarios con los problemas que intentan, pero muchos de ellos, no disponen de ningún medio para extraer toda la información que pueda ser relevante para los diferentes roles que puedan llegar a usar esos jueces en línea. En particular, los paneles de control son una de las carencias de *¡Acepta el reto!*, un juez online desarrollado en la Universidad Complutense de Madrid.

En este proyecto, vamos a construir un panel de control que nos muestre los datos más relevantes para los administradores y los profesores que usen *¡Acepta el reto!* de una forma organizada, es decir, va a permitir obtener datos a los usuarios sin necesidad de realizar una búsqueda exhaustiva en toda la aplicación.

1.1 Antecedentes

¡Acepta el reto! se creó en 2014 y en la actualidad, contiene más de 400 problemas de programación en español con diferente complejidad (Gómez-Martín & Gómez-Martín). Este juez, se creó con el propósito de que los alumnos de la Universidad Complutense de Madrid, principalmente, pudiesen resolver problemas y un juez en línea se los verificase, de esa manera, el usuario podía establecer un ámbito de aprendizaje por sí mismo e ir viendo su evolución. Este nuevo sistema de aprendizaje también formaría parte del método de enseñanza y los profesores se podrían beneficiar de él para recopilar datos sobre sus alumnos.

Hasta el momento, tanto los profesores como los administradores se habían creado sus propios sistemas para visualizar los datos que creían más relevantes para ellos, pero no se había creado ningún panel de control como por ejemplo si lo tienen otros muchos jueces.

1.2 Objetivo

El objetivo principal de este proyecto consiste en crear una aplicación para que se puedan visualizar a partir de unos paneles de control los datos más relevantes, de una forma aislada, del juez en línea de *¡Acepta el reto!* (Acepta el reto). Para ello, se ha realizado un estudio sobre las necesidades de un determinado grupo de usuarios que suelen visualizar datos obtenidos de este juez en línea.

Para crear esta aplicación, vamos a seguir una serie de subobjetivos para llegar al objetivo final:

1. Investigar qué son los jueces en línea y cuáles existen.
2. Investigar qué son los paneles de control y cómo se construyen.
3. Entrevistar a los dos tipos de usuarios de *¡Acepta el reto!:* administradores y profesores.
4. Analizar a la competencia que tiene *¡Acepta el reto!*
5. Obtener los requisitos funcionales de los administradores y de los profesores.
6. Diseñar la aplicación en base a las necesidades obtenidas en las entrevistas.
7. Estudiar las herramientas de trabajo para la implementación de la aplicación.
8. Implementar la aplicación: Dashboard para jueces en línea, en base al diseño.
9. Evaluar la aplicación creada con los usuarios.

Cada uno de estos pasos, se explicarán en los respectivos capítulos de una forma más detallada y específica.

1.3 Plan de trabajo

Con el objetivo principal ya fijado y los subobjetivos definidos, nos toca establecer el plan de trabajo que vamos a seguir.

Primeramente, investigaremos qué son los jueces en línea y cuáles existen en la actualidad. También investigaremos en qué consisten los paneles de control y cómo se construyen. Toda esta investigación, formará parte del Capítulo 2 - Jueces en línea y dashboards.

Después continuaremos con el Capítulo 3 - Diseño, dónde nos centraremos en explicar que es el Diseño Guiado por Objetivos (DGO), analizaremos a la competencia que existe de jueces en línea y entrevistaremos a los futuros usuarios de nuestra aplicación para saber sus necesidades y en base a los requisitos funcionales, diseñaremos la aplicación en papel para más tarde implementarla.

En el Capítulo 4 - Dashboard para jueces en línea, explicaremos los elementos que forman parte de nuestra aplicación y veremos cómo está organizada. Todo esto estará explicado desde la parte visual, pero, continuando con el Capítulo 5 - Arquitectura e implementación llegaremos a la explicación de la arquitectura cliente-servidor que es la que se ha utilizado en nuestra aplicación junto con el Modelo Vista Controlador (MVC). En este capítulo explicaremos cómo hemos ido desarrollando la parte del servidor y la parte del cliente para finalmente llegar a obtener nuestra aplicación web[1].

Una vez que nuestra aplicación estaba implementada y es accesible sin ningún problema, se llevaría a cabo la evaluación con los usuarios que se encuentra en el Capítulo 6 - Evaluación con usuarios.

La memoria termina con el Capítulo 7 - Conclusiones y trabajo futuro correspondiente a las conclusiones obtenidas y al trabajo futuro que se podría realizar sobre lo que ya se ha implementado.

[1] https://onlinejudgedashboard.es/

1.4 Introduction

Nowadays, online judges are a very popular tool used for programming, as it is an easy choice to resolve problems in an individual way being the online judge in its own who corrects the solution of the problem solved.

These online judges have a great amount of user's interaction data with the problems they try, but a good number of them, do not provide any mode to get all the relevant information for the different roles can use these online judges. That is the reason why we are going to create a dashboard, which will show the most relevant data in an organised way to the teachers and administrators who uses "*¡Acepta el reto!*" in other words, it will allow the users to collect data with no need of an exhaustive search across the application.

1.5 Background

"*¡Acepta el reto!*" was created in 2014, and it currently contains more than 400 Spanish programming problems of different complexity (Gómez-Martín & Gómez-Martín). This judge was created, mainly with the propose of the Complutense University of Madrid´s students could resolve the problems and an online judge could verify them, that way the user could develop a self-taught learning and evaluation.

This new learning system will also be part of a teaching system and the teachers could benefit from the data about their students.

Until today, both, teachers and administrators have created their own systems to visualise the most relevant data, but there has not been created any dashboard like other judges have had.

1.6 Objectives

In this project the main goal is to create an app to visualise in a dashboard the most relevant data, in isolation, for the online judge "*¡Acepta el reto!*" (Acepta el reto). For that purpose, there has been a research about the main needs of a specific user's group who generally use this online judge.

To develop this app, we will follow a series of steps:

1. Research what are online judges, and which one exist already.
2. Research what are dashboards and how to create them
3. Interview the two types of users of "*¡Acepta el reto!*": administrators and teachers.
4. Analysed "*¡Acepta el reto!*" competences
5. Obtain the administrators and teachers functional requirements.
6. Design the app based on the needs from the interviews.
7. Study the working tools to implement the app.
8. Implement the app: Dashboard for online judges, based on the design.
9. Assess with the users the app designed.

Each of the steps above will be explained in a more specific and detailed way through the different chapters.

1.7 Working plan

Once we have the objectives and sub-objectives, the working plan will be based on the following:

Firstly, we will research about what are online judges and which ones already exist. We will also research about what dashboards are and how to create them. All this information will be collected in Chapter 2 - Online judges and dashboards (*"Capítulo 2 - Jueces en línea y dashboards"*).

Afterwards, *Chapter 3 - Design ("Capítulo 3* - Diseño"*),* we will focused in the explanation of what is a Goal-Directed Design, we will analysed the online judges competencies and will interview future users to know what their needs are, afterwards based on the functional requirements we will design the app on paper to implement it later.

In Chapter 4 – Dashboard for Online judges ("*Capítulo 4* - Dashboard para jueces en línea"*),* we will explained all the elements as part of our app and will check how it is organised. All of this, it will be explained from the visual part, but, following with Chapter 5 – Architecture and implementation ("*Capítulo 5* - Arquitectura e implementación*)"* the client-server architecture that has been used in the app within the Model View Controller (MVC) will be explained. In this chapter we will explain how the server's and the client's part has been developed to obtain our web application[2].

Once our application is implemented and accessible without any problem, the user's evaluation is done which it is found in Chapter 6 – User's evaluation ("Capítulo 6 - Evaluación con usuarios"*)*.

This report will be finalised with Chapter 7 – Conclusions and future work (*"Capítulo 7* - Conclusiones y trabajo futuro"*),* where it is explained the conclusions after the research and the future work that could be done over the work already created.

[2] https://onlinejudgedashboard.es/

Capítulo 2 - Jueces en línea y dashboards

El objetivo principal de este proyecto consiste en crear un panel de control para jueces en línea.

Primeramente, se va a hacer un análisis del funcionamiento de un juez en línea y ver cuales están presentes en la actualidad centrándonos en *¡Acepta el reto!*, ya que es el juez en el que nos vamos a basar para realizar nuestro proyecto.

Después, se estudiarán los paneles de control o *dashboards* para saber qué son, cómo se usan, cuáles son los más adecuados para mostrar los diferentes tipos de información, cuáles son los principales errores que se comenten con los *dashboards*, etc.

2.1 Jueces en línea

Un juez en línea (*online judge*) en el ámbito de la programación es un sistema que contiene un conjunto de problemas, los cuales, pueden ser de estructuras de datos, algoritmos y programación y normalmente están categorizados por dificultades y temáticas. Estos problemas, disponen de un enunciado donde se explica en qué consiste el ejercicio y la salida que se debe obtener junto con algunos ejemplos para que se comprenda mejor. Los usuarios, por tanto, eligen cual quieren realizar según sus intereses y/o capacidades, y una vez resuelto, se lo envían al juez que será su evaluador.

Un juez, dispone de una serie de casos de prueba que usará para testear las soluciones enviadas por los usuarios. Al recibir una solución, el juez compilará y ejecutará el código recibido sobre esos casos. Una vez ejecutado ese código, comparará las salidas con las de la solución correcta de la que dispone. Si las salidas son idénticas, el veredicto será ***Accepted (AC)***, mientras que, si son diferentes, el

veredicto será incorrecto y podrá derivar en varios tipos de errores. Estos son los más comunes:

- ***Presentation error (PE)***: la solución es correcta, pero la salida no sigue exactamente el formato de la salida de texto. Esto se debe a que hay algún salto de línea, espacio o tabulado diferente a la salida esperada por el juez.
- ***Wrong Answer (WA)***: la solución es incorrecta. Los casos de prueba no se han superado, por lo que la resolución no es adecuada.
- ***Compile Error (CE)***: el juez no puede compilar el código por lo que existen errores de sintaxis y por tanto no puede ejecutarse.
- ***Run-Time Error (RTE)***: el juez ha compilado la solución, pero la ejecución ha acabado de manera indeseada y no ha llegado a procesar los casos de prueba.
- ***Time Limit Exceeded (TLE)***: el juez ha compilado y ha ejecutado el código, pero, ha tardado en dar una respuesta y eso ha derivado en la cancelación de la ejecución.
- ***Memory Limit Exceeded (MLE)***: se ha superado el límite de memoria máximo que se permite usar al código, por lo que la ejecución se ha cancelado.
- ***Output Limit Exceeded (OLE)***: el código se ha compilado y ejecutado de forma correcta, pero, la salida es más larga que la permitida.

Una vez el juez ha testeado el código, el usuario recibirá el veredicto obtenido en su ejercicio y el usuario podrá rehacer el ejercicio y volverlo a enviar al juez las veces que quiera teniendo o no un ***AC*** como veredicto. Hay veces que ese número de intentos puede estar limitado dependiendo del juez que evalúe o si se trata de un concurso o de una prueba.

A través de los jueces en línea existentes en la actualidad, los profesores pueden "retar" a sus alumnos a realizar una serie de problemas de forma entretenida, ya que los usuarios tienen que poner en marcha sus habilidades como programadores a la hora de resolver algoritmos.

La inclusión de estos sistemas en el aprendizaje sobre todo en el ámbito universitario o de formación profesional ayuda tanto a profesores como a alumnos y les proporciona una serie de ventajas:

- **Flexibilidad de resolución**: en cualquier momento se puede acceder a ellos.
- **Veredictos inmediatos**: el usuario que resuelve el problema obtiene un veredicto a partir del cual sabe si su resolución es correcta o no.
- **Reducción del trabajo a los profesores**: al ser el propio juez el que da el veredicto, el alumno es autónomo y el profesor dispone de más tiempo para la resolución de dudas o preparación de clases.
- **Despertar el interés**: al no tener que resolver el problema y esperar la corrección por parte del profesor, el alumno puede realizar más rápidamente los problemas y saber sus errores más comunes.

Actualmente existen multitud de jueces en línea. Algunos de ellos son: ***¡Acepta el reto!*** (Acepta el reto), **Kattis** (Kattis), **SPOJ** (Spoj), **URI** (URI Online Judge), **UVa** (UVa - onlinejudge) y **Codeforces** (Codeforces).

2.1.1 ¡Acepta el reto!

¡Acepta el reto! (Acepta el reto), es una página web lanzada desde la Facultad de Informática de la Universidad Complutense de Madrid, por un grupo de profesores del Grupo de Aplicaciones de Inteligencia Artificial (GAIA) en 2014.

Esta web nace como una iniciativa de dos profesores de la Universidad Complutense de Madrid que en 2011 habían contribuido a poner en marcha ProgramaMe (ProgramaMe), un concurso de programación para alumnos de Ciclos Formativos. Además, habían desarrollado varios problemas similares para realizar pruebas de evaluación continua a sus alumnos de grado.

Debido a que esos problemas no se estaban aprovechando al máximo, decidieron poner en marcha el desarrollo de un juez en línea similar a otros existentes, pero con problemas en español para alumnos de Ciclos Formativos y los primeros años de Universidad.

Este juez en línea acepta soluciones de problemas en C, C++ y Java y corrige automáticamente las soluciones a los problemas enviadas por los usuarios. Dispone de

más de 15000 usuarios hispanohablantes ya que, es de los pocos jueces en línea que dispone de problemas escritos en español y no en inglés como la gran mayoría de jueces.

Actualmente dispone de más de 400 problemas organizados en diferentes subcategorías:

- **Programación**: problemas relacionados con nociones de programación.
- **Exámenes**: problemas que han usado en exámenes o pruebas de alguna institución educativa.
- **Concursos**: problemas usados en concursos.
- **Temática**: problemas categorizados por el tema del enunciado.
- **Recopilaciones**: ejercicios hechos por terceros.

Estas subcategorías, se pueden recorrer, por lo que se simplifica la búsqueda del problema más adecuado para cada momento. Además, muchos de los problemas, están pensados para forzar soluciones con complejidades específicas no solo en tiempo sino también en espacio, algo que es poco habitual en otros jueces.

¡Acepta el reto! dispone de un conjunto de casos de prueba que mantiene en secreto, por lo que cuando recibe el código de la solución a un problema, lo compila y lo ejecuta enviándole por la entrada estándar los casos de prueba que posee.

Además de los veredictos hablados anteriormente, *¡Acepta el reto!* dispone de:

- ***Resctricted Function (RF)***: el código ha intentado realizar una acción peligrosa para el servidor, por lo que se ha cancelado la ejecución.
- ***In Queue (IQ)***: el juez ha recibido el código, pero no ha tenido tiempo de ejecutarlo todavía.
- ***Internal Error (IE)***: el ejecutar el código se ha producido un error, pero no tiene que ver con el código sino con el propio juez en línea.

En la Figura 1, podemos ver la apariencia de los problemas de *¡Acepta el reto!,* donde se puede apreciar el enunciado del problema, y el formato de la entrada y de la salida con unos casos de ejemplo. El usuario debe estar registrado para poder hacer el envío una vez resuelto el problema y el juez de *¡Acepta el reto!* será quién le dé el veredicto.

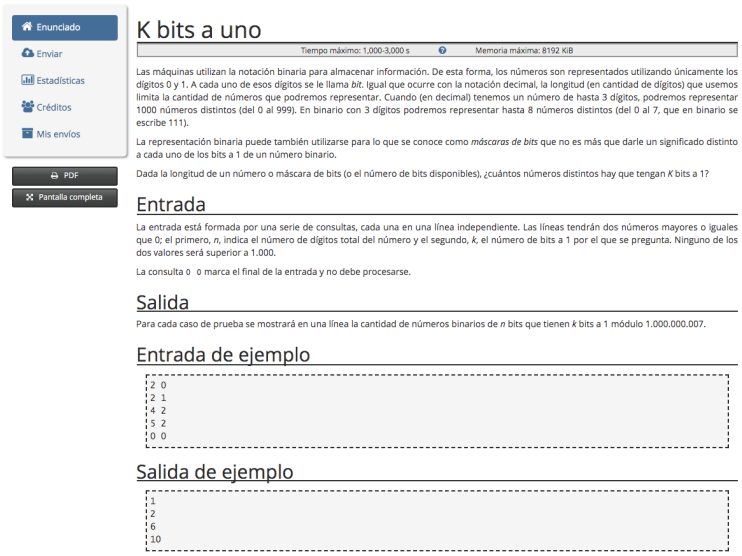


Figura 1: Apariencia de un problema en *¡Acepta el reto!*

2.1.2 Kattis

Kattis (Kattis) es un juez en línea de origen sueco. Ofrece una gran variedad de retos de programación, una buena comunidad y un ranking global. En diversas universidades se utiliza a nivel mundial y algunas compañías lo utilizan en sus procesos de selección de personal. Kattis corrige el código y evalúa la calidad del código.

Con este juez se puede comprobar de plagio ya que poseen un algoritmo de detección de plagio. Se puede hacer un análisis de progreso general. Sus resultados son instantáneos y tiene una amplia variedad de problemas para resolver en más de 10 idiomas.

Además, tiene la posibilidad de ser contratado por grandes empresas a través de las pruebas que proponen dichas empresas.

En la Figura 2, vemos la apariencia de un problema en Kattis. Se ve el enunciado del problema junto con las entradas y las salidas. También vemos que en el lateral derecho tenemos dos botones: el botón de envío de la solución y el de

estadísticas de ese problema. A continuación, tenemos también cierta información relevantes sobre el problema, así como el autor, la fuente y la licencia.

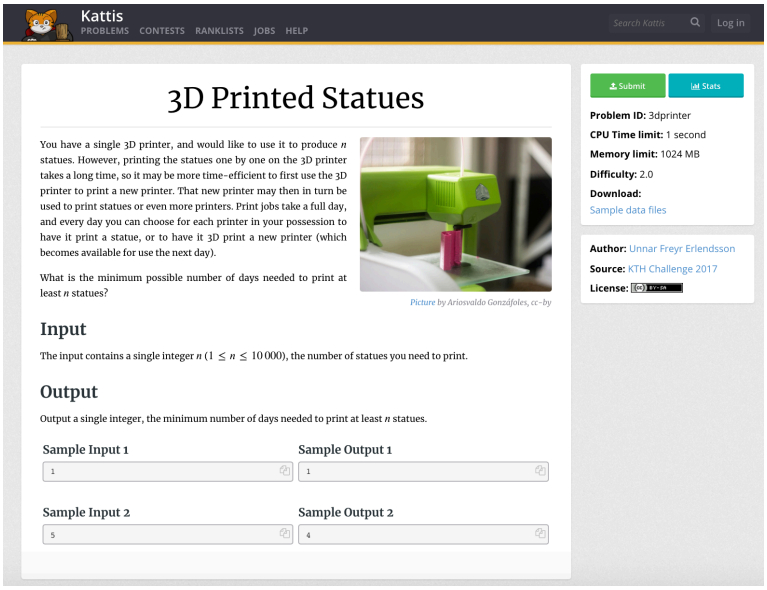


Figura 2: Apariencia de un problema en *Kattis*

2.1.3 SPOJ

La plataforma SPOJ (Sphere online judge) (Spoj) está centrada en el sistema de jueces online que sirve para la evaluación automática de los programas que envían los usuarios.

Soporta más de 45 lenguajes de programación y compiladores incluyendo: C, C++, Pascal, JAVA, C#, Perl, Python, Ruby, Haskell y otros lenguajes esotéricos. Ha tenido un rápido crecimiento del conjunto de problemas y alrededor de 13000 tareas disponibles para practicar 24 horas/día en inglés, polaco, vietnamita, portugués y otros idiomas.

Su sistema de testeo flexible apoya la interacción dinámica con los problemas enviados y la gestión de contenido muy intuitiva.

Tiene más de 2400 concursos organizados desde 2012, desde retos instantáneos hasta cursos de aprendizaje a largo plazo.

En la Figura 3, apreciamos la apariencia de un problema en SPOJ. En este caso también tenemos el enunciado del problema con las entradas y las salidas del mismo y el botón de envío de la solución. A la derecha se ven determinados datos de quién y cuándo ha añadido ese problema y los lenguajes en los que se ha resuelto el problema entre otros datos.

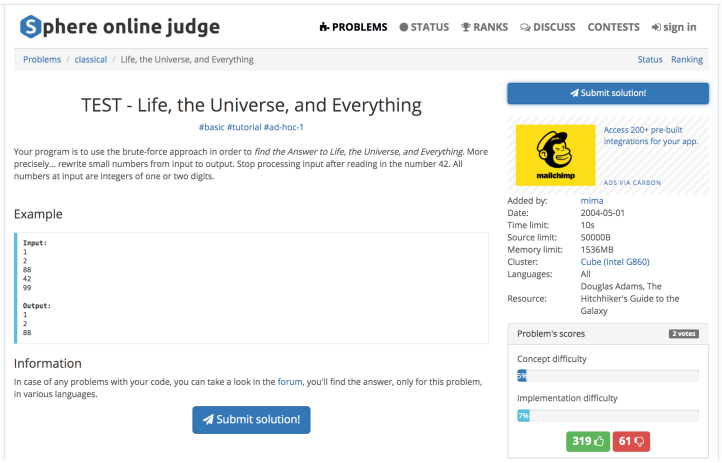


Figura 3: Apariencia de un problema en *Spoj*

2.1.4 URI

El juez online URI (URI Online Judge) es un proyecto que se ha desarrollado por el departamento de Informática de la Universidade Regional Integrada de Brasil. La principal finalidad del proyecto es la práctica de programación y compartir conocimientos.

Este juez online, está disponible en inglés, portugués y español y tiene una resolución de problemas en 11 lenguajes de programación diferentes. Tiene más de 1000 problemas divididos en 8 categorías principales. El personal académico puede crear disciplinas y listas de problemas para evaluar el progreso de los alumnos y obtener *feedback* inmediato.

En la Figura 4 vemos la apariencia de un problema en URI. Igual que en las figuras anteriores, apreciamos el enunciado del problema con las entradas y salidas del mismo.

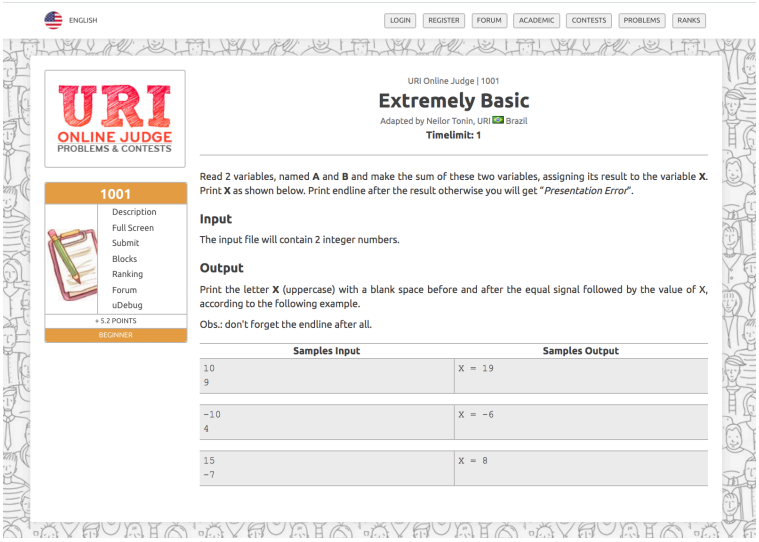


Figura 4: Apariencia de un problema en *URI*

2.1.5 UVa

Online judge (UVa - onlinejudge) es el juez online de más larga tradición ya que, se empezó a poner en marcha a finales de 1995 (Revilla, Manzoor, & Liu, 2010). Este juez pertenece a la Universidad de Valladolid por lo que se le ha conocido como "el juez de la Uva". A día de hoy, ha sufrido varias reimplementaciones y ha pasado por diferentes etapas, por ejemplo, en 2019, los responsables del juez se desvincularon de dicha Universidad y hoy se conoce simplemente como "online judge".

UVa además trabaja con UHunt-UVa (uHunt UVa) que es una herramienta que proporciona estadísticas, selecciones de problemas para resolver y expone una API web para que otros desarrolladores web la desarrollen.

La Figura 5, dispone de una hoja que contiene el problema con las entradas y salidas del mismo. También vemos que, en la parte superior, tenemos una serie de

botones donde podemos enviar la solución, ver las estadísticas, depurar fallos y ver el problema en formato pdf.

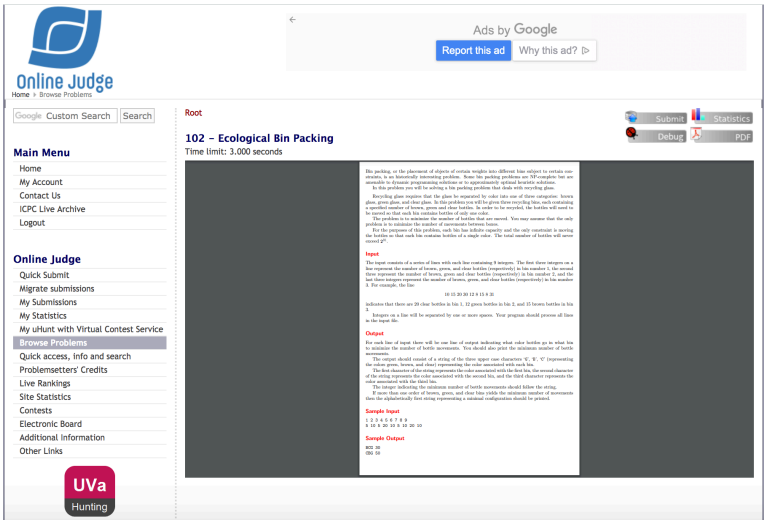


Figura 5: Apariencia de un problema en *Uva*

2.1.6 Codeforces

Codeforces (Codeforces) es una red social dedicada a la programación y a los concursos de programación y a su vez es una plataforma donde regularmente hay concursos y los participantes pueden ver sus habilidades.

Lleva 10 años funcionando y ha albergado más de 700 rondas y competiciones abiertas. El archivo de problemas está próximo a los 6000 y se han alcanzado más de 60 millones de envíos.

Su infraestructura se usa muy a menudo en el mundo para hacer competiciones locales, eventos de entrenamiento y preparación de problemas para diferentes concursos.

En la Figura 6, vemos también que la apariencia del problema en Codeforces es muy parecida a las anteriores conteniendo también el enunciado, entradas y salidas del problema.

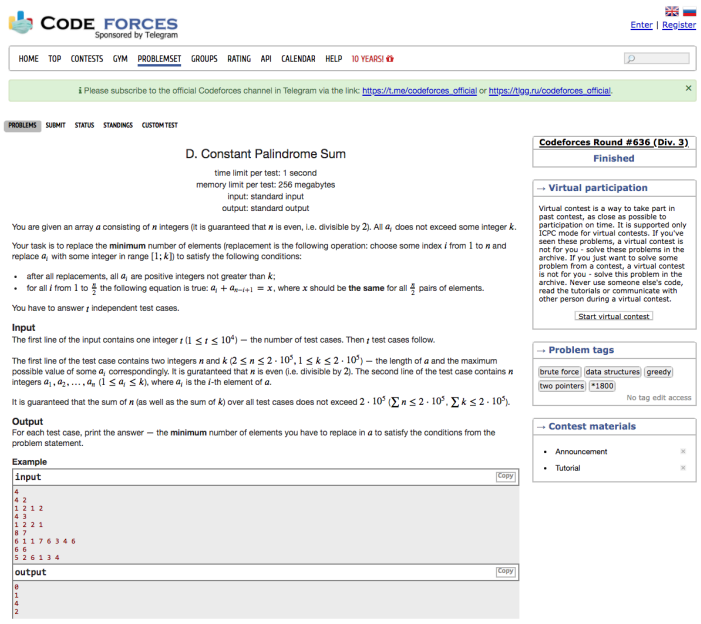


Figura 6: Apariencia de un problema en *Codeforces*

2.2 Dashboards

En nuestro día a día, normalmente nos encontramos gran cantidad de datos que debemos procesar, los cuales encontramos principalmente en tablas relacionadas entre sí que tenemos que ir analizando de una manera exhaustiva.

Un dashboard o panel de control es una representación gráfica de una cantidad determinada de datos. Esos datos, deberían mostrar una información clara y concisa de lo que queremos visualizar, pero hasta la fecha, en algunos casos, se suele prestar demasiada atención al diseño visual y menos a sacar el potencial de los datos importantes, por lo que estos dashboards no funcionan correctamente y por tanto no hay un entendimiento sólido. Fallan en eficiencia y efectividad, pero no por la tecnología usada sino porque la implementación de su diseño es pobre.

Los dashboards surgieron de los sistemas de información ejecutiva (EISs) en los años 80. Su propósito principal en aquel momento fue mostrar muchas claves financieras a través de una simple interfaz que "incluso un ejecutivo podía entender"

pero, en esa misma época, terminaron hibernando ya que la información requerida era incompleta, no fiable y se extendió por demasiadas fuentes dispares (Few, 2006). A día de hoy, sin embargo, los paneles de control son muy comunes en nuestro día a día y nos facilitan la lectura de datos.

Como cada vez disponemos de más avances tecnológicos y de menor tiempo para hacer nuestras tareas de análisis, crear paneles de control claros y en los que podamos visualizar datos de forma inmediata nos ayuda mucho a enfrentarnos en nuestro día a día. Estos paneles de control tienen un gran poder en la percepción visual para comunicar, pero esto solo ocurre si los que los implementan entienden la percepción visual y aplican ese entendimiento a los diseños principales de forma que contengan lo que el usuario ve y piensa.

Los dashboards son una representación visual de información que va a venir originalmente en forma de gráficos y de texto. Que estén principalmente diseñados con gráficos no es porque sea más estético sino porque, habitualmente los gráficos tienen más eficiencia para comunicar y un significado más rico que el texto por sí solo, de esta manera si los gráficos son correctos, los ojos pueden ver rápidamente y el cerebro puede extraer de forma concisa la información más significativa. Stephen Few (Few, 2006) en base a todo lo mencionado, nos advierte acerca de los trece errores que se suelen cometer al diseñar paneles de control y que debemos tener en cuenta:

- Evitar que el scroll proporcione pérdida de información (las pantallas muy largas no permiten ver toda la información de una única vez).
- Añadir información contextual para que las medidas de los gráficos estén enriquecidas.
- Evitar excesivos detalles o precisión.
- Saber qué medimos y qué unidades usamos.
- Elegir la forma correcta de mostrar los datos. Hay veces que es mejor usar una tabla en vez de un gráfico o viceversa.
- No usar el mismo tipo de gráfico para mostrar diferentes datos.
- Hacer un buen diseño de los medios de visualización.
- Representaciones de datos cuantitativos diseñados correctamente y con valores exactos.

- Organizar correctamente los datos.
- Destacar datos importantes de una manera adecuada.
- Añadir decoración inútil en el panel.
- Hacer buen uso de los colores.
- Diseñar una pantalla atractiva.

A la hora de elegir los elementos para construir un panel de control, también hay que tener cuidado. Cuando ya sepamos si vamos a usar gráficas, iconos o cualquier otro elemento, hay que saber dónde lo vamos a ubicar dentro de la página para que tenga más sentido, sea apropiado y sea usable.

Dentro de los elementos existentes para construir un *dashboard,* según Few, hay seis categorías diferentes, dependiendo de lo que queramos mostrar:

- Gráficos.
- Iconos.
- Imágenes.
- Textos.
- Dibujos de objetos.
- Organizadores.

  En nuestro dashboard vamos a usar gráficos, iconos y organizadores (tablas).

2.2.1 Gráficos

En esta categoría predominan la mayoría de los datos cuantitativos que se suelen mostrar en los paneles de control. Existen gran cantidad de gráficos diferentes, pero dentro de los más conocidos están:

- **Gráficos de barras (horizontal y vertical):** Muestran múltiples instancias. Se usan para mostrar medidas que están asociadas con elementos en una categoría.

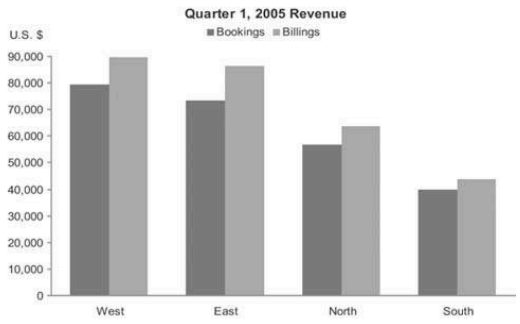


Figura 7: Gráfico de barras vertical extraído de (Few, 2006)

En la Figura 7, vemos dos claves de medida: las reservas y los ingresos subdivididos en las regiones de venta.

- **Gráficos circulares, por sectores o de donut (pie charts):** Este gráfico se usa para mostrar una proporción de casos sobre el total usando porcentajes de cada valor. La representación de datos se lleva a cabo dividiendo un círculo en tantas partes como valores están siendo representados.

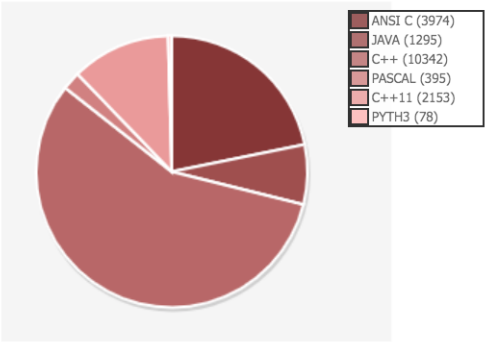


Figura 8: Gráfico por sectores extraído de (UVa - onlinejudge)

En la Figura 8, se ve el número de problemas que se han resuelto en cada uno de los lenguajes que aparece en la leyenda de la gráfica.

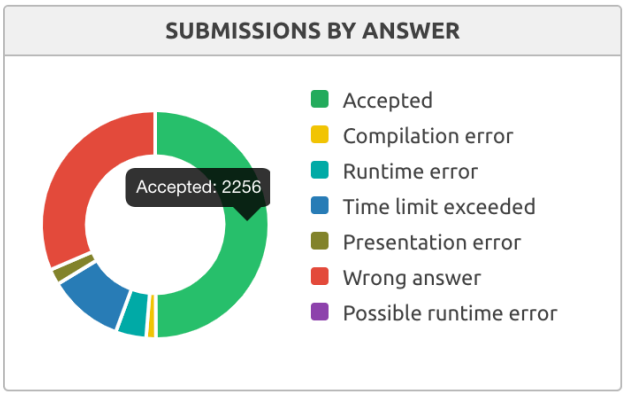


Figura 9: Gráfico de donut extraído de (URI Online Judge)

En la Figura 9, se ve un gráfico en forma de donut con el número de aceptados y el número de cada uno de los errores cometidos en la solución de los problemas por un usuario concreto.

- **Gráficos de barras apiladas (horizontal y vertical):** Es una variación del gráfico de barras. Es una buena elección solo cuando quieres mostrar múltiples instancias de un todo y sus partes, con énfasis en el todo.

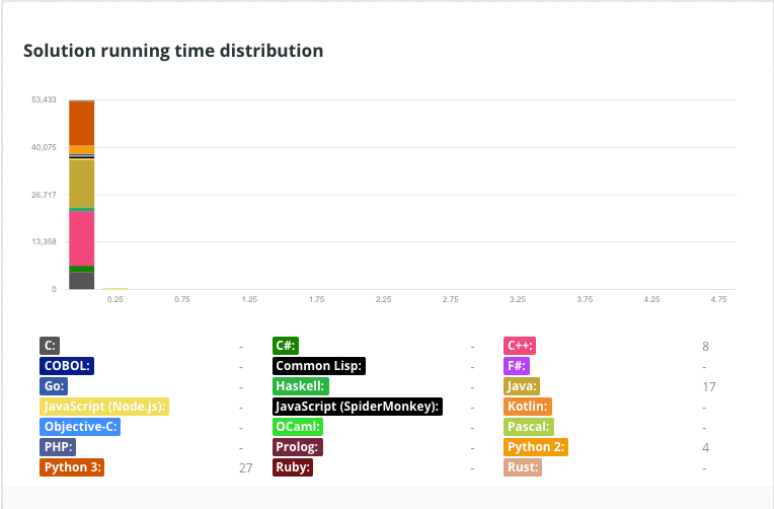


Figura 10: Gráfico de barras apilada extraído de (Kattis)

En la Figura 10, vemos los lenguajes que más se usan en la resolución de los problemas.

- **Gráficos lineales:** Este gráfico, se compone de una serie de datos representados por puntos, unidos por segmentos lineales. Mediante este gráfico se puede comprobar rápidamente el cambio de tendencia de los datos.

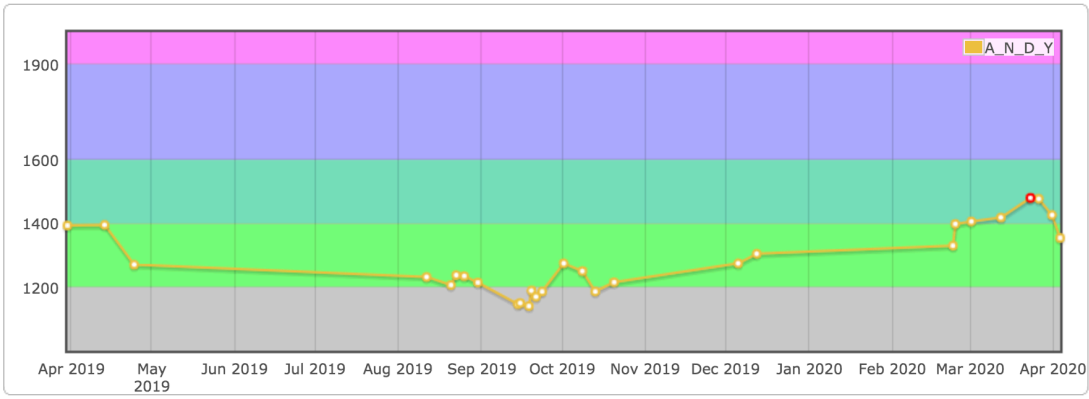


Figura 11: Gráfico lineal extraído de (Codeforces)

En la Figura 11, vemos un gráfico lineal con la evolución de puntos obtenidos en diferentes problemas que ha realizado el usuario.

- **Heatmaps o mapas de calor:** en estos gráficos, se muestra la actividad del usuario en la aplicación. Se puede usar para diferentes iteraciones del usuario con nuestra aplicación, por ejemplo, para saber en qué zona hay mayor navegación, dónde hace mayor número de clics, etc. Pero también se puede usar para saber en que franja del día hay mayor iteración, a qué hora, en qué mes, etc.

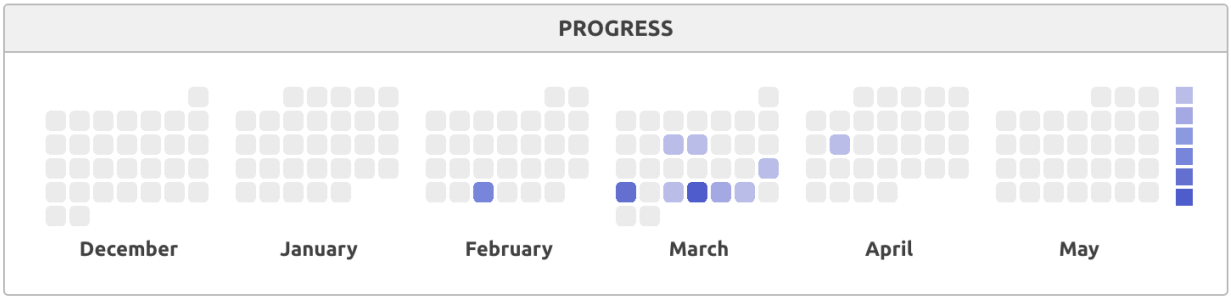


Figura 12: Heatmap extraído de (URI Online Judge)

En la Figura 12, vemos el gráfico del progreso en una determinada franja de tiempo. Siendo el color azul oscuro cuándo más progreso ha habido y el color gris cuándo no ha habido nada de progreso.

- **Gráficos radiales:** Es una buena forma de comparar variables cuantitativas. De esta manera es muy fácil reconocer cuáles son las que son más similares y cuáles no.

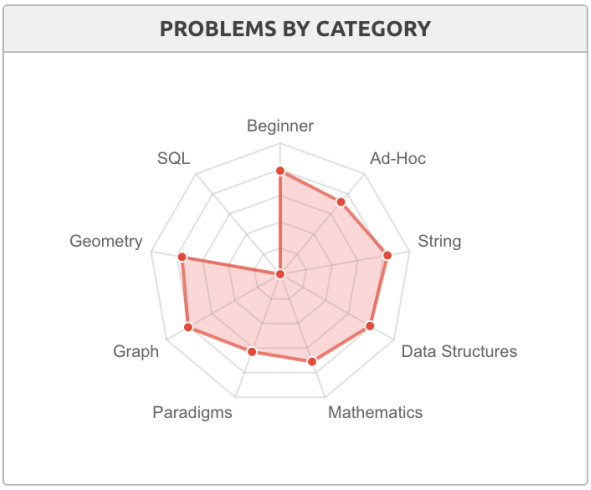


Figura 13: Gráfico radial extraído de (URI Online Judge)

En la Figura 13, vemos qué categorías de problemas suele resolver el usuario.

2.2.2 Iconos

Los iconos son imágenes simples que comunican de manera sencilla y clara. Los iconos más útiles en los paneles de control son:

- **Alertas:** Se usan para atraer atención sobre una cierta información en el panel de control sobre todo cuando algo está mal y requiere cierta atención.
- **Up/down:** Indican si las medidas han crecido o han bajado comparándolas con algún punto en el pasado. Se suelen usar los triángulos o las flechas apuntando hacia arriba, si ha habido una crecida o hacia abajo si ha habido un descenso.
- **On/off:** Se usan para identificar que unos objetos son diferentes a otros. Por ejemplo, si se dispone de una lista de actividades a realizar, se podrían marcar en *on* las ya realizadas y en *off* las que quedan por realizar. También se podrían usar símbolos como verdadero o falso.

En nuestro panel de control, usaremos las alertas para mostrar errores e información sobre algunos datos relevantes.

2.2.3 Organizadores

Según Few (Few, 2006) otra forma de poder organizar los datos es en organizadores los cuales, se usan cuando la información se debe comunicar de una manera determinada. Este tipo de información se puede organizar en:

- **Tablas:** organizan los datos en filas y columnas. Es una buena forma de organizar texto.
- **Mapas espaciales:** se usan para asociar datos categóricos y cuantitativos en un espacio físico. Colocando las medidas en un mapa, ayuda a entender mejor los datos que se están representando.

Dentro de estos organizadores, vamos a usar las tablas de una manera habitual para mostrar datos importantes y el mapa espacial también lo usaremos.

2.3 Ejemplos de Dashboards de jueces en línea

Una vez que sabemos lo que son los paneles de control y sus funcionalidades, los errores comunes que se suelen dar al construir un panel de control y los tipos de elementos de los que se compone, ha llegado el momento de empezar con nuestro diseño del Dashboard para jueces en línea.

Antes de empezar con el diseño, vamos a ver algún ejemplo de panel de control de los jueces en línea de los que hemos hablado anteriormente, para coger algunas ideas y examinarlos.

En la Figura 14 se ve el detalle de un problema en el juez en línea Kattis. En la parte de arriba tenemos el nombre del problema y a continuación, podemos apreciar tres áreas diferentes con información importante acerca del problema.

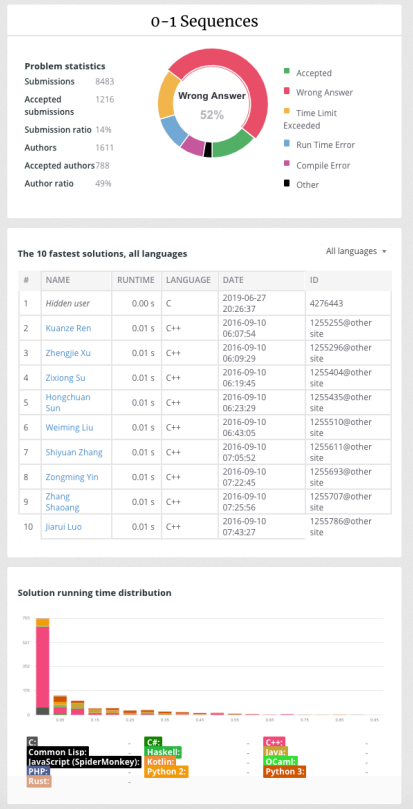


Figura 14: Dashboard del detalle de un problema en el juez en línea Kattis.

Vemos que los elementos que se usan para construirlo son: un gráfico de sectores, texto, una tabla y una gráfica de barras apiladas.

Por su parte, en la Figura 15 vamos a ver el panel de control referente al detalle del usuario en la aplicación de URI, donde apreciamos que hay muchas más áreas diferentes y mucha más información relativa al usuario.

Figura 15: Detalle de un usuario en el juez en línea URI.

En este dashboard vemos como elementos: gráficos por sectores, heatmaps, gráfico de líneas y texto.

Una vez hecha toda la investigación relativa tanto a los jueces en línea como a los dashboards y sabiendo su funcionamiento, nos toca empezar con la fase del diseño dónde pondremos en práctica lo aprendido sobre los paneles de control y haremos un análisis de la competencia para poder extraer las ideas más importantes y tenerlas en cuenta en el siguiente capítulo.

Para crear un aplicación coherente, sencilla y práctica deberemos tener en cuenta lo que nos resulta usable o no de los jueces en línea que vamos a comparar y los errores más comunes que se cometen a la hora de diseñar paneles de control.

Capítulo 3 - Diseño

El diseño guiado por objetivos o DGO (Cooper, Reimann, & Cronin, 2007), es una metodología de diseño implantada por Alan Cooper la cuál he usado a la hora de diseñar la aplicación "Dashboard para jueces en línea". Esta metodología pone en práctica un diseño centrado en el usuario y el proceso de ingeniería de usabilidad.

"*Si diseñamos y construimos productos de tal manera que la gente que los use llegue a sus objetivos, esta gente estará satisfecha, segura y contenta y pagarán con gusto por los productos y los recomendarán a otras personas.*" (Cooper, Reimann, & Cronin, 2007)

Normalmente, los desarrolladores, no ven las necesidades o motivaciones de los usuarios, es decir, se basan solo en las tendencias y limitaciones técnicas, lo cual, no ayuda a un buen resultado ni a una buena coherencia de experiencia de usuario.

Alan Cooper, viendo que la mayoría de los productos están atascados hasta una tercera iteración, propone que el proceso de diseño de la interfaz preceda al código y al testeo para estar seguros de que los productos creados verdaderamente van a conocer las necesidades del usuario.

Para saber si un producto es bueno para el usuario, hay que cumplir tres puntos:

- Conocer a los usuarios.
- Acordar las necesidades humanas y las prioridades de construcción.
- Habilitar un proceso para entender las necesidades del humano como ayuda para desarrollar un producto adecuado.

Alan Cooper, con esta metodología, provee soluciones que conocen las necesidades y objetivos de los usuarios y también engloba los imperativos negocio/organizativo y técnico.

El DGO, se divide en seis fases: investigación, modelado, definición de requerimientos, definición de frameworks, refinamiento y desarrollo, las cuales siguen las cinco actividades de componentes del diseño interactivo diseñado por Gillian

Crampton Smith y Philip Tabor: entendimiento, abstracción, estructura, representación y detalle, con un mayor énfasis en el modelaje del comportamiento del usuario y en las definiciones de los comportamientos del sistema.

Para la realización de esta aplicación, se seguirá el Diseño guiado por objetivos de Alan Cooper, pero no todas sus fases, ya que algunas de ellas no son necesarias para este proyecto.

Primeramente, realizaré la fase de investigación 3.1, la fase de modelado no la ejecutaré ya que en el modelaje se definen a las personas que van a usar la aplicación y en este proyecto hay dos grupos diferenciados: los administradores y los profesores, por lo que ya están modelados. Después llegaré a la fase de los requerimientos de diseño 3.2 y en la cuarta fase definiré el framework de diseño 3.3, donde crearé un primer diseño visual de la aplicación que voy a crear. Cómo basándonos solo en un diseño, seguro que no vamos a cubrir las necesidades de los usuarios, vamos a realizar varias iteraciones con diferentes diseños cada una aproximándonos cada vez más a las necesidades de los usuarios, es decir, vamos a refinar el diseño inicial de forma que cuando lleguemos a la última fase de desarrollo sea lo más real posible.

3.1 Fase de Investigación

Esta fase, va a ser la partida de nuestro estudio DGO. En esta primera fase tenemos un grupo ya preseleccionado de usuarios: los administradores y los profesores. Ambos grupos representan usuarios expertos que conocen la web de "Acepta el reto" muy bien y saben cuáles son los objetivos que persiguen para conocer los datos más relevantes de esa web.

Además de conocer las necesidades de estos usuarios, las cuales vamos a obtener a través de una serie de entrevistas. También queremos conocer qué aplicaciones existen en el mercado actual que sean similares al panel de control para jueces en línea que queremos implementar. Sobre estas aplicaciones vamos a hacer

un estudio detallado de pros y contras de cada una de ellas para saber qué características de cada una de ellas resultan útiles y/ o atractivas a los usuarios y cuáles son los problemas que surgen en cada una de ellas para así crear una aplicación sencilla, usable, útil, detallada y lo más próxima posible a las necesidades de los usuarios.

Para realizar esta fase, siguiendo el DGO, se comienza obteniendo información de los usuarios con una serie de entrevistas para averiguar los objetivos y motivaciones del usuario a través de las respuestas obtenidas durante las entrevistas y después se continuará por el análisis de la competencia.

3.1.1 Entrevistas a los usuarios

En nuestra primera parte de la investigación, queremos realizar a los usuarios de nuestro estudio: administradores y profesores, una serie de preguntas exploratorias para aprender del usuario ya que, al principio, no buscamos algo en concreto, sino saber del usuario. Estas entrevistas, van a estar compuestas de una serie de preguntas orientadas a objetivos para saber qué información necesitan, preguntas orientadas al sistema para saber cómo están usando a día de hoy la web de "Acepta el reto", preguntas orientadas a flujos de trabajo para ver cómo realizan una cierta acción o cada cuanto la realizan y por último preguntas orientadas a actitudes para saber las motivaciones del usuario.

A la hora de efectuar las entrevistas a los profesores, éstas se realizaron a través del software de *Google Encuestas* (Google). Sin embargo, a los administradores se les realizaron personalmente. Cada entrevista duró aproximadamente 30 minutos.

En esta parte de entrevistas, existen dos tipos diferentes:

- La primera, realizada en 2018 que consta de 20 preguntas que se realizaron a profesores y una ampliación de 7 preguntas más que se realizaron a los administradores de esta aplicación. Disponibles en el Apéndice A - Entrevistas Acepta el reto 2018 de esta memoria.

- La segunda realizada en 2020 que consta de dos entrevistas: "Acepta el reto 2020", con 10 preguntas realizadas a profesores de la UCM y a profesores de formación profesional usuarios de www.aceptaelreto.com (Acepta el reto) y "Acepta el reto UCM 2020" realizadas a profesores de la UCM únicamente, ya que solo ellos son usuarios de la plataforma http://ucm.aceptaelreto.es (Acepta el reto). Disponibles en el Apéndice B - Entrevistas 2019 de esta memoria.

Las personas a las que se entrevistó, como ya hemos explicado antes, son usuarios especialistas ya que tanto los profesores como los administradores, tienen buen conocimiento de la aplicación y saben cuáles son sus necesidades para la visualización de resultados.

El objetivo de estas entrevistas era averiguar las necesidades que tenían los usuarios de la aplicación existente respecto a la visualización de resultados, para fijar una meta común entre todos los datos obtenidos y, además, se necesitaba aprender de los usuarios, por lo que en estas entrevistas se propusieron preguntas de distintos tipos enfocadas a ambos grupos. El estudio se realizó sobre 6 profesores y 2 administradores/ profesores. A través de las entrevistas, se obtuvo información sobre las necesidades y carencias a partir de una serie de resultados.

Concluidas las entrevistas del 2018 y las del 2019-2020, se sacan las necesidades de cada uno de los dos grupos participantes en dichas entrevistas. Por lo que las conclusiones a las que llegamos son:

Conclusiones de los profesores entrevistas 2018:

- Para los profesores es importante conocer los problemas que se resuelven del temario propuesto, ya que así pueden ver qué problema es el que más se resuelve y cual menos y pueden preguntar o replantearse porque es así. Así como ver cuántos de esos problemas se han resuelto bien y si se han resuelto mal saber cuáles son los errores que se han producido.
- Consideran interesante incluir un ranking de alumnos donde poder ver que alumnos resuelven más problemas, es decir, obtienen más aceptados.

- Un dato relevante, sería conocer la evolución de los envíos para así saber si se mantiene el mismo ritmo de envíos durante todo el curso, si se incrementa en época de exámenes…
- Resultaría interesante conocer los nuevos problemas existentes en la web de "Acepta el reto".
- Si se pudiese ver el código que han mandado los alumnos, se podría saber si el código escrito para resolver el problema es original o una copia y saber los cambios que se hacen en el código en los diferentes envíos que realiza un alumno. Esta información, no la vamos a marcar como relevante en la implementación de nuestra aplicación, pero sí sería interesante para futuras implementaciones de la web de "Acepta el reto".

Conclusiones de los profesores entrevistas 2020:

- A los profesores les gustaría poder ver qué alumnos han hecho envíos entre todos los que los tienen que hacer y quieren ver un porcentaje de la tasa de éxito, es decir, cuántos intentos han hecho antes de llegar a un aceptado y cuántos intentos se han hecho antes de abandonar el problema sin llegar a resolverlo.
- Este grupo de usuarios, echan en falta un buscador de problemas por título más sencillo y una sugerencia de problemas.
- También les interesaría ver las "amistades" de sus alumnos.
- Una forma muy interesante de ver que problemas se han intentado o no sería con una tabla con colores.
- Los errores que les gustaría visualizar serían los "TLE" (Time Limit Exceeded) frente a los "WA" (Wrong Answer) para ver si se usa o no el algoritmo adecuado.
- Coinciden con las entrevistas del 2018 en que les gustaría ver una progresión de envíos en el tiempo con las tasas de éxito y poder ver el código que envían los alumnos.
- Querrían tener datos de los últimos envíos, los lenguajes usados, el número de envíos frente a los aceptados y los errores obtenidos.

Conclusiones de los administradores entrevistas 2018:

- A este grupo de usuario, les resulta relevante dentro de los errores de la aplicación, saber cuándo existe un "Internal Error" (IE), ya que es un error bastante importante y ellos no tienen notificación a día de hoy sobre él.
- También les resulta importante saber cuál es el problema que más se hace y cual menos, ya que un problema que se hace poco puede ser porque no se entienda bien o porque es demasiado complicado para el nivel de los usuarios que van a resolver los problemas.
- Los nuevos usuarios es un dato que les gustaría obtener ya que de esta manera podrían tener una serie de estadísticas sobre cuántos usuarios nuevos se han registrado por ejemplo en la última semana.
- Otro dato relevante sería visualizar los envíos realizados de manera que pudieran saber, por ejemplo, el mes que ha habido más envíos, el número de aceptados o fallidos frente a los envíos…

Las necesidades obtenidas a través de las entrevistas a cada grupo de usuarios nos ayudarán a la hora de diseñar de forma iterativa la aplicación que queremos realizar.

3.1.2 Análisis de la competencia

Para realizar un buen análisis de la competencia siguiendo la metodología DGO, es necesario seguir una serie de pasos para conseguir la información principal que quiero extraer. Para ello, es necesario identificar quién es nuestra competencia, es decir, que webs/ aplicaciones existen a día de hoy que sean similares al panel de control que queremos crear para *¡Acepta el reto!* (Acepta el reto).

A continuación, vamos a ver las principales webs/ aplicaciones que vamos a analizar, de las cuales ya hemos hablado anteriormente:

- Kattis (Kattis).
- Spoj (Spoj).

- URI (URI Online Judge).
- uHunt (uHunt UVa).
- Codeforces (Codeforces).

Una vez identificadas las aplicaciones/ webs, se van a detallar las dimensiones del producto en las que nos vamos a centrar para el análisis de la competencia y vamos a analizar a la competencia dentro de esas dimensiones. Van a ser cuatro:

- Detalles del usuario
- Detalles del problema
- Listado de problemas
- Rankings

3.1.2.1 Detalles del usuario

En esta dimensión, se definen los detalles del usuario, es decir, aquellos datos que vemos que son relevantes para mostrar en el dashboard del usuario referentes al propio usuario y a los problemas intentados:

- Datos usuario
- Últimos envíos
- Envíos realizados
- Nº de problemas resueltos/enviados
- Efectividad de los envíos
- Tipos de veredictos obtenidos
- Estadísticas de los envíos
- Actividad últimos meses/año
- Listado problemas resueltos

Detalles usuario/ Jueces	Kattis	Spoj	URI	uHunt	Codeforces
Datos usuario	SÍ	SÍ	SÍ	NO	SÍ
Últimos envíos	NO	SÍ	NO	SÍ	NO
Envíos realizados	NO	NO	SÍ	NO	NO
Nº prob resueltos/ enviados	NO	SÍ	SÍ	SÍ	NO
Efectividad envíos	NO	SÍ	SÍ	NO	NO
Tipos de veredictos obtenidos	NO	NO	NO	SÍ	NO
Estadísticas de envíos	NO	NO	NO	SÍ	NO
Actividad últimos meses/ año	NO	SÍ	SÍ	NO	NO
Listado problemas resueltos	NO	SÍ	NO	NO	NO

Tabla 1: Matriz de resultados para la dimensión *Detalles del usuario*

3.1.2.2 Detalles del problema

En esta dimensión, los datos son en relación con cada problema y los datos importantes referentes a él:

- Listado de envíos
- Gráfico de veredictos
- Gráfico de lenguajes usados
- Total de envíos
- Total usuarios únicos que lo han intentado
- Total usuarios que tienen AC
- Top 20 usuarios
- Top 10 soluciones más rápidas

Detalles problema/ Jueces	Kattis	Spoj	URI	uHunt	Codeforces
Listado envíos	NO	NO	SÍ	NO	SÍ
Gráfico de veredictos	SÍ	NO	SÍ	SÍ	NO
Gráfico de lenguajes usados	NO	NO	NO	SÍ	NO
Total envíos	SÍ	NO	SÍ	SÍ	NO
Total usuarios únicos que lo han intentado	NO	NO	NO	SÍ	NO
Total usuarios que tienen AC	SÍ	NO	SÍ	SÍ	NO
Top 20 usuarios	NO	NO	NO	SÍ	NO
soluciones más rápidas	SÍ	NO	NO	NO	NO

Tabla 2: Matriz de resultados para la dimensión *Detalles del problema*

3.1.2.3 Listado de problemas

En esta dimensión los datos son en relación con todos los problemas existentes:

- Total envíos
- Total de envíos/total resueltos
- Total usuarios/total resueltos
- Total usuarios
- Total aceptados
- Dificultad

Listado problema/ Jueces	Kattis	Spoj	URI	uHunt	Codeforces
Total envíos	SÍ	NO	NO	SÍ	NO
Total envíos/ total resueltos	SÍ	NO	NO	SÍ	NO
Total usuarios/ total resueltos	SÍ	NO	NO	SÍ	NO
Total usuarios	SÍ	NO	NO	NO	NO
Total aceptados	SÍ	SÍ	SÍ	NO	SÍ
Dificultad	NO	SÍ	NO	NO	SÍ

Tabla 3: Matriz de resultados para la dimensión *Listado de problemas*

3.1.2.4 Ranking

La última dimensión está relacionada con el ranking tanto de usuarios como de problemas:

- Usuarios
- Problemas
- Países
- Instituciones
- Lenguajes

Ranking/ Jueces	Kattis	Spoj	URI	uHunt	Codeforces
Usuarios	SÍ	SÍ	NO	SÍ	SÍ
Problemas	NO	SÍ	SÍ	NO	NO
Países	SÍ	SÍ	NO	NO	NO
Instituciones	SÍ	SÍ	NO	NO	NO
Lenguajes	NO	SÍ	NO	NO	NO

Tabla 4: Matriz de resultados para la dimensión *Ranking*

Una vez detalladas todas las dimensiones de cada una de las webs/ aplicaciones, las conclusiones obtenidas en relación a mi proyecto son:

- Al analizar cada uno de ellos, principalmente se rigen por un número de puntos de cada usuario para los rankings de usuarios, evolución a lo largo del tiempo… como nosotros no nos regimos por puntos esa información no va a ser de gran relevancia, pero si me va a servir para poder tener ideas a la hora de diseñar mis plantillas para crear tablas o diagramas.
- Analizando el detalle del usuario, se aprecia que los datos del usuario son importantes ya que casi todas las webs/ aplicaciones, disponen de ellos.

- En cuanto a los envíos realizados por el usuario, los envíos totales son algo relevantes. Creemos que es interesante para el usuario conocer los últimos envíos realizados para saber los veredictos y saber un porcentaje de aceptados frente a envíos, es decir, cuántas veces ha tenido que intentar el problema para llegar a resolverlo correctamente.
- Las estadísticas de envíos creo que también es un punto fuerte para el usuario, de esta manera puede ver cuáles son los veredictos de todos los problemas resueltos: cuantos aceptados tiene, qué tipo de errores ha tenido… En esta parte también sería interesante tener una evolución de envíos en el tiempo.
- La información de los problemas aceptados, fallidos y el número de intentos también es un dato relevante para el usuario. De hecho, la idea de UHunt de la tabla con todos los problemas y los diferentes colores para saber si están resueltos, intentados, pero con errores o no intentados me resulta muy interesante.
- En cuanto al detalle de un problema, es interesante saber cuántos intentos tiene, cuántos aceptados y cuántos fallidos y los gráficos de los lenguajes en los que se ha resuelto y de los veredictos obtenidos (aceptados y tipos de errores).
- El listado de los últimos envíos también puede ser un dato interesante para mostrar y el tipo de errores con el número de ocurrencias de cada uno de ellos.
- Para el listado de problemas las dificultades no nos interesan ya que nuestros problemas no tienen diferentes niveles de dificultad y en cuanto al resto de opciones, para tenerlas lo más resumidas posibles creo que es muy interesante el número de resueltos frente a los envíos y frente a los usuarios. Es una buena forma de saber que problemas se resuelven más fácilmente y cuales se resuelven más por los usuarios.
- Para acabar en cuanto a los rankings, creo que para mi proyecto los más interesantes son los el de los usuarios y el de los problemas ya que el de países, instituciones o lenguajes tiene más sentido para concursos.

3.2 Diseño de requisitos funcionales

Ahora que ya tenemos el análisis de los usuarios y el análisis de la competencia, vamos a definir los requisitos funcionales en base a las conclusiones obtenidas. Estos requisitos, nos van a ayudar a realizar nuestros diseños iterativos de la aplicación que queremos implementar.

Como tenemos dos tipos de usuarios principales, vamos a crear una aplicación que la puedan usar ambos. Las necesidades no siempre son las mismas por lo que dependiendo del usuario que sea, vamos a crear distintos requisitos funcionales.

3.2.1 Requisitos funcionales de los administradores

En cuanto a los administradores, sus principales necesidades, no tienen que ver en si con los usuarios, sino con el funcionamiento de la aplicación en general.

Vamos a mostrarles, cuándo hay ocurrencias de "Internal Error" (IE), ya que a día de hoy no tienen notificaciones acerca de él a no ser que un usuario se lo notifique a través de un correo electrónico.

También vamos a mostrarles los nuevos usuarios registrados en la última semana, de forma que podrían tener una serie de estadísticas de los últimos registros de la semana, mes, año y así poder tener una evolución en el tiempo de registros.

En este panel de control, vamos a exponer también: que problema es el que más aceptados tiene, el que resulta más sencillo (intentos frente a aceptados) y el más complicado (intentos sin llegar a aceptado).

Finalizando los requisitos, vamos a presentar también: los lenguajes que más se usan, los que menos, los aceptados, los errores y las ocurrencias de errores.

3.2.2 Requisitos de los profesores

En cuanto a los profesores, sus necesidades difieren un poco de las de los administradores. A ellos les resulta más interesante conocer datos de sus alumnos, sobre todo.

En el panel de control de los profesores, vamos a mostrar que problemas se resuelven del temario propuesto sabiendo, cuáles son los que más se resuelven y cuales menos. También vamos a enseñar, los alumnos que consiguen llegar al aceptado y cuáles no porque se han rendido.

La evolución de los envíos a lo largo del curso también es lo enseñaremos de forma que puedan ver cuándo hay más o menos envíos y que errores se producen en esos envíos.

A continuación, con los requisitos de diseño definidos vamos a continuar con el siguiente paso de la metodología DGO: el diseño iterativo.

3.3 Diseño iterativo

Una vez realizada la investigación a través del análisis de la competencia y de las entrevistas a los profesores y a los administradores, se obtienen una serie de ideas para la realización de dicho dashboard, por lo que se fueron creando las diferentes plantillas de las diferentes secciones del panel de control para los administradores y los profesores.

En esta fase, se crea una primera iteración que va a ser un diseño sencillo con ideas obtenidas, pero nada próximo a la realidad de lo que quieren los usuarios ya que, en sucesivas iteraciones, será donde se diseñen el resto de las iteraciones para así llegar a esa realidad.

3.3.1 Primera iteración

En esta iteración, la aplicación creada para el administrador y para el profesor se construyen de forma casi similar creando así un diseño inicial que se irá modificando en las siguientes iteraciones. En los administradores los usuarios son globales y en los profesores son alumnos de su asignatura y de su grupo.

En esta primera imagen, véase la Figura 16, se muestra la página referente al administrador donde, se pueden apreciar cuatro áreas:

- **Área de totales:** esta primera área está compuesta por cuatro módulos que nos muestran una serie de números relacionados con los usuarios (usuarios totales), países (países participantes), problemas (problemas totales) e intentos (intentos totales).
- **Área de lenguajes usados, aceptados y fallidos:** la segunda área acoge tres diagramas referidos a los lenguajes de programación, a los problemas aceptados y a los problemas fallidos. En esta área, con solo tres módulos, tenemos una idea global muy visual de las tres propiedades antes nombradas.
- **Área geográfica:** esta tercera área, contiene un mapamundi donde se muestran los países participantes con colores, los que no participen aparecen en gris, y una tabla con un listado en el que aparece el país, la bandera, el número de participantes y el porcentaje con respecto al total de usuarios.
- **Área de problemas con más AC:** la cuarta área, se compone de una tabla con un listado de los problemas con más aceptados. Esta tabla tiene un id, el nombre del problema, aceptados frente a envíos y aceptados frente a usuarios.

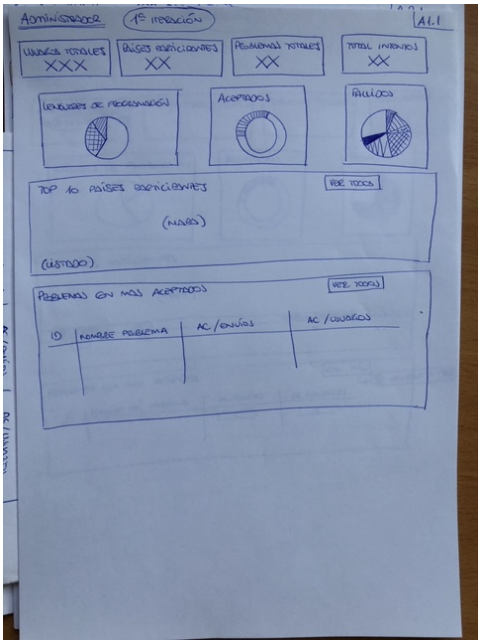


Figura 16: Vista previa de la página del administrador

En la segunda imagen, véase la Figura 17, se muestra la página referente al profesor, donde se aprecian cinco áreas diferentes.

- **Área descriptiva:** nos muestra el nombre de la asignatura y el curso al que va referido.
- **Área de totales:** esta segunda área está compuesta por cuatro módulos que nos muestran una serie de números relacionados con los usuarios o alumnos (usuarios totales), países (países participantes), problemas (problemas totales) e intentos (intentos totales).
- **Área de lenguajes, aceptados y fallidos:** la tercera área acoge tres diagramas referidos a los lenguajes de programación, a los problemas aceptados y a los problemas fallidos. En esta área, con solo tres módulos, tenemos una idea global muy visual de las tres propiedades antes nombradas.
- **Área geográfica:** esta cuarta área, contiene un mapa del mundo donde se muestran los países participantes con colores, los que no participen aparecen en gris, y una tabla con un listado en el que aparece el país, la bandera, el número de participantes y el porcentaje con respecto al total de usuarios.

- **Área de problema con más AC:** la quinta área, se compone de una tabla con un listado de los problemas con más aceptados. Esta tabla tiene un id, el nombre del problema, aceptados frente a envíos y aceptados frente a usuarios.

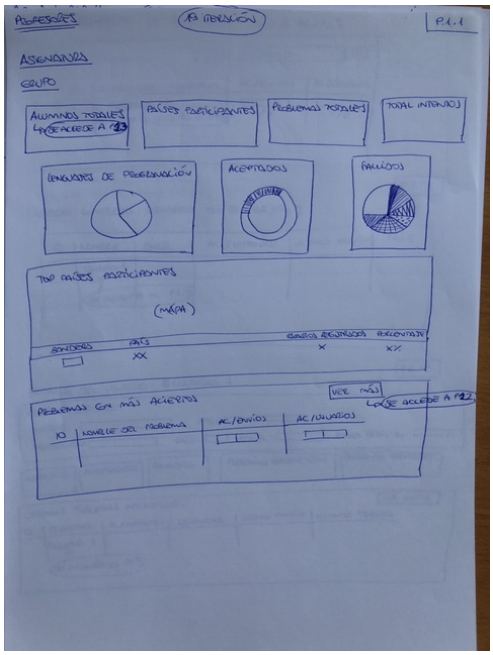


Figura 17: Vista previa de la página del profesor.

En la tercera imagen Figura 18, vemos la página del listado de problemas para el grupo.

En esta página, se encuentra el área de problemas, donde se ve una tabla que contiene todos los problemas de la web con su id, nombre (haciendo clic accederíamos a la página de detalle del problema), problemas aceptados frente a envíos y problemas aceptados frente a usuarios que han realizado el problema.

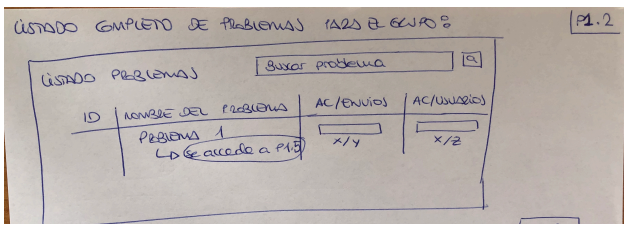


Figura 18: Vista previa del listado de problemas para el grupo.

En la cuarta imagen, véase la Figura 19, se encuentra un área de alumnos, donde se ve una tabla que contiene todos los alumnos del grupo: id, nombre (haciendo clic llegaríamos al detalle del alumno), apellidos, email, problemas aceptados frente a intentos de problemas, último envío y el país al que pertenecen.

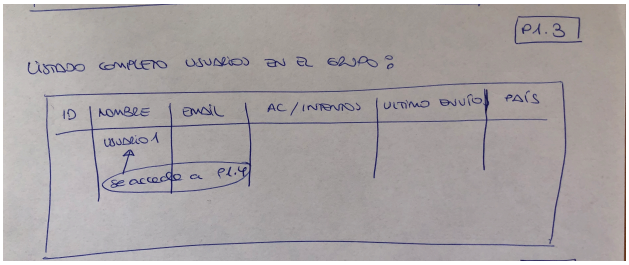


Figura 19: Vista previa del listado de alumnos de grupo.

En la quinta imagen, véase la Figura 20, está la página del detalle del usuario/alumno diferenciamos tres áreas:

- **Área descriptiva:** en esta zona vemos una descripción del alumno donde se muestra el perfil completo: nombre, apellidos, institución, país, email y fecha de registro.
- **Área de totales:** en esta área tenemos cinco módulos que nos muestran el número de intentos de todos los problemas, el número de aceptados, el número de fallidos, problemas enfrentados y problemas resueltos.
- **Área de últimos problemas intentados:** en esta zona tenemos una tabla con los últimos problemas intentados donde aparece el id del problema, el nombre del problema (haciendo clic accedemos a la página del detalle del problema a p1.5), los aceptados frente a los intentos, el lenguaje en el que se ha resuelto, el último envío y estado.

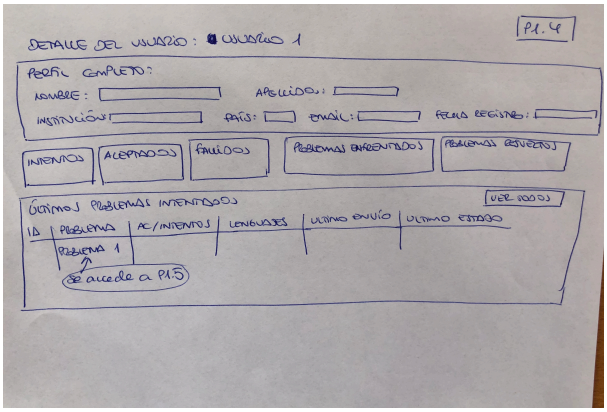


Figura 20: Vista previa de la página del detalle de usuario.

En la cuarta imagen, véase la Figura 21, apreciamos la página del detalle de problema. En esta página, tenemos cuatro áreas:

- **Área de totales:** En esta área tenemos cinco módulos que nos muestran de forma numérica el número de intentos del problema que se está visualizando, el número de aceptados, el número de fallidos, el número de usuarios únicos que han intentado el problema y el número de DACU (número de usuarios únicos que han conseguido al menos un AC para ese problema).
- **Área de evolución de los envíos:** En esta zona tenemos un gráfico con la evolución de los intentos de ese problema junto con una tabla de usuarios que han resuelto ese problema: usuario, aceptados frente a intentos, lenguaje usado, última fecha de envío y el último estado del problema.
- **Área de lenguajes, errores y aciertos por país:** Esta área, se compone de tres módulos: diagrama con los lenguajes de programación usados para ese problema, diagrama con los errores presentados y el diagrama con los aciertos por países.
- **Área de errores:** se compone de una tabla con los errores que se cometen en dicho problema. Esta tabla tiene: el prefijo del error, el tipo de error, el total de ocurrencias y las ocurrencias frente a los errores totales de ese problema.

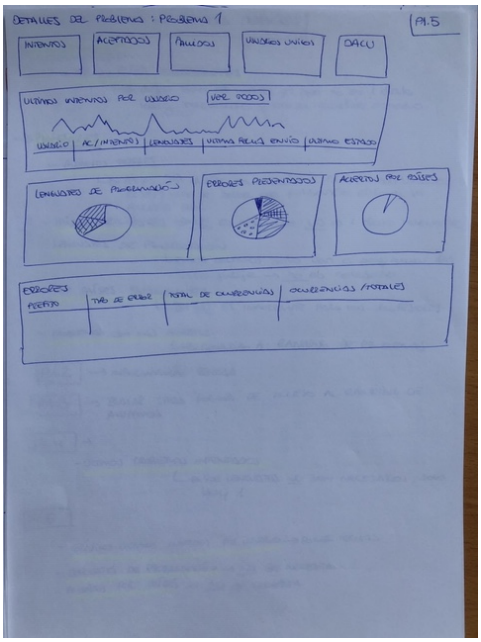


Figura 21: Vista previa de la página del problema.

Después de esta primera iteración, se hace una revisión minuciosa de cada una de las páginas con correcciones, errores… a tener en cuenta para la segunda iteración:

- Como se aprecia en esta iteración, en general, las páginas no siguen una coherencia interna, es decir, cada una tiene un orden de áreas diferente e incluso los nombres de las tablas similares difieren también entre ellos. Esto no es usable ya que un usuario, espera siempre lo mismo en el mismo sitio.
- En la página del administrador, en el área geográfica, no es necesario mostrar una tabla con todos los países participantes ya que ocupa mucho espacio y apenas es relevante, por lo que dejamos únicamente el mapa con una etiqueta con porcentajes.
- En la página de los profesores, no se sabe que al hacer clic en el módulo de usuarios totales del área numérico accedas al listado completo de alumnos, por lo tanto, no es usable, hay que buscar otra forma de acceder a ese listado y el módulo de los países participantes no es un dato relevante para los profesores.
- Los lenguajes de programación no es un dato relevante ya que solo van a usar un lenguaje: C++.

- El top de países participantes no es un dato relevante para los profesores.
- El listado de problemas con más aciertos no sigue una coherencia interna con el resto, llamarlo "Ranking de problemas".

3.3.2 Segunda iteración

En esta iteración vamos a cambiar los diseños de forma que eliminemos todo aquello que no era necesario en la iteración anterior, vamos a seguir una coherencia interna en las páginas para que sean similares entre ellas y vamos a buscar usabilidad en alguna de las funciones de clic que en la primera iteración no tenían validez.

En esta primera imagen, véase la Figura 22, se muestra la página del administrador donde, se diferencian siete:

- **Área de alertas**: en esta área, he añadido dos alertas que son de utilidad para los administradores: el número de ***"Internal Errors"*** (errores internos), ya que es un dato muy relevante para ellos y el número de los nuevos usuarios registrados.
- **Área de totales** permanece igual que en la primera iteración.
- **Área de lenguajes, aceptados y fallidos** permanece igual que en la primera iteración.
- **Área geográfica**: en esta cuarta área, vamos a añadir al mapa los porcentajes de participación y el listado contendrá una tabla con el país, la bandera y el número de participantes.
- **Área de problemas con más AC:** la quinta área, sigue siendo el listado de los problemas con más aceptados de la primera iteración, pero además de tener el id y el nombre del problema, vamos a añadir el último veredicto, el número de aceptados y el número de fallidos.
- **Área de problemas con más fallidos:** que va a tener un listado de los problemas con más fallidos, ya que interesa saber cuáles son los problemas que más fallan los estudiantes. De esta manera se puede saber si no se entienden los problemas al

resolverlos, si al realizar el envío está fallando la recepción, si falla en la corrección el juez... Esta tabla tiene un id, el nombre del problema, el último veredicto, el número de aceptados y el número de fallidos.

- **Área de últimos envíos:** va a contener un listado con los últimos envíos de la semana. Esta tabla está formada por el nombre, apellidos, país del usuario, el nombre del problema, la fecha de envío, el veredicto y el número de intentos.

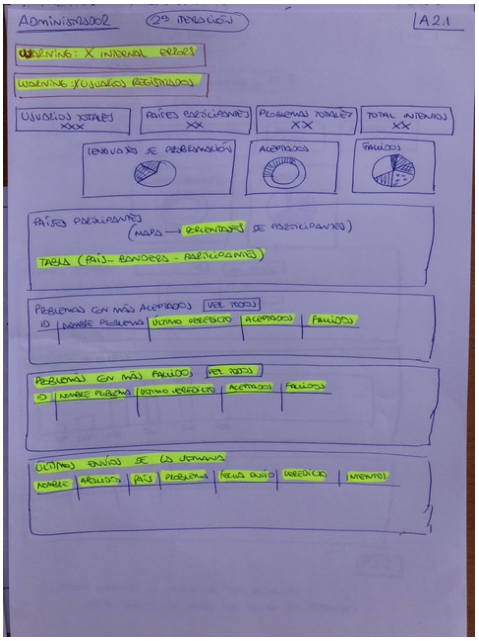


Figura 22: Vista previa de la página del administrador.

En esta segunda imagen, véase la Figura 23, se muestra la página del profesor. Esta página está dividida en seis áreas:

- **Área descriptiva:** además del nombre de la asignatura y el curso al que va referido, hemos añadido una alerta en la que aparecerían los nuevos problemas añadidos a la aplicación.
- **Área de totales:** esta segunda área está compuesta por cuatro módulos que nos muestran una serie de números relacionados con los usuarios o alumnos (usuarios totales), se mantiene igual que en la primera iteración, problemas (problemas del curso), número de envíos realizados y número de aceptados, estos tres últimos módulos cambian con respecto a la primera iteración.

- **Área de aceptados/fallidos y errores:** en esta segunda área unificamos los aceptados frente a los fallidos, eliminamos el diagrama de los lenguajes y mantenemos la gráfica referente a los tipos de errores.
- **Área últimos envíos:** esta área muestra los últimos envíos de problemas, donde aparece el id, el nombre del problema, los aceptados frente a los envíos y los aceptados frente a los usuarios.
- **Área de la evolución de los envíos**: este espacio está dedicado a la evolución de los envíos, mediante un gráfico de barras, en el cual el profesor puede de un simple vistazo cuando ha habido más o menos envíos durante el curso.
- **Área del ranking AC**: nos encontramos otra tabla con el ranking de aceptados donde aparece el nombre, apellido, email, número de intentos, número de aceptados y número de fallidos.

La página del listado de problemas de la primera iteración se mantiene.

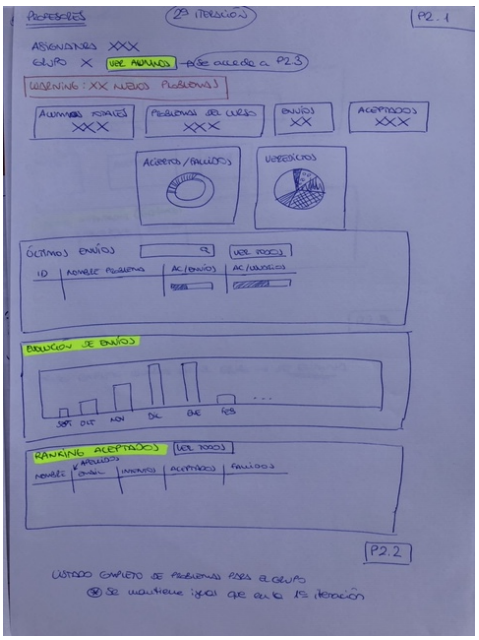


Figura 23: Vista previa de la página del profesor.

En esta tercera imagen, véase la Figura 24, se aprecia la vista del detalle del usuario.

En esta página identificamos cuatro áreas:

- **Área descriptiva:** mostramos solo el nombre y apellidos y el email del usuario.

- **Área de la evolución de los envíos:** añadimos una gráfica de barras con la evolución de los envíos a lo largo del curso.
- **Área de totales:** esta área está formado por tres módulos que contienen: el número de intentos, el número de aceptados y el número de fallidos.
- **Área de problemas intentados**: se compone de una tabla con los últimos problemas intentados: id, nombre del problema, fecha del envío, veredicto y número de intentos.

La página del listado completo de usuarios por grupo de la primera iteración se elimina. No es relevante.

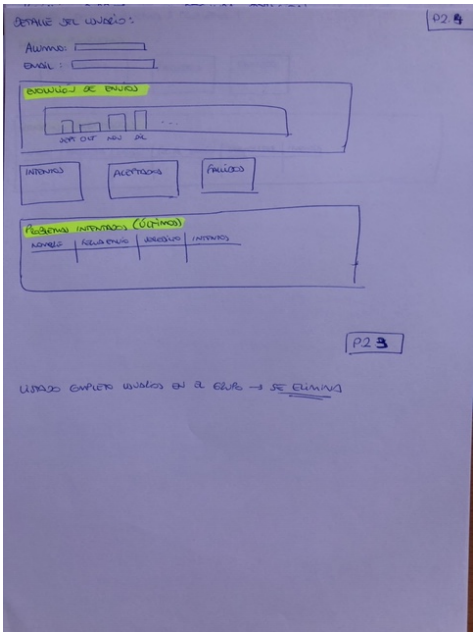


Figura 24: Vista previa del detalle del usuario.

En esta cuarta imagen, véase la Figura 25, se muestra la página del detalle de un problema. Esta página se compone de dos áreas:

- **Área de totales**: compuesta solo por tres módulos: número de aceptados, número de fallidos y número de envíos, en vez de los cinco de la iteración anterior.
- **Área de ranking de problemas:** se compone de una tabla con el nombre, los apellidos y el email, del usuario que lo ha hecho, la fecha del envío, el veredicto y los intentos.

Con respecto a la iteración anterior, esta página elimina gran parte de su contenido anterior cambiando el aspecto.

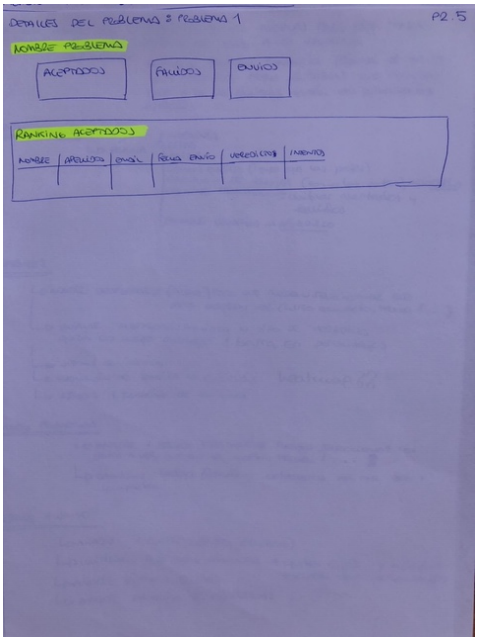


Figura 25: Vista previa del detalle del problema.

Después de esta segunda iteración, se vuelve a hacer otra revisión de los nuevos diseños para corregir de nuevo, ver errores y ver qué información relevante nos sigue faltando para realizar un tercer diseño de la aplicación:

- En esta revisión, seguimos apreciando que el aspecto general de las páginas es diferente, por lo que tenemos que seguir una coherencia interna.
- En la página del administrador, añadir una evolución de envíos igual que en la página del profesor. En la tabla de los últimos envíos unificar esa tabla con la de los profesores.
- En esta página además se necesita añadir un ranking de usuarios igual que se va a añadir en la página de los profesores.
- En la página de los profesores es necesario añadir un desplegable para que se pueda elegir los datos que se quieren ver: curso completo, tema 1, tema 2, … Hay que unificar los aceptados con los fallidos y el tipo de veredicto, quizá es mejor añadir una barra con porcentajes.

- También es necesario añadir el ranking de alumnos, para que tenga una coherencia interna.
- En el listado de problemas, hay que añadir un desplegable para ver qué datos se quieren ver, si por curso completo, tema 1, tema 2, … También hay que cambiar las cabeceras de la tabla para que sea más completa.

En la página del detalle del alumno añadir un perfil con los datos del alumno, unificar los aceptados con los fallidos, quitar los módulos y añadir una barra con porcentajes. Añadir una tabla con los últimos envíos y añadir el ranking de problemas.

3.3.3 Tercera iteración

En esta iteración vamos a cambiar los diseños de forma que eliminemos todo aquello que no era necesario en la iteración anterior, vamos a seguir una coherencia interna en todas las páginas para que sean similares entre ellas y vamos a buscar que todo sea usable.

En esta primera imagen, véase la Figura 26, se muestra la página del administrador donde, se diferencian finalmente nueve áreas:

- **Área de alertas:** esta área se mantiene igual que en la segunda iteración.
- **Área de totales:** esta área se mantiene igual que en la segunda iteración.
- **Área de lenguajes, aceptados y fallidos:** esta área se mantiene igual que en la segunda iteración.
- **Área geográfica:** en la cuarta área, vamos a mantener el mapa del mundo con los países participantes y vamos a eliminar el listado.
- **Área de últimos envíos:** esta quinta área, está formada por una tabla y unos diagramas.

La tabla de los últimos envíos la movemos de lugar y en la tabla vamos a mostrar la información del nombre, apellidos y email del usuario, el problema resuelto,

el veredicto, el lenguaje, los aceptados frente a los envíos, los aceptados frente a los usuarios y la fecha del envío.

- **Área de evolución de los envíos:** gráfica con barras con una evolución de envíos con un diagrama de barras en el que se muestran los meses del curso y el volumen de envíos de cada mes.
- **Área de problemas intentados:** es una tabla donde aparecen los ids de todos los problemas y muestra si están intentados (color verde) o no (sin color). Cuando se pasa el ratón por encima existe una etiqueta con el nombre del problema y se accede directamente al detalle del problema si se hace clic.
- **Área de ranking de problemas:** en esta área, añadimos una tabla con un ranking de los problemas, donde se muestran los diez problemas que más aceptados han obtenido frente a los envíos. Esta tabla se compone del nombre del problema y el id, el último veredicto obtenido, el lenguaje en el que se resuelto, aceptados frente a envíos, aceptados frente a usuarios y la fecha del último envío.
- **Área de ranking de usuarios:** añadimos un ranking de usuarios, donde se muestran los diez usuarios que más problemas diferentes han resuelto. Esta tabla, se compone del nombre, apellidos y email del usuario, el lenguaje o lenguajes que utiliza para resolver los problemas, los aceptados frente a los envíos del usuario, los aceptados frente a los fallidos del usuario y la fecha del último envío.

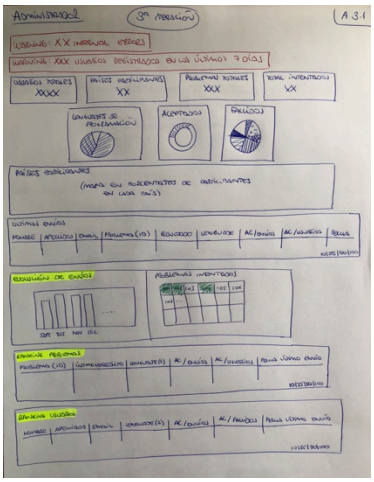


Figura 26: Vista previa de la página del administrador

En esta segunda imagen, véase la Figura 27, se muestra la página del profesor. Esta página está dividida en siete áreas:

- **Área descriptiva:** en esta área, añadimos un desplegable donde el profesor puede elegir los datos que quiere ver: curso completo o por temas, y además hay una alerta en la que aparecerían los nuevos problemas añadidos a la aplicación. Desde esta área, vamos a poder acceder al listado completo de los alumnos del grupo y al listado completo de problemas del curso.
- **Área de aceptados/fallidos:** hemos dejado los cuatro primeros módulos y hemos unificado los aceptados frente a los fallidos en una barra.
- **Área de últimos envíos:** añadimos una tabla de los últimos envíos de problemas, va a mostrar el nombre, apellido y email del alumno, el nombre del problema con el id, el resultado, los aceptados frente a los envíos y la fecha de envío.
- **Área de la evolución de los envíos:** mantenemos el gráfico de la evolución de los envíos.
- **Área de problemas intentados:** es una tabla donde aparecen los ids de todos los problemas del curso y muestra si están intentados (color verde) o no (sin color). Cuando se pasa el ratón por encima existe un tooltip con el nombre del problema y se accede directamente al detalle del problema si se hace clic.

La evolución de los envíos, que ya estaba en la iteración anterior y una tabla donde se muestran los problemas del curso. Mediante una tabla formadas por los ids de los problemas vemos que problemas se han intentado (color verde) y cuales no (sin color). Además, al pasar el ratón por encima nos saldría en una etiqueta el nombre del problema y si hacemos clic, accederíamos al detalle del problema.

- **Área de ranking de problemas:** en esta área, añadimos una tabla con un ranking de los problemas, donde se muestran los diez problemas que más aceptados tienen frente a los envíos. Esta tabla se compone del nombre del problema y el id, el último veredicto obtenido, el lenguaje en el que se resuelto, aceptados frente a envíos, aceptados frente a usuarios y la fecha del último envío.

- **Área de ranking de alumnos:** se muestra una tabla con el ranking de alumnos que más problemas diferentes han resuelto. Aparece el nombre, apellido y email del usuario, el número de aceptados frente a los envíos, el número de aceptados frente a los fallidos y la fecha del último envío.

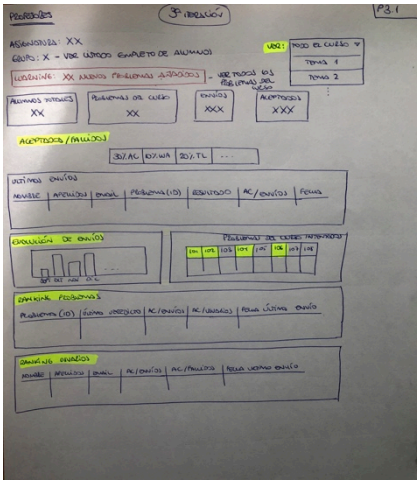


Figura 27: Vista previa de la página del profesor

En la siguiente imagen, véase la Figura 28, se muestra el listado de problemas y el listado de alumnos.

La página del listado de problemas es una página nueva que añadimos en esta iteración. Se compone de dos áreas:

- **Área de selección de la vista de datos:** en esta área nos vamos a encontrar un desplegable donde el profesor va a poder elegir ver el listado de problemas del curso completo o por temas.
- **Área de problemas:** esta área se compone de una tabla con el listado de problemas perteneciente a la elección del desplegable anterior donde se muestra: el nombre del problema junto con el id, los intentos frente a los usuarios, los aceptados únicos, los envíos de ese problema (envíos totales), los aceptados frente a los envíos totales y los aceptados frente a los usuarios únicos.

La página referente al listado de alumnos se había eliminado en la segunda iteración, pero se ha vuelto a añadir ya que es interesante presentar la información relativa al grupo de alumnos. Se compone de un solo área:

- **Área de alumnos:** esta zona, se compone de una tabla con el nombre, apellido y email del alumno, así como los aceptados frente a los envíos, los fallidos frente a los envíos y la fecha del último envío.

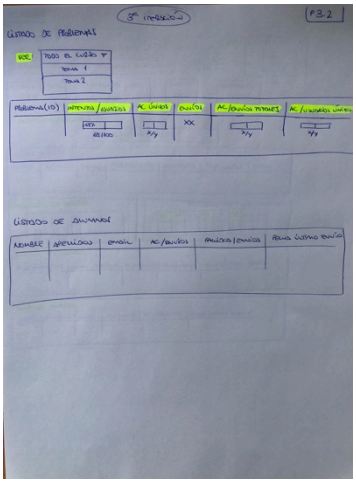


Figura 28: Vista previa de la página del listado completo de problemas y del listado de alumnos

En esta tercera imagen, véase la Figura 29, se aprecia la vista del detalle del usuario/ alumno. En esta página identificamos seis áreas:

- **Área descriptiva:** se muestran los datos del usuario: nombre, apellidos, email, que ya estaban, y añadimos: fecha de registro, país y la institución a la que pertenece.
- **Área de aceptados/fallidos:** en la segunda área añadimos una barra con los aceptados frente a los fallidos en porcentajes. Los fallidos además están divididos en los diferentes tipos de errores.
- **Área últimos envíos:** esta tabla muestra los últimos envíos de problemas, donde aparece el nombre del problema con el id, el resultado, los aceptados frente a los envíos y la fecha de envío.

- **Área de evolución de los envíos:** está dedicado a la evolución de los envíos, mediante un gráfico de barras en el cual el alumno/ usuario puede ver de un simple vistazo cuando ha hecho más o menos envíos durante el curso.
- **Área de problemas intentados:** está dedicado a los problemas del curso. Mediante una tabla formadas por los ids de los problemas vemos que problemas se han intentado y han sido aceptados (color verde), cuales se han intentado y han sido fallidos (color rojo) y cuales no (sin color). Además, al pasar el ratón por encima nos saldría en un tooltip el nombre del problema y si hacemos clic, accederíamos al detalle del problema.
- **Área de ranking de problemas:** En el tercer espacio, nos encontramos una tabla con el ranking de problemas, donde aparecen los problemas que más aceptados frente a envíos han obtenido. En la tabla aparece el nombre del problema junto con el id, el último veredicto obtenido en ese problema, los aceptados frente a los envíos y la fecha del último envío.

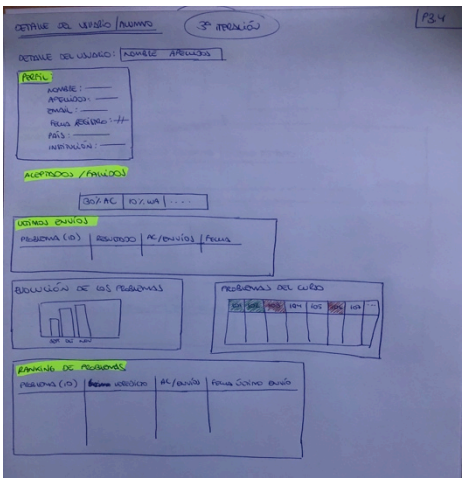


Figura 29: Vista previa de la página del detalle de un alumno/usuario

En esta cuarta imagen, véase la Figura 30, se muestra la página del detalle de un problema. Esta página se compone de cinco áreas:

- **Área de totales:** se mantiene igual que en la segunda iteración.
- **Área de selección de la vista de datos:** añadimos un desplegable donde se puede elegir si los datos que se quieren ver sean del curso completo o por temas.

- **Área últimos envíos:** que muestra los últimos envíos referentes a ese problema donde aparece el nombre, los apellidos y el email, del usuario que ha hecho el envío, el resultado, los aceptados frente a los envíos y la fecha del envío.
- **Área de la evolución de los envíos:** el diagrama de barras muestra la evolución de los envíos a lo largo del curso.
- **Área ranking de alumnos:** esta tabla muestra el ranking de alumnos que más problemas diferentes han resuelto y donde aparece el nombre, apellido y email del usuario, el número de aceptados frente a los envíos, el número de aceptados frente a los fallidos y la fecha del último envío.

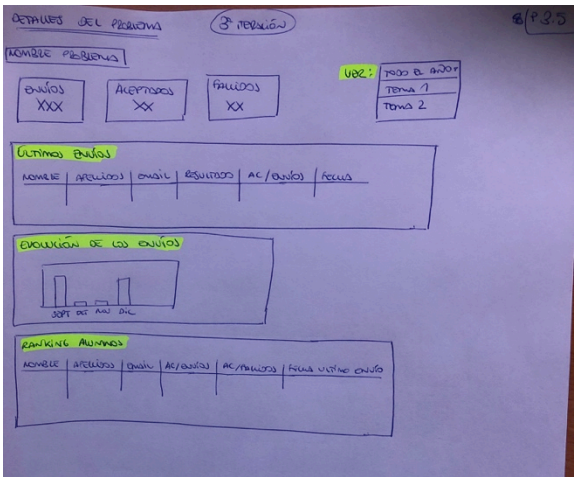


Figura 30: Vista previa de la página del detalle del problema

Capítulo 4 - Dashboard para jueces en línea

En este capítulo vamos a ver el aspecto de la aplicación después de su implementación y explicaremos los puntos clave de la misma.

Inicialmente vamos a hacer una introducción a los elementos usados en cada una de las páginas del panel de control y luego explicaremos cómo se organizan esas páginas dentro del proyecto creado.

4.1 Elementos usados en el dashboard

Para el diseño de esta aplicación, hemos usado una serie de elementos comunes en todas las vistas y que se usan normalmente en el diseño de paneles de control. Estos elementos los hemos explicado de una manera concreta en el Capítulo 2 - Jueces en línea y dashboards y ahora explicaremos para que los hemos usado.

Los objetos principales que vamos a visualizar son:

- Iconos: alertas (Figura 31).
- Cajas de totales (Figura 32).
- Gráficos: gráfico de sectores y donut (Figura 33), gráfico de barras apiladas (Figura 34) y gráfico lineal (Figura 35).
- Organizadores: mapa espacial (Figura 36) y tablas (Figura 37).

4.1.1 Área de alertas

En la página principal tanto del administrador como del profesor, vamos a tener una serie de iconos de tipo alerta. Estas alertas si son errores van a ir en color rojo y si son avisos en color amarillo:

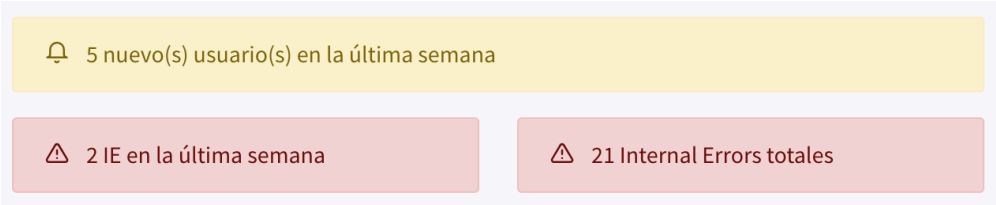


Figura 31: Diferentes iconos de alertas en la vista general del administrador.

4.1.2 Área de totales

En la página principal del administrador, del profesor y del detalle del problema, también nos vamos a encontrar una serie de cajas con totales.

Estas cajas nos muestran datos totales relevantes tanto para los profesores como para los administradores.

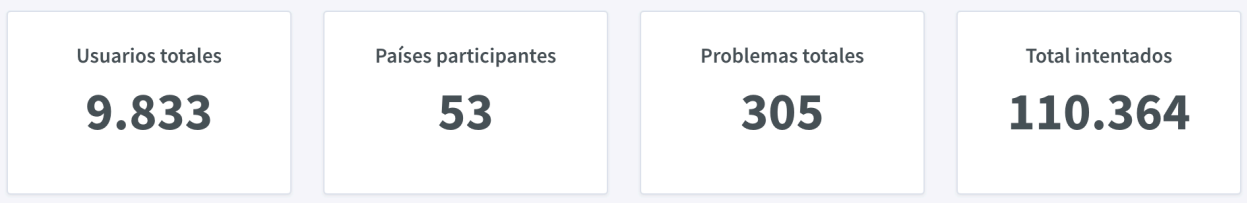


Figura 32: Diferentes cajas de totales en la vista general del administrador.

4.1.3 Gráficos

En este apartado vamos a hablar de los diferentes gráficos que hemos usado en la implementación de nuestra aplicación.

4.1.3.1 Área de lenguajes, Aceptados y Fallidos

En los paneles de control, son muy habituales los gráficos, por lo que al diseñar este panel de control los debíamos tener en cuenta.

En toda la aplicación, vamos a encontrar diferentes gráficos de diferentes tipos dependiendo la información que queramos mostrar.

En esta primera Figura 33, vemos dos gráficos por sectores donde mostramos tanto los porcentajes de lenguajes usados, como los porcentajes de los errores cometidos en los envíos de la resolución de los problemas. También vemos un gráfico de donut con el porcentaje de aceptados frente a fallidos.

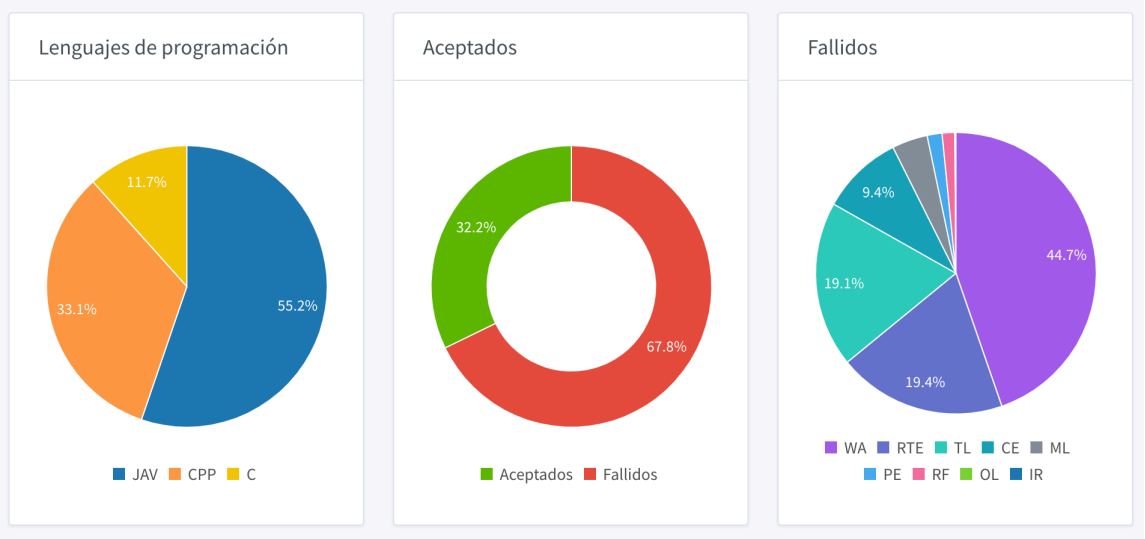


Figura 33: Gráficos por sectores y de donut en la vista del administrador.

4.1.3.2 Área de Aceptados/fallidos

En la Figura 34, sin embargo, apreciamos otro tipo de gráfico. En este caso usamos un gráfico de barras apiladas en horizontal para mostrar los números de aceptados y los tipos de errores obtenidos en la resolución de los problemas.

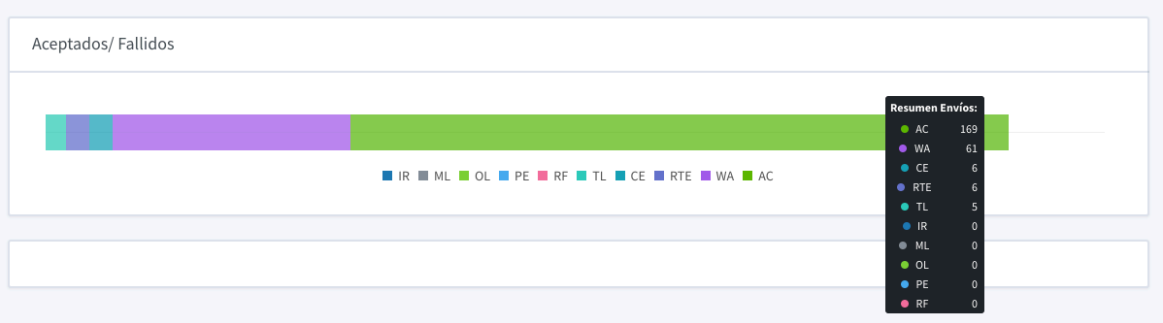


Figura 34: Barra apilada con aceptados y errores en la vista del detalle del usuario.

4.1.3.3 Área de evolución de los envíos

En la Figura 35, vemos un gráfico lineal donde se presenta la evolución de los envíos con los datos relativos a los enviados frente a los aceptados.

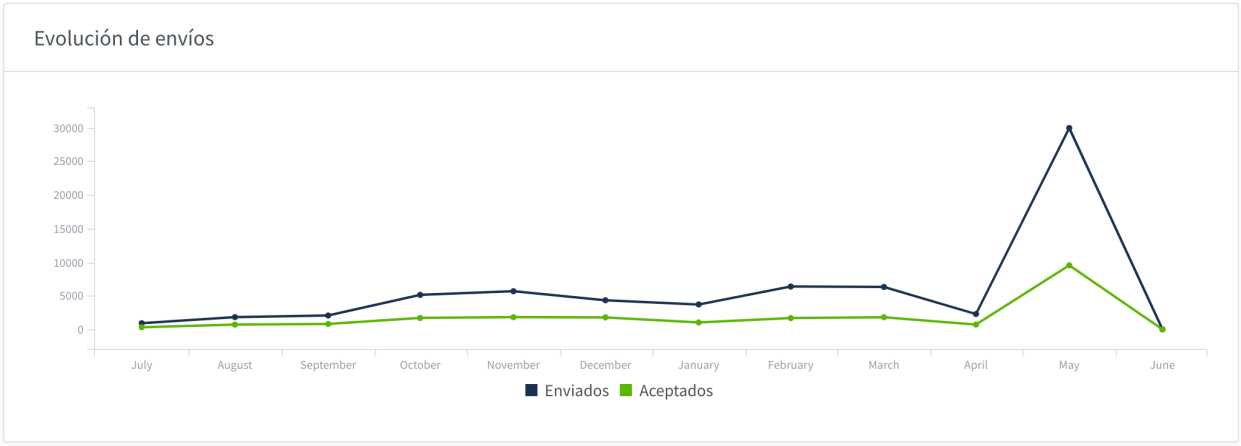


Figura 35: Gráfico de líneas con la evolución de envíos.

4.1.4 Organizadores

En nuestra implementación hemos incluido también los mapas espaciales y las tablas, ambos pertenecientes a los elementos del grupo de organizadores.

4.1.4.1 Área geográfica

En la página principal del administrador vamos a encontrar también un mapa del mundo. En la Figura 36 vemos el mapa espacial, donde se aprecian los países participantes en color azul y cuando pasemos el ratón por encima nos aparecerá el total de participantes de ese país y el porcentaje frente al total de usuarios.

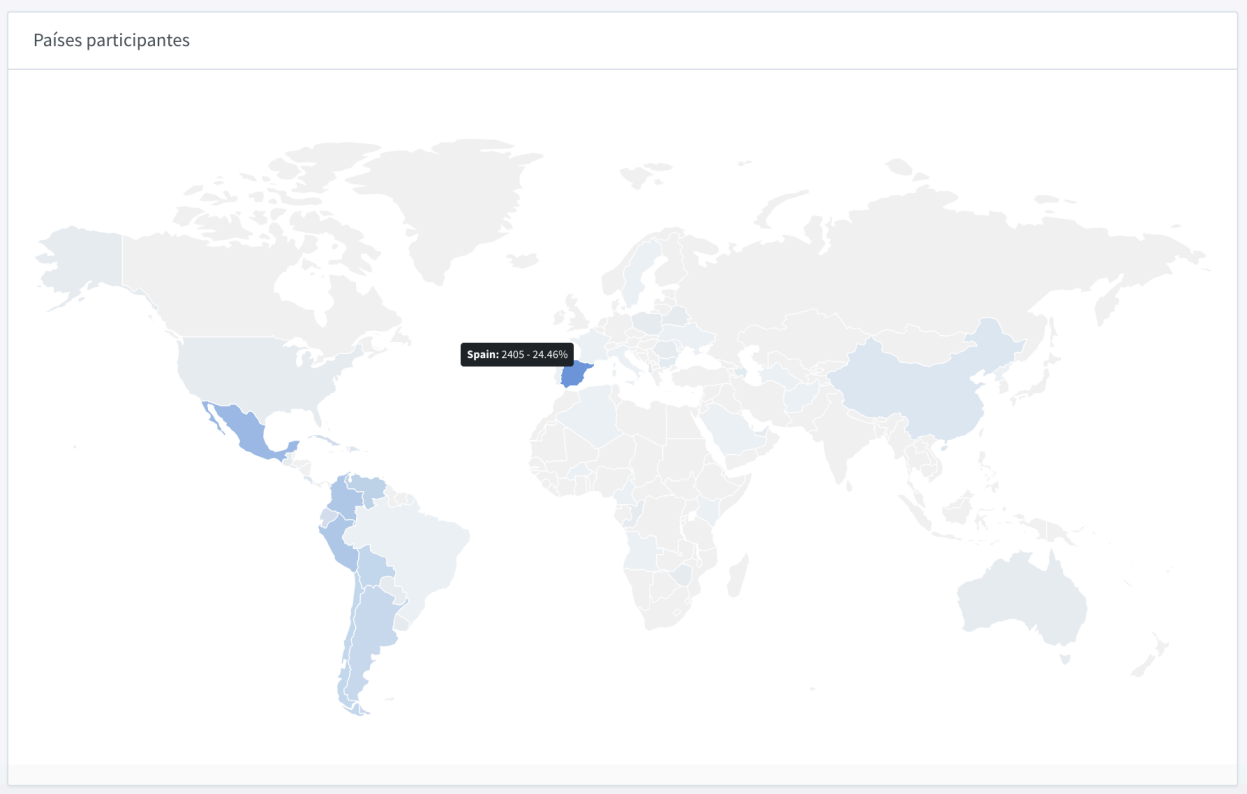


Figura 36: Mapa espacial que muestra los países participantes en la vista del administrador.

4.1.4.2 Área de últimos envíos, área de ranking de problemas y área de ranking de usuarios

Las tablas de datos las vamos a encontrar en todas las pantallas de la aplicación ya que es una buena forma de mostrar mucha información relevante que tiene relación entre sí.

Estas tablas también pertenecen al grupo de organizadores y siempre van a disponer de unos campos en la cabecera que irán cambiando, dependiendo de los datos que se muestren.

Dichas tablas van a poderse visualizar con más o menos registros, van a tener paginación e incluso se van a poder buscar palabras clave dentro de ellas mismas. También van a poderse ordenar por el campo de la cabecera que se desee en orden ascendente o descendente.

En la Figura 37, vemos la tabla de últimos envíos de un usuario.

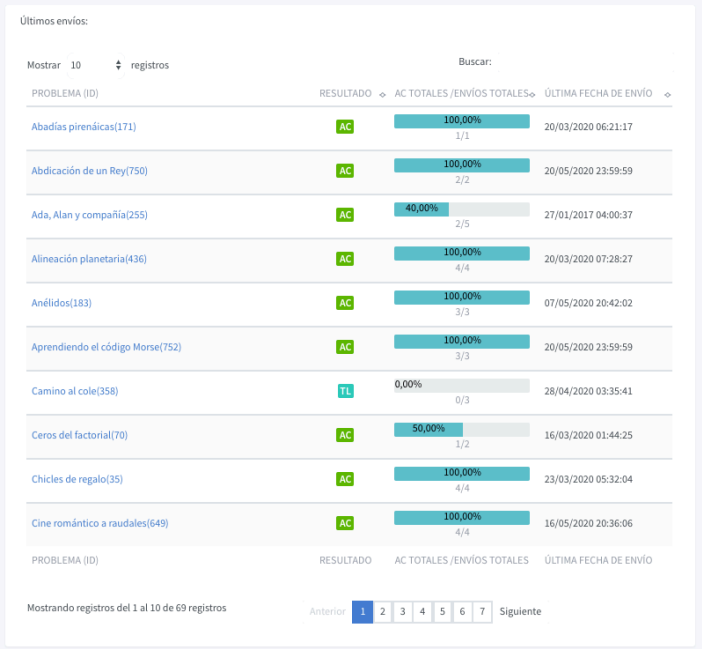


Figura 37: Área de últimos envíos perteneciente a la vista del detalle del usuario.

En la Figura 38, vemos el ranking de problemas y en la Figura 39 vemos el ranking de usuarios.

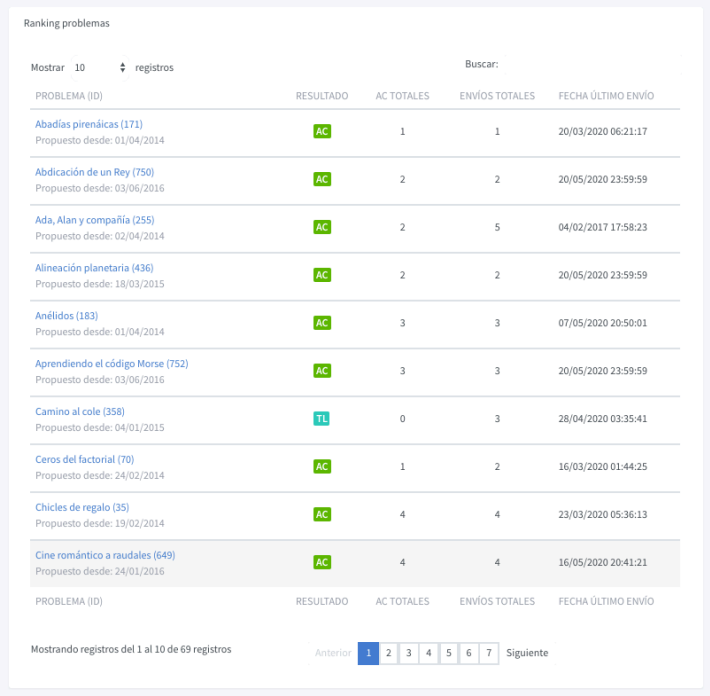


Figura 38: Área de ranking de problemas del detalle del usuario.

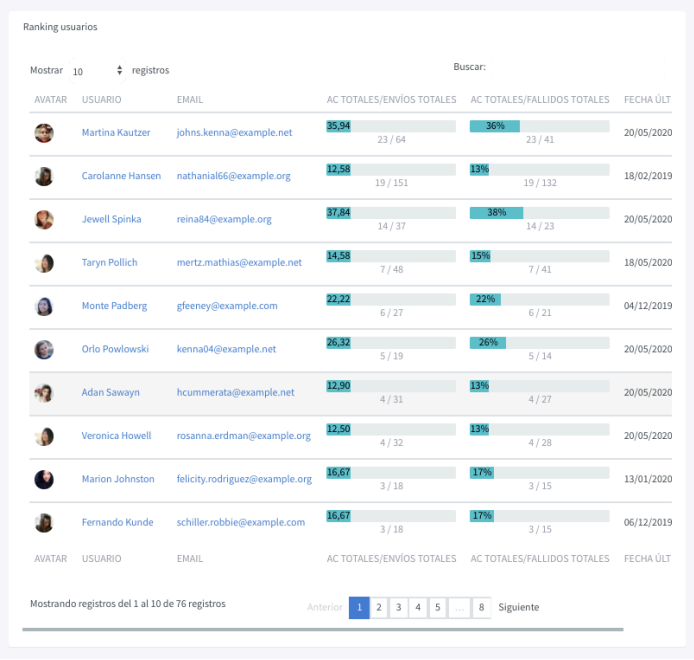


Figura 39: Área de ranking de usuarios del dashboard del profesor.

Con todos los elementos nombrados anteriormente, hemos ido construido todas las vistas de nuestra aplicación "*Dashboard para jueces en línea*".

4.2 Vistas

En este apartado, vamos a explicar cómo tenemos organizado nuestro Dashboard para jueces en línea.

Como en nuestro proyecto la autenticación de usuarios y roles no es el objetivo, no hemos creado el proceso de login a través del cual, se activaría el rol de administrador o el rol de profesor. Por ello, la primera vista que nos vamos a encontrar es una selección de rol. Una vez elegido el rol, se entraría sin login y se podría hacer las iteraciones del rol elegido.

Después de esa pantalla inicial, ya tendríamos cada una de las iteraciones de los roles: profesor o administrador, las cuales están asociadas a los diseños de la tercera iteración del Capítulo 3 - Diseño.

Ese diseño lo habíamos implementado en papel varias veces hasta poder llegar a las necesidades reales de los usuarios de forma que el panel de control resultase usable y sencillo para ellos.

A la hora de ir creando las diferentes páginas, todas debían tener una coherencia interna de forma que resultase sencillo visualizar los diferentes elementos en las páginas de forma rápida, por lo que las iteraciones para el administrador y el profesor van a ser casi idénticas.

Las vistas principales a las cuales accederemos serán:

- **Dashboard del administrador:** esta página será la principal de todas nuestras iteraciones con el rol de administrador.
- **Dashboard del profesor:** esta página será la principal de todas nuestras iteraciones con el rol de profesor.

Después de esas dos vistas principales, dispondremos de ocho vistas más derivadas de las principales. De las cuales, cuatro serán de las iteraciones del administrador y otras cuatro de las iteraciones del profesor:

Iteraciones del administrador:

- **Listado de problemas:** esta vista muestra el listado completo de problemas de *¡Acepta el reto!*
- **Listado de usuarios:** esta vista muestra todos los usuarios registrados en *¡Acepta el reto!*
- **Detalle del problema:** en esta vista tendremos el detalle del problema. Estará formado por diferentes datos relevantes sobre un problema concreto sobre todos los usuarios que lo han intentado resolver.
- **Detalle del usuario:** en esta vista tendremos el detalle de un usuario concreto con diferentes elementos para mostrar datos que sean relevantes para el administrador.

Iteraciones del profesor:

- **Listado de problemas del curso:** este listado es igual que la vista "Listado de problemas" del administrador, pero, el listado de problemas serán los elegidos por el profesor para resolver durante el curso.
- **Listado de alumnos del curso:** este listado es igual al "Listado de usuarios" del administrador, pero, el listado será de los alumnos del profesor.
- **Detalle del problema del curso:** este detalle es igual que el "Detalle del problema" del administrador, pero, los datos mostrados serán en relación al grupo de alumnos.
- **Detalle del alumno del curso:** este detalle es igual que el "Detalle del usuario" para el administrador, pero, el profesor podrá ver diferentes datos relevantes de cada uno de sus alumnos de su grupo.

Desde cada una de estas últimas ocho vistas nombradas anteriormente, podremos volver siempre a la vista general del rol en el que estemos e incluso

podremos acceder a otras vistas. En la Figura 40, vemos todas las iteraciones posibles dentro de nuestra aplicación.

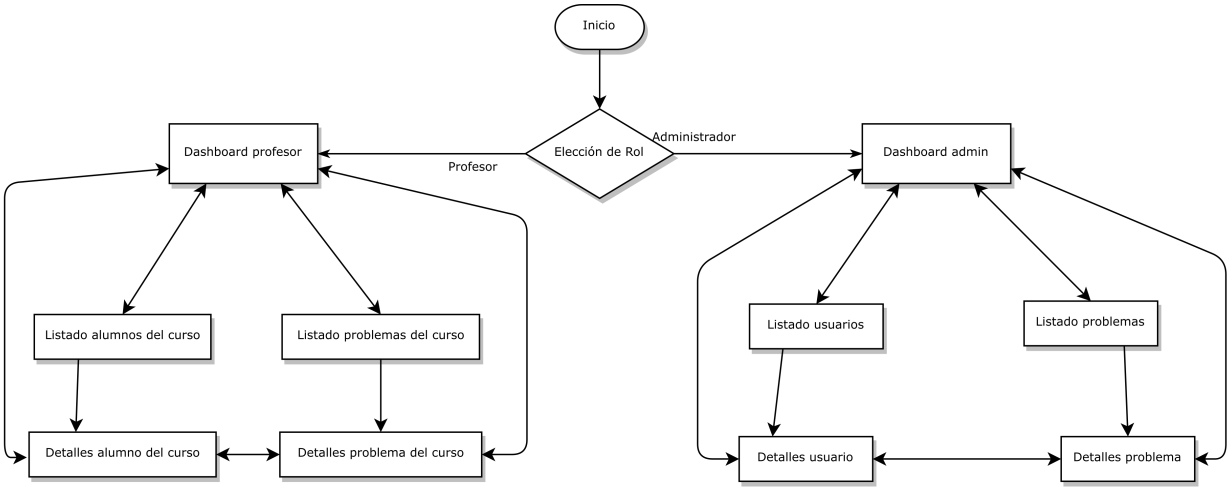


Figura 40: Organización de las vistas de Dashboard para jueces en línea.

Una vez que ya hemos visto la organización de vistas, vamos a ver como nos ha quedado el diseño final implementado del Dashboard del administrador de nuestra aplicación. Si observamos a simple vista, en las páginas consecutivas, nos vamos a encontrar una serie de figuras, por lo que vemos que la página del panel de control del administrador es extensa y su visualización requiere de scroll.

Según vimos anteriormente, Few (Few, 2006) definía cómo errores comunes al diseñar un panel de control el hecho de crear una pantalla muy larga en la cuál, no se pudiesen visualizar los datos de un simple vistazo, en nuestro caso, hemos cometido ese "error" si se puede llamar así porque hoy en día no se diseñan pantallas de forma monolítica: la cantidad de datos que hay que pintar en ellas, así como la enorme variedad de tamaños en los dispositivos que las pintan lo hacen completamente inviable. No tiene sentido condicionar la representación de datos en base a un dispositivo "modelo", ya que eso exigiría estar revisando constantemente qué tamaño es el considerado "estándar" cada poco tiempo.

A día de hoy, salvo que los datos tengan como objetivo rellenar pantallas informativas no interactivas (como los paneles en los aeropuertos), las interfaces deben ser fluidas y adaptables, sin restringir tamaños.

En esta primera Figura 41, observamos que existe la cabecera de nuestra aplicación y hay una división clara en la imagen ya que, nos encontramos dos columnas:

- La columna de la izquierda dispone de una serie de botones con el nombre de las áreas del panel de control creado. De esta forma, podemos acceder a los datos de una manera más rápida ya que, los botones, nos llevan al área seleccionado. Esta columna es fija y nos acompaña en el scroll de la columna derecha.

- La columna de la derecha dispone de lo que son en sí los elementos del panel de control creado. En este caso nos encontramos con el área de alertas: alertas informativas y de errores, el área de totales: cajas con números totales y gráficos de sectores y en forma de donut: porcentajes de lenguajes usados y porcentajes de resultados obtenidos en los envíos realizados.

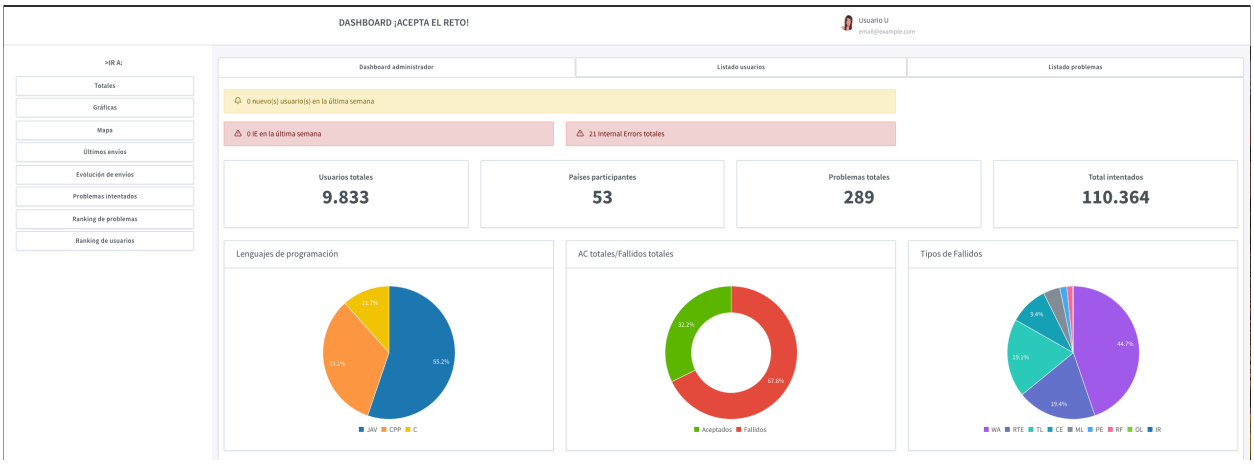


Figura 41: Vista del administrador de Dashboard para jueces en línea (1)

En la Figura 42, nos encontramos ahora en la columna de la derecha el mapa de los países participantes en el cual, cuando se pasa el ratón por encima, nos sale el

tooltip con la información acerca del país sobre el que estamos: porcentaje y número de participantes.

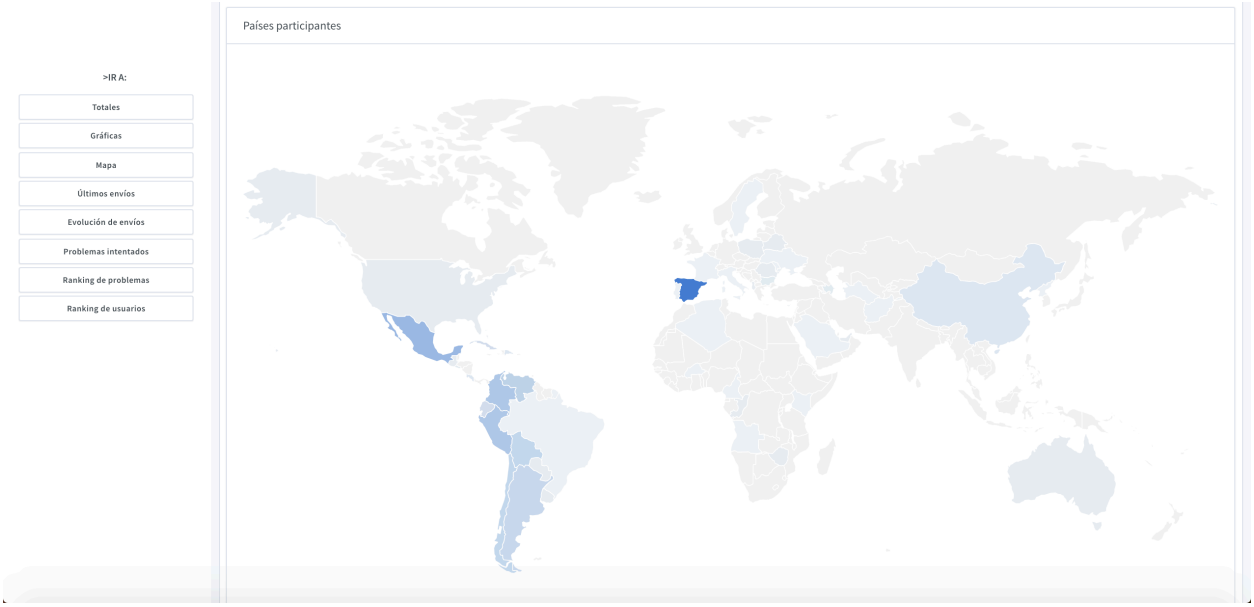


Figura 42: Vista del administrador de Dashboard para jueces en línea (2)

En la Figura 43, nos vamos a encontrar con el área de últimos envíos, en forma de tabla, y con la evolución de los envíos, en forma de gráfico lineal, con los envíos realizados en cada uno de los meses (mes actual menos doce meses) y los aceptados en cada uno de los meses.

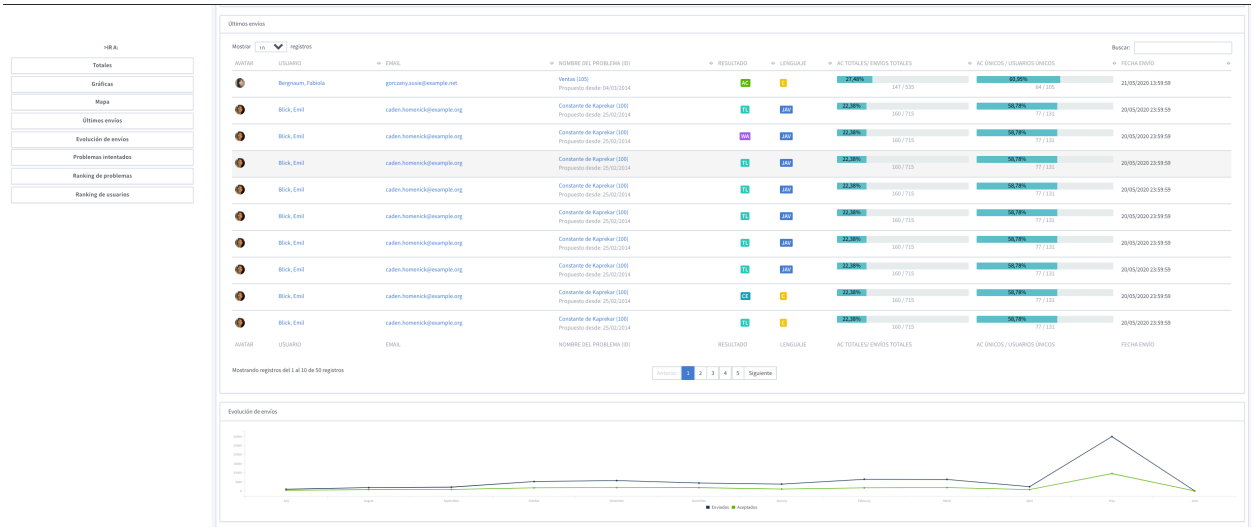


Figura 43: Vista del administrador de Dashboard para jueces en línea (3)

Continuamos con la Figura 44. En esta figura, nos encontramos con los problemas intentados por los usuarios y con el ranking de problemas el cuál también está pintado en forma de tabla.

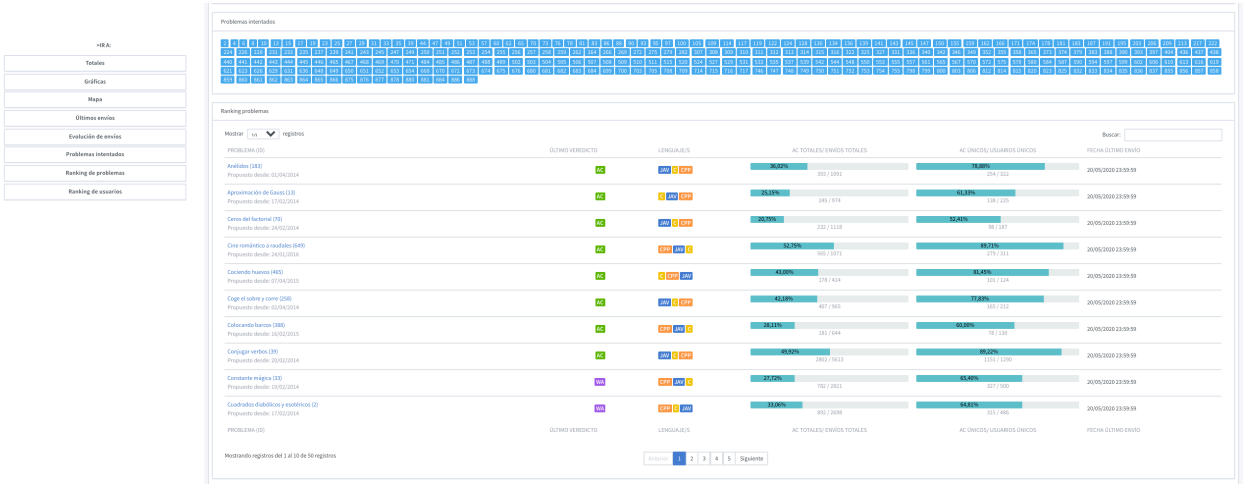


Figura 44: Vista del administrador de Dashboard para jueces en línea (4)

Finalmente acabamos esta vista con la Figura 45, donde vemos el ranking de los usuarios y el pie de la aplicación.

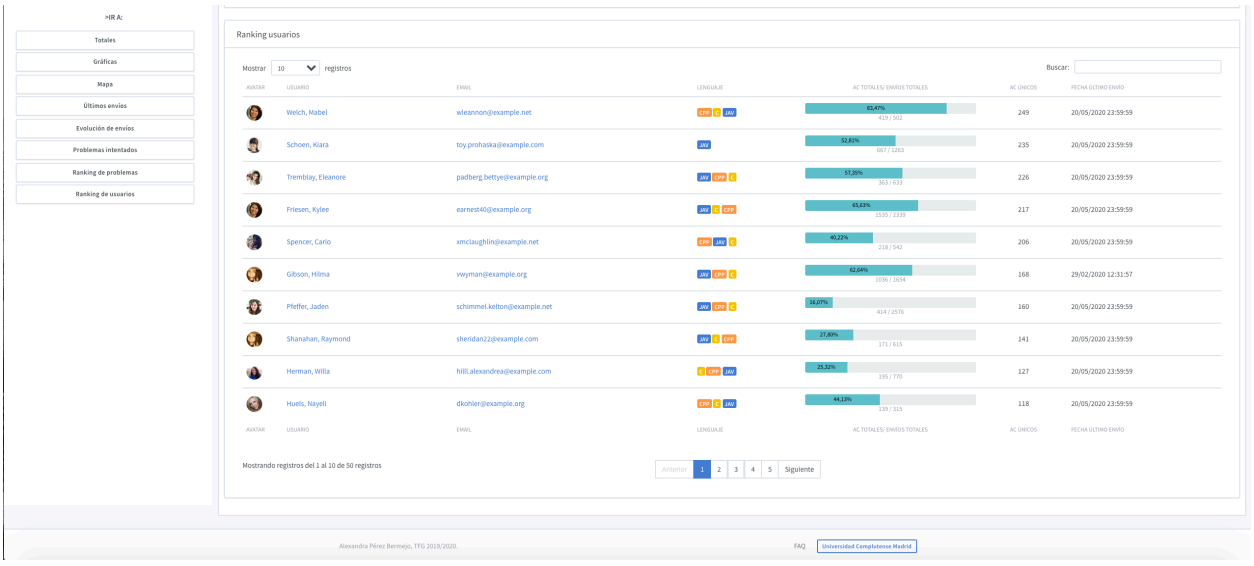


Figura 45: Vista del administrador de Dashboard para jueces en línea (5)

Capítulo 5 - Arquitectura e implementación

La aplicación web está creada con una arquitectura cliente-servidor y desarrollada con el framework de Codeigniter para la parte de servidor (backend) y con Tabler en la parte del cliente (frontend).

En este capítulo se va a explicar cómo funciona la arquitectura cliente-servidor y cómo hemos ido realizando la implementación del proyecto. También se expondrán las dificultades encontradas en el progreso de desarrollo.

5.1 Arquitectura

Esta aplicación se va a crear a través de una arquitectura cliente-servidor. El cliente va a realizar peticiones de servicios y el servidor es quien le va a dar las respuestas y por tanto se va a encargar de satisfacer dichas necesidades: el cliente le pide un recurso al servidor, y éste le da la respuesta (Figura 46).

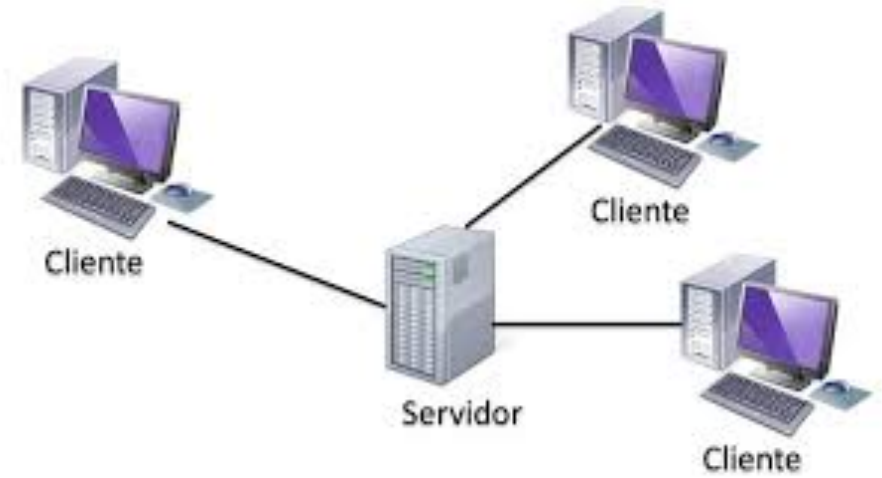


Figura 46: Arquitectura cliente-servidor.

5.1.1 Servidor

El modelo que se usa es el Modelo-Vista-Controlador (MVC) que es un patrón de arquitectura software que separa la lógica de control, la interfaz del usuario y los datos del sistema. De esta manera vamos a construir tres componentes diferentes: el modelo, la vista y el controlador, esto significa que por un lado vamos a tener la representación de la información y por otro la iteración con el usuario:

- **El modelo**: Esta capa trabaja con los datos que vamos a tener en la base de datos. Por ello sus funciones van a ser: acceder a la información y actualizar su estado.
- **La vista**: En esta capa, sin embargo, tendremos el código de la aplicación que va a permitir renderizar los estados de nuestra aplicación en HTML para que se visualicen las interfaces del usuario.
- **El controlador**: En esta capa, el código que encontraremos será el necesario para responder a las solicitudes de la aplicación. Va a servir de enlace entre la vista y el modelo.

En la Figura 47 perteneciente a Thomas Myer (Myer, 2008), se ve que modelo MVC tiene en cuenta que todo comienza en el usuario. El usuario es quien interactúa con el controlador. El controlador manipula el modelo y el modelo actualiza la vista que se muestra al usuario.

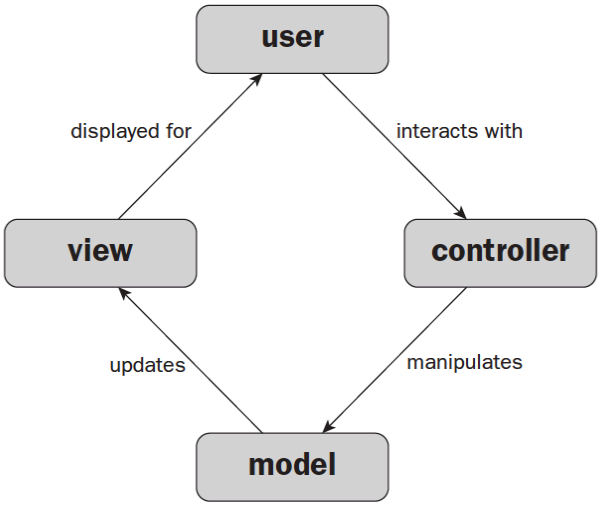


Figura 47: Modelo vista controlador.

Los proyectos creados con este modelo MVC, disponen de la mayoría del trabajo en el controlador. En ese elemento va a ser donde se carguen las bibliotecas, se extraen los datos del modelo y se meten los datos del modelo en las vistas. Todo está a la vista por lo que es fácil tener todo organizado y ver los problemas cuando ocurran. La única pega es que cuando los principiantes crean controladores son muy básicos y a medida que pasa el tiempo son complicados de mantener.

5.1.1.1 Modelos

En los modelos se implementan todas las *consultas a la base de datos* que vamos a necesitar para mostrar en las vistas y esos modelos los cargan los controladores. Los modelos, además van a manejar la persistencia a la base de datos.

La base de datos que vamos a usar es la proporcionada por los tutores. Esta base de datos ha sido reestructurada ya que había varias tablas necesarias para nuestra aplicación que no teníamos y por tanto hemos tenido que crear.

En la Figura 48, vemos el diagrama inicial de la base de datos que fue proporcionada al principio:

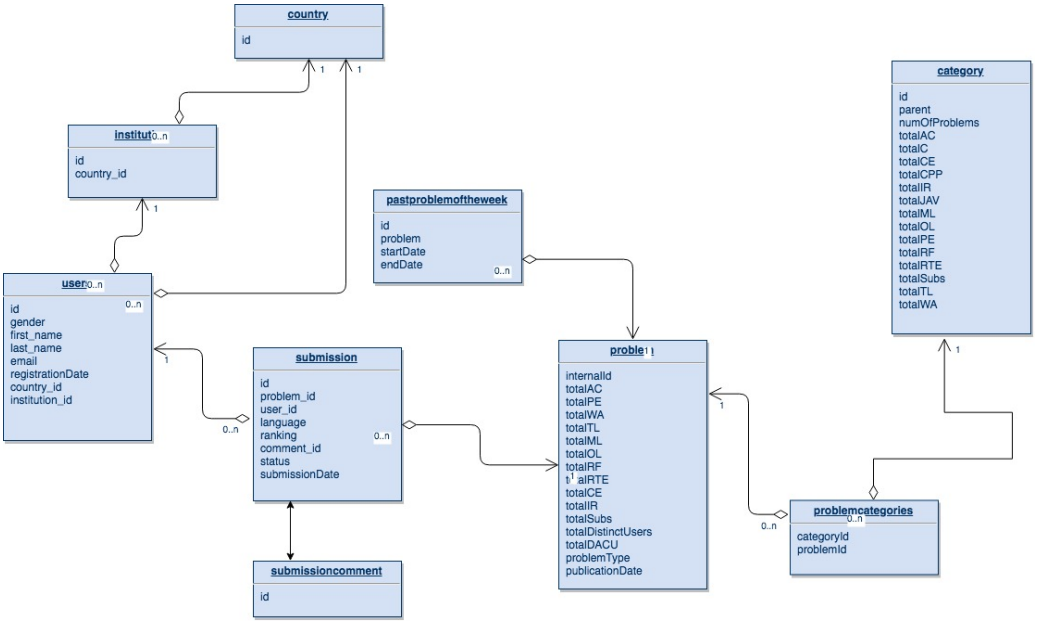


Figura 48: Modelo relacional de la base de datos proporcionada

Después de la reestructuración, nuestra base de datos va a ser la que vemos en la Figura 49:

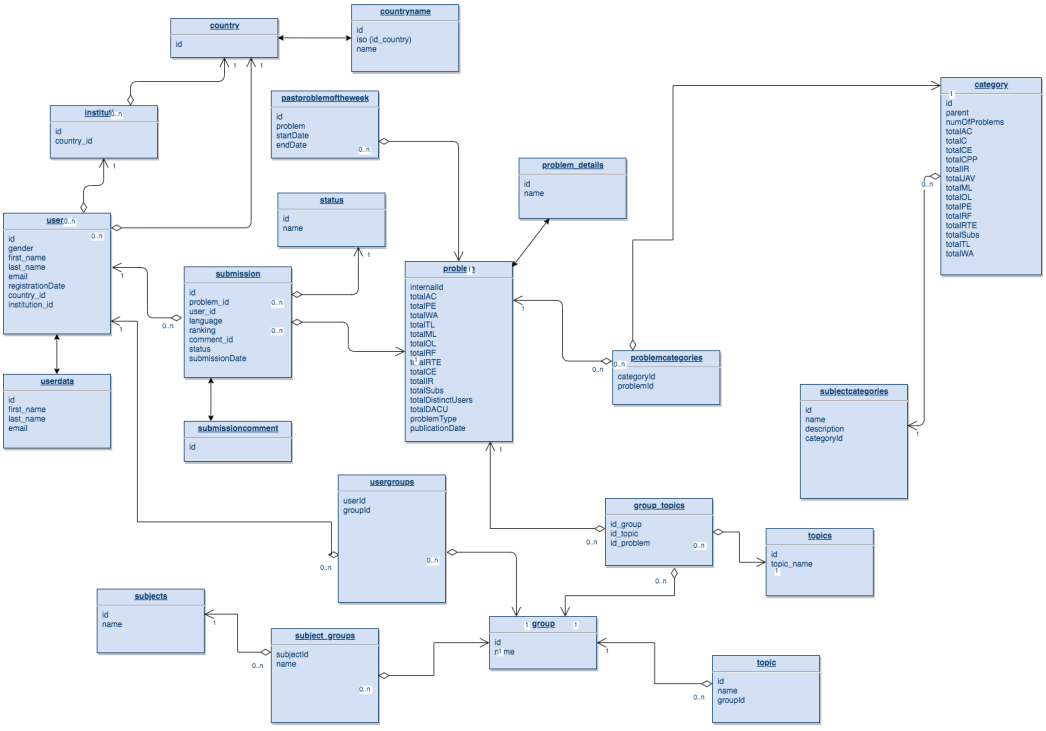


Figura 49: Modelo relacional de la base de datos final.

En nuestro proyecto, vamos a disponer de cuatro modelos.

- **Mcountries:** Dispone de todas las consultas a la base de datos relativas a los países.
- **Mproblems:** Dispone de todas las consultas a la base de datos relativas a los problemas.
- **Msubjects:** Dispone de todas las consultas a la base de datos relativas a las asignaturas.
- **Musers:** Dispone de todas las consultas a la base de datos relativas a los usuarios.

5.1.1.2 Vistas

Las vistas se cargan con los controladores completas o por sectores para conseguir una vista final. El diseño está controlado por CSS y el formato de las plantillas es puro HTML con marcado PHP, por lo que es muy sencillo crear vistas y optimizarlas.

La lógica va a ir toda en los modelos y en los controladores, por lo que estarán más localizados los errores cuando los haya.

Vamos a disponer de las siguientes vistas:

- **Template-admin-dashboard:** Vista del panel de control general del administrador.
- **Template-all-problem-submissions-by-user-table:** Vista de la tabla de todos los envíos por usuario.
- **Template-all-problem-submissions-table:** Vista de la tabla de todos los envíos.
- **Template-all-problems-table:** Vista del panel de control del listado completo de problemas.
- **Template-all-users-table:** Vista del panel de control del listado completo de usuarios/alumnos.
- **Template-base:** Vista base.
- **Template-main:** Vista principal.
- **Template-problem-details:** Vista del panel de control del detalle del problema.
- **Template-select-role:** Vista de la selección del panel de control de los administradores o de los profesores.
- **Template-teacher-dashboard:** Vista del panel de control general del profesor.
- **Template-user-details:** Vista del panel de control del detalle del usuario/alumno.

5.1.1.3 Controlador

Los controladores son los pilares de la aplicación. Cualquier cosa que un usuario puede hacer en la aplicación, incluso las rutas de destino se representan aquí.

Los controladores únicamente inicializan las funciones y cada función o método tiene su propio mapa de destino en la aplicación. Por ejemplo, si un usuario quiere ir a la ruta */page/foo*, tiene que tener un controlador que sea *Page* y dentro un método *foo()*.

Disponemos de los siguientes controladores:

- **Admindashboard:** Controlador con las llamadas a todos los métodos de la vista del panel de control del administrador.
- **Problem:** Controlador con las llamadas relativas a todo lo relacionado con los problemas.
- **Teacherdashboard:** Controlador con las llamadas a todos los métodos de la vista del panel de control del profesor.
- **Users:** Controlador con las llamadas relativas a todo lo relacionado a los usuarios.
- **Welcome:** Controlador con la llamada inicial.

5.1.2 Cliente

El cliente va a acceder al servidor a través de una petición HTTP (*HyperText Transfer protoc*ol) – el protocolo de transferencia de hipertexto es el que nos permite las transferencias de información en la WWW (*World Wide Web*) -, que en este caso, implementa el protocolo/certificado de seguridad SSL (*Secure Sockets Layer*) - la capa de puertos seguros, es la que nos va a asegurar una comunicación segura a través de internet - .

El servidor atiende la petición y devuelve una respuesta que está en formato HTML (*HyperText Markup Language*), que va a ser el lenguaje de marcada para la elaboración de páginas web y que el navegador se va a encargar de renderizar.

5.1.3 Infraestructura

La infraestructura de este proyecto es de tipo LAMP (Linux, Apache, MySQL y PHP), ya que es la que mayor compatibilidad tiene para su despliegue.

Los cuatro componentes nombrados anteriormente forman la infraestructura en el servidor para que sea posible la creación y alojamiento de páginas web dinámicas.

El funcionamiento es sencillo. Linux es el sistema operativo base para llevar a cabo el servidor web Apache, el cual no puede interpretar contenidos dinámicos, pero para ello PHP es el que, del lado del servidor, realiza sus funciones de programación.

En cuanto a los requisitos mínimos del sistema son aquellos que dictamina el framework sobre el que se basa todo el aplicativo, en este caso, CodeIgniter (Codeigniter):

- Servidor Apache.
- PHP versión 5.6, aunque se recomienda una versión más reciente.
- MySQL (5.1+) a través de los drivers mysqli y pdo.

5.2 Implementación

A continuación, se van a explicar los frameworks que se han usado para la implementación de este proyecto tanto en la parte del backend, como en la parte del frontend y cómo se han unido.

5.2.1 Implementación del servidor (backend)

En el inicio, para el framework base, se estuvieron investigando varios paquetes para ver cuál podía ser el más interesante a la hora de implementar nuestro proyecto. Por ello, se buscaron varios frameworks de backend con base en PHP y se indagaron:

- **Laravel** (Laravel): "El framework PHP para artesanos". De esta manera, anuncian su framework en la web. Este framework fue creado en 2011 y es de código abierto. Se usa para desarrollar aplicaciones y servicios web con PHP5 y PHP7. Su objetivo es ser un framework que permita una sintaxis elegante y expresiva para crear código de forma elegante y simple permitiendo muchas funcionalidades. Muchas de las dependencias que lo forman son de Symfony.
- **Symfony** (Symfony): "Es un conjunto de Componentes PHP, un framework de Aplicaciones Web, una Filosofía y una Comunidad – todo trabajando en armonía". Así es cómo se presentan en su web. Este framework se creó en 2005. Esta basado en Modelo Vista Controlador (MVC), que se explicará más adelante, y fue diseñado para desarrollar aplicaciones web.
- **Codeigniter** (Codeigniter): Su presentación en la web es la siguiente: "Es un poderoso framework PHP construido para desarrolladores que necesitan un conjunto de herramientas simple y elegante para crear aplicaciones web con todas sus funciones". Es el framework más rápido y popular a día de hoy ya que es muy liviano y está construido para desarrolladores que necesiten un conjunto de herramientas simple y elegante para crear unas completas aplicaciones web.

Los tres frameworks, son de código abierto y utilizan el Modelo Vista Controlador (MVC) para desarrollar aplicaciones en php de manera más rápida.

Además de estos beneficios comunes, cada uno de ellos tiene sus ventajas propias y sus desventajas. En la Tabla 5, podemos verlas:

Framework	Ventajas	Desventajas
Laravel	• Arquitectura limpia • Compositor amigable • Comunidad en crecimiento	• Muy lento • Muchas dependencias • Muchos métodos estáticos • Es muy pesado
Symfony	• Inyección de dependencias • Gran comunidad	• Muchas dependencias • Muchos archivos de configuración
Codeigniter	• Configuración sencilla • Gran soporte de la comunidad • Bien documentado	• No tiene ORM (Object Relational Mapping) • No tiene CLI (Command Line Interface)

Tabla 5: Ventajas y desventajas de Laravel, Symfony y Codeigniter.

Entre los tres frameworks mencionados anteriormente, Laravel fue descartado al resultar innecesariamente más complejo, exigir unos requisitos de infraestructura mayores y ser el más lento de todos. Symfony también se descarta por tener una configuración con muchos archivos y tener muchas dependencias.

Codeigniter, tal y como reza su web oficial, es un potente framework de PHP muy liviano, construido para desarrolladores que necesitan un kit de herramientas simple y elegante, capaz de crear aplicaciones web completas basadas en un alto rendimiento y escalabilidad.

Con menos de 2MB, el sistema se caracteriza por:

- Un sistema de tipo MVC tradicional completamente modular y escalable.
- Alto rendimiento debido a su ligereza y estructura base. Las páginas se procesan más rápido y tiene fácil edición y creación.
- Alta seguridad, con protección activa contra ataques CSRF y XSS.

- Mínima configuración: Solo hay que subir los archivos al ftp y editar un archivo de configuración para definir el acceso a la base de datos.
- Amplia comunidad que ofrecen soporte y herramientas adicionales (plugins) con las que complementar el proyecto base.
- Open Source. Todo el sistema es código abierto licenciado bajo MIT (Massachusetts Institute of Technology).

NOTA: Para conocer los detalles sobre la Instalación del proyecto, consúltese el Apéndice D - Instalación

5.2.2 Implementación del cliente (frontend)

Para la visualización del proyecto, se ha escogido el proyecto de código abierto Tabler (Tabler). La elección de esta plataforma se ha realizado en base a sus características más destacables:

- Open Source: Todo el sistema es de código abierto licenciado bajo MIT.
- Diseño adaptable (responsive) a través de estructuras modernas.
- Componentes UI elegantes y de acuerdo a las corrientes estéticas más actuales.
- Uso de herramientas internas ampliamente extendidas y documentadas en el ámbito frontend.
- Multinavegador: Soporte completo para las versiones actuales de Chrome, Firefox, Safari, Opera, Internet Explorer 10+, y una amplia variedad de navegadores para dispositivos móviles.
- HTML5 y CSS3.

Esta herramienta, como se ha comentado, recurre además a bibliotecas de terceros de amplia difusión y excelentemente documentadas. Las más relevantes se citan a continuación:

- Bootstrap 4 (Bootstrap): Versión más reciente del framework HTML, CSS y Javascript más utilizado del mundo.
- jQuery (JQuery): La biblioteca Javascript más utilizada a lo largo de la web.

- SASS (Sass): Herramienta de pre procesado CSS más extendida.

Otras dependencias más específicas son:

- Sparkline (Sparkline): Biblioteca para la elaboración de gráficos.
- Selectize (Selectize): Componente UI para la creación de elementos seleccionables HTML.
- Tablesorter (Tablesorter): Herramienta para la ordenación de datos tabulados.
- Vector Map (Vectormap): Visualización de mapas vectoriales interactivos.
- Font Awesome (Font Awesome): Paquete con iconos. Existen una enorme cantidad de soluciones similares para la gestión de un proyecto de estas características.

Ante la misma, uno de los puntos que me decantó la elección, es la excelente aceptación con la que Tabler disfruta en GitHub, alcanzando a día de hoy más de 15.000 valoraciones positivas y más de 1.000 replicados (forks).

5.2.3 Unión servidor y cliente (backend y frontend)

Para unir ambas caras del desarrollo, backend y frontend, se ha optado por una biblioteca de motor de plantillas HTML en PHP moderna y flexible como es Twig (Twig).

Twig es un motor de plantillas en PHP cuya finalidad es facilitar el trabajo a los desarrolladores ya que, es un sistema muy fácil de aprender y genera un código conciso y de fácil lectura. Su licencia de distribución es BSD (Berkeley Software Distribution).

Esta elección se ha basado en sus tres características principales:

- Rendimiento y velocidad: Las plantillas se compilan directamente en un código PHP plano y optimizado.
- Seguridad: El sistema interpola variables de un modo completamente seguro, además de incorporar un modo 'sandbox' que pre evalúa el código de las plantillas antes de su compilación.
- Flexibilidad.

5.3 Herramientas de trabajo

Para realizar el trabajo, se preparó un entorno local basado en el mismo paradigma LAMP que requiere la aplicación final. Dicho entorno se confió en una herramienta preconfigurada (XAMPP) que levanta todos los servicios necesarios para tal fin: Servidor Apache, MySQL y PHP en sus dos ramas más frecuentes: 5.3 y 7.2. Además, para comprobar el correcto funcionamiento del sistema a medida que se avanzaba el desarrollo, se utilizó un servidor real al que, periódicamente, se fue desplegando el código final para ver su impacto en un entorno real de producción.

Además de lo ya citado, se utilizó el IDE PHPStorm desarrollado por JetBrains para la codificación del aplicativo. Este software de edición de código facilita tanto el desarrollo como las pruebas, además como no, del control de versiones y de los despliegues en el servidor remoto.

Para el control de versiones se ha utilizado en primera instancia GIT, repartiendo el desarrollo en diversas ramas organizadas según funcionalidades que periódicamente se han ido volcando a la rama maestra. Al tratarse de un trabajo individual, esta metodología se ha llevado a cabo más como respaldo de lo ya realizado que como herramienta colaborativa. Finalmente, para alojar el historial de cambios se ha escogido la plataforma GitHub dada su indudable calidad contrastada, las herramientas que facilita y sus planes gratuitos recientemente ampliados. El proyecto completo se puede encontrar en: https://github.com/al3xMad/Dashboard

Para las entrevistas de los profesores y la evaluación con los usuarios, se utilizó Google Forms, ya que resultaba una forma sencilla de entrevistarlos o que evaluasen la aplicación en el momento más adecuado para ellos ya que con esta herramienta, podrían acceder y contestar en cualquier momento.

5.4 Despliegue

La aplicación web se ha montado en un Servidor Cloud que es una infraestructura creada a partir de la división de un servidor físico en múltiples servidores virtuales.

El servidor que hemos usado tiene las siguientes características:

- Sistema operativo: CentOS 7 x64 Plesk 12 (Linux)
- 6 cores AMD
- 5GB de RAM
- 2,1 GHz por núcleo
- Velocidades de transferencia garantizadas: 50Mbits

Además, hemos comprado el dominio: onlinejudgedashboard.es que es donde tendremos alojada nuestra aplicación.

La transferencia de archivos se hace por SFTP (Secure File Transfer Protocol o Protocolo Seguro de Transferencia de Archivos), de esta manera nuestra transferencia y manipulación de los datos, se hace sobre un flujo de datos fiable y para la gestión del servidor hemos usado Plesk de Parallels (Parallels) que es un software de administración del servidor muy completo, fácil de usar y de los más estables y seguros del mercado.

Nuestra aplicación se ha implementado en local y los cambios se desplegarán siempre que lo deseemos, tanto a Github: http://github.com/al3xMad/Dashboard como a nuestra web: http://onlinejudgedashboard.es/, teniendo siempre las últimas versiones actualizadas en ambos sitios.

La sincronización de subida la hemos hecho desde PHPStorm que es nuestra herramienta de desarrollo y nos permite ambas sincronizaciones sin necesidad de usar más herramientas.

5.5 Complicaciones sufridas durante la implementación

A la hora de implementar mi aplicación, he ido encontrando diversas dificultades que he tenido que ir afrontando.

Al inicio, las preguntas de las entrevistas se rehicieron varias veces ya que muchas de ellas eran preguntas dirigidas y condicionaban al sujeto entrevistado y también había preguntas de respuesta binaria en vez de hacer las preguntas abiertas.

A la hora del diseño de la aplicación, se comenzó por dibujar una primera iteración la cuál apenas tenía sentido ya que no era intuitiva para los usuarios, dependiendo de la página en la que te encontrases, los elementos dispuestos siempre cambiaban de sitio y sus características eran diferentes. Por ello, se tuvieron que ir rediseñando varias veces cada una de las páginas hasta que finalmente había una coherencia interna entre ellas. Si se hubiese hecho un análisis inicial correcto, esto no nos habría pasado.

Con respecto a la implementación, me encontré con diversos problemas iniciales que tuve que ir solucionando para avanzar en el desarrollo. El más complejo fue el relativo a la consistencia de los datos sobre los alumnos y problemas que se me facilitó como fuente. Entendiendo que había datos de carácter personal que se omitieron deliberadamente, y que varias de las tablas no resultaban consistentes entre sí, hubo que hacer un trabajo de reconstrucción y ampliación en aquellos campos que iban resultando necesarios para elaborar los componentes.

Además de este trabajo de minería de datos, y siempre buscando que la aplicación resultara lo más amigable posible, se recurrió a rellenar todos los campos vacíos utilizando para ellos herramientas automatizadas orientadas al terreno de los mocks: Generación de thumbnails, datos ficticios de carácter personal, etc...

Ya desde una perspectiva más técnica, algunas de las herramientas utilizadas (CodeIgniter, Twig, ...) requirieron de un importante estudio documental para lograr integrarlas de forma coherente y consistente entre ellas. Hay que remarcar en este

sentido que la diferencia temporal entre varias de ellas requirió actualizar diversas partes consideradas nucleares de los mismos para su correcto funcionamiento. Tal fue el caso por ejemplo del motor de plantillas Twig dentro de la arquitectura de CodeIgniter. Ya, en las tareas de maquetación, hubo que adaptar gráficas y contenedores a los diseños previos, modificando para ello el comportamiento natural de algunas herramientas gráficas como fue el caso de los mapas vectoriales, o los diferentes sistemas de barras y líneas.

Finalmente, cabe destacar como el paso de un entorno local a uno de producción implica siempre reajustes de última hora que tratan de limar los posibles problemas derivados de las versiones con las que cuente la máquina, o máquinas, finales, así como la configuración necesaria de las rutas y permisos para cada una de las carpetas y ficheros clave para su correcto funcionamiento.

Capítulo 6 - Evaluación con usuarios

Si queremos que nuestra aplicación esté bien diseñada, es importante hacer una evaluación con usuarios para saber cómo trabajan y cómo ejecutan las tareas que hemos diseñado (Nielsen, 2001). Es decir, debemos observar qué hacen y cómo se comportan los usuarios sin creer en aquello que dicen o hacen o van a hacer en un futuro.

En la actualidad, cuando se crea una nueva aplicación, el desarrollador piensa que va a ser fácil, competente y grata de usar pero, ¿es realmente cierto? De primeras, todo va a resultar útil y sencillo, pero cuando los usuarios empiezan a usar la aplicación, el propio desarrollador puede ver que no es cierto ya que, por ejemplo, un color que ha usado y ha interpretado de una forma no se interpreta de la misma manera por parte de los usuarios de la misma, un botón que se ha situado en un sitio específico y sin embargo, debería estar en otro porque el usuario no lo encuentra o no se lo espera donde está…

En nuestra evaluación con usuarios, vamos a fijar una serie de etapas que vamos a ir completando para finalmente obtener las conclusiones acerca de todo lo que vamos a ir evaluando:

1. Objetivo de la evaluación
2. Seleccionar a los participantes
3. Fijar las tareas de evaluación
4. Analizar los datos recogidos
5. Informe de resultados
6. Conclusiones finales

En nuestro plan de evaluación, vamos a evaluar el proyecto creado: "Dashboard para jueces en línea". Este proyecto consiste en una aplicación formada por diferentes paneles de control, gráficas y tablas con datos relevantes cuyo objetivo es facilitar la visualización de diferentes datos relevantes del juez en línea de "¡Acepta el reto!" para nuestros dos grupos de usuarios: administradores y profesores.

6.1 Objetivo de la evaluación

Una vez acabada la implementación de la aplicación del "Dashboard para jueces en línea" y en una versión web estable, nos viene bien hacer una comprobación final con usuarios para saber si se han cumplido las necesidades de los usuarios entrevistados en el Capítulo 3 - Diseño y si nuestra aplicación es sencilla de usar.

En nuestra evaluación con usuarios vamos a fijar una serie de objetivos principales en base a lo que queremos evaluar de nuestra aplicación:

- Saber si nuestra aplicación es fácil de usar: los usuarios ejecutan las tareas sin ningún problema.

- Saber si nuestra aplicación es eficiente: los usuarios saben cómo interactuar en nuestra aplicación y cómo acceder a las diferentes pantallas.

- Saber el grado de satisfacción con la aplicación.

A partir de estos objetivos, vamos a fijar una serie de preguntas que queremos que responda la evaluación:

- ¿"Dashboard para jueces en línea" es una aplicación sencilla de usar?
- ¿Se entiende bien la disposición de los elementos en las diferentes pantallas?
- ¿Los usuarios encuentran los datos que desean?
- ¿Los usuarios interactúan con facilidad entre las diferentes pantallas?
- ¿Se ocasionan errores de interpretación?
- ¿Se echa en falta alguna funcionalidad?

6.2 Seleccionar a los participantes

En cuanto a los participantes de nuestras tareas de evaluación, se van a seleccionar a los administradores de *¡Acepta el reto!* y a algunos de los profesores que participaron en las encuestas realizadas en el Capítulo 3 - Diseño en la fase de investigación.

Finalmente las tareas han sido realizadas por un total de seis participantes: dos administradores y cuatro profesores. Uno de los participantes es administrador y también profesor por lo que ha realizado ambas evaluaciones y en la parte del administrador no ha contestado las que son idénticas en ambas evaluaciones. De todos ellos obtendremos el feedback relevante sobre nuestra aplicación: usabilidad, visibilidad de datos e interacciones entre páginas.

Ahora que ya tenemos a los participantes es el momento de describir las tareas que van a realizar.

6.3 Fijar las tareas de evaluación

En esta etapa vamos a definir las tareas que realizarán cada uno de los participantes. Estas tareas van a estar agrupadas según los dos roles que tenemos de participantes: los administradores y los profesores.

Las tareas a realizar por cada uno de los dos grupos de participantes van a ser diferentes ya que, sus paneles de control van a ser parecidos, pero no idénticos y queremos recoger la mayor cantidad de datos para analizarlos en la siguiente etapa. Las tareas que vamos a proponer estarán formuladas en forma de preguntas, las cuales deben ser específicas, evitando que el usuario nos responda con un simple sí o no, precisas y medibles para que nos ayuden a saber cuál es la impresión real de los usuarios.

Nuestras preguntas van a estar relacionadas con la facilidad de uso, facilidad de interactuar en la aplicación y facilidad a la hora de buscar datos:

Administradores:

- ¿Cómo accedería al listado de usuarios y al Detalle de un problema desde el Dashboard del Administrador?
- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿cómo calificaría la forma de acceder a las diferentes páginas que forman el dashboard?
- ¿Sabría decirme cuántos WA se han producido en todos los envíos realizados en el Dashboard del Administrador?
- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Se entienden bien las leyendas de los gráficos?
- ¿Cuántos participantes tiene Perú?
- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Sabría acceder a ver el número de participantes de cada país?
- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Cómo valoraría la disposición de los botones?
- En una escala del 1 (no me gusta) al 5 (me encanta), ¿Cómo valoraría el aspecto de la interfaz?
- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Cómo valoraría la facilidad de uso de la interfaz?
- ¿Qué es lo que más le ha gustado de la aplicación?
- ¿Qué le ha gustado menos en la aplicación?
- ¿Qué mejoraría de la aplicación?

Profesores:

- ¿Cómo accedería al listado de alumnos y al Detalle de un problema del curso desde el Dashboard del profesor?
- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Cómo calificaría la forma de acceder a las diferentes páginas que forman el dashboard?
- ¿Podría decirme cuántos TLE se han producido en los envíos de nuestros alumnos?

- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Se entiende bien la leyenda de Aceptados/Fallidos?
- Acceda al Detalle de un alumno y vaya al área de Problemas intentados, ¿qué significa cada uno de los colores usados en esa área?
- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Se entiende el área de problemas intentados?
- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Cómo valoraría la disposición de los botones?
- En una escala del 1 (no me gusta) al 5 (me encanta), ¿Cómo valoraría el aspecto de la interfaz?
- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Cómo valoraría la facilidad de uso de la interfaz?
- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Recomendaría esta aplicación a otros profesores?
- ¿Qué es lo que más le ha gustado de la aplicación?
- ¿Qué le ha gustado menos en la aplicación?
- ¿Qué mejoraría de la aplicación?

Una vez detalladas cada una de las tareas a realizar por cada uno de los participantes, se les van a presentar a través de Google Forms otras preguntas:

- Nivel de satisfacción de la aplicación en una escala del 1(nada satisfecho) al 5 (muy satisfecho).
- Saber qué es lo que más ha gustado y lo que menos en la aplicación.
- Mejoras en base a la aplicación.

Las tareas que se van a realizar por nuestros participantes han de estar documentadas para que el usuario tenga claro lo que va a realizar, cómo lo va a realizar y nos reporte si ha conseguido realizarlo o no, también obtendremos el grado de satisfacción de la interfaz, problemas encontrados y posibles sugerencias en cuanto a ella.

La realización de todas las pruebas que hemos propuesto, debido a la situación actual, van a ser de forma remota a través del software de *Google Encuestas* (Google) será en cada una de estas "encuestas", dónde encontrarán un guión orientativo para que el uso de la aplicación sea más sencillo y se empiecen a familiarizar con ella ya que les dará una idea global de la misma.

Al realizarse estas evaluaciones de forma remota, el usuario va a estar solo, va a efectuar las pruebas con la información de la que dispone, y va a responder a las preguntas que se le han a propuesto. De esta manera, obtendremos la mayor parte del feedback necesario basándonos principalmente en las respuestas obtenidas por esas personas evaluadas.

Para los datos recolectados, vamos a ver si los usuarios que van a realizar las pruebas son caso de éxito en cuanto a completar las diferentes tareas o no, si esas tareas son sencillas de realizar o si por el contrario son complicadas, si al hacer las tareas se ven elementos que llevan a una interpretación diferente de la que se quiere dar y si el usuario está satisfecho una vez completada la tarea.

Una vez tengamos las respuestas por parte de todos los participantes de nuestra evaluación, nos toca recoger toda la información y revisar y analizar todos los datos obtenidos por parte de los evaluados.

6.4 Informe de resultados

Nuestro objetivo final en la evaluación, ahora que ya tenemos todos los datos examinados, va a ser resumir los hallazgos, proponer soluciones y especificar prioridades.

Una vez concluida la evaluación con usuarios, hemos ido analizando las conclusiones a partir de las preguntas y tareas realizadas.

la forma de acceder a las diferentes páginas del dashboard resulta sencillo tanto a los profesores como a los administradores ya que, puntúan con un 4 o un 5 dentro de una escala de 1(muy complicado) a 5 (muy fácil) (Figura 52 y Figura 53).

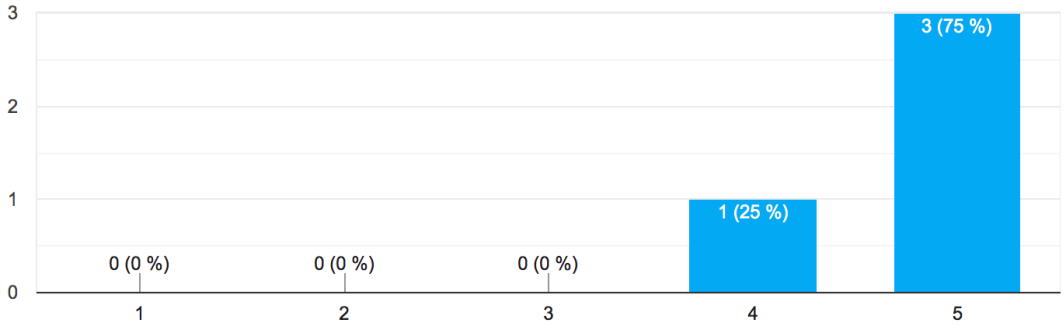


Figura 50: Resultado de la evaluación con usuarios (profesores) sobre la calificación de acceder a las diferentes páginas.

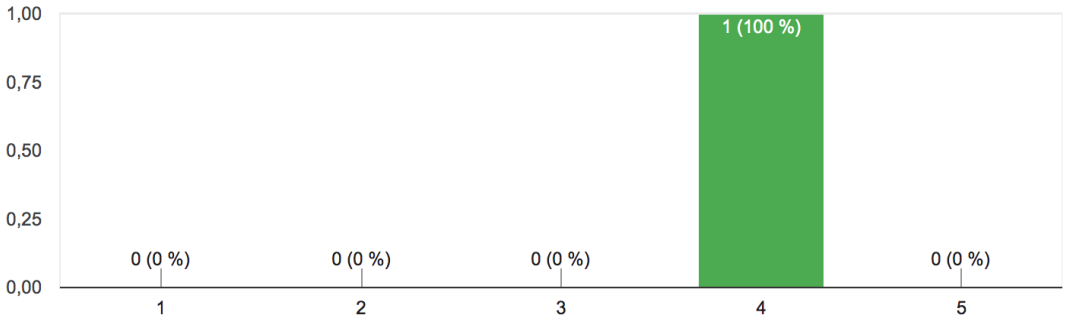


Figura 51: Resultado de la evaluación con usuarios (administradores) sobre la calificación de acceder a las diferentes páginas.

La lectura de datos, en el caso de los administradores saber el número de participantes que tiene Perú, y en el caso de los profesores saber el número de TLE que existen en los envíos realizados, se realiza de forma correcta por lo que, de igual manera, el acceso a los datos es sencillo.

Vemos además que a los participantes les ha gustado la aplicación y es sencilla de usar. Todos los participantes nos evalúan la facilidad de uso de la aplicación con un 4 (Figura 52 y Figura 54). En cuanto al aspecto de la aplicación, las respuestas son más diferentes, pero la califican, en general positivamente (Figura 53 y Figura 55).

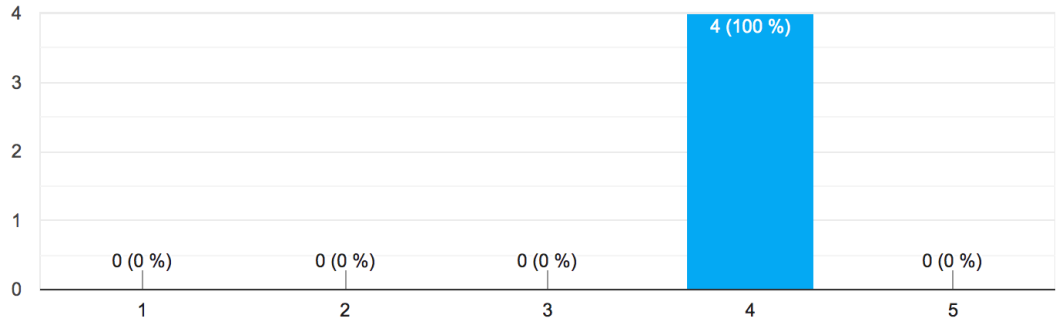


Figura 52: Resultado de la evaluación con usuarios (profesores) de la facilidad de uso.

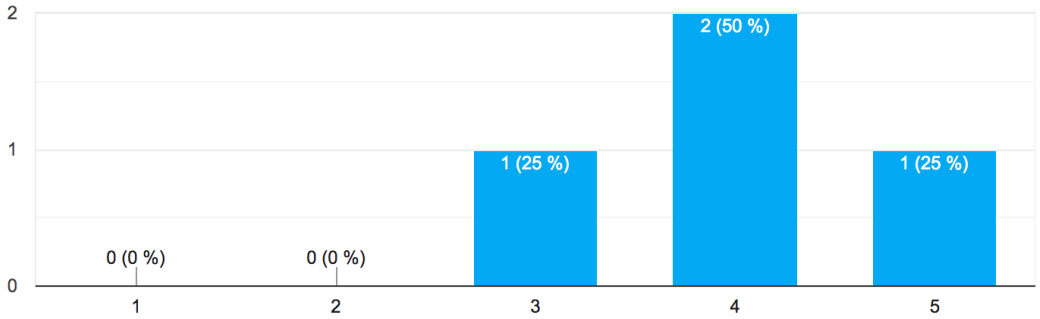


Figura 53: Resultado de la evaluación con usuarios (profesores) del aspecto de la interfaz.

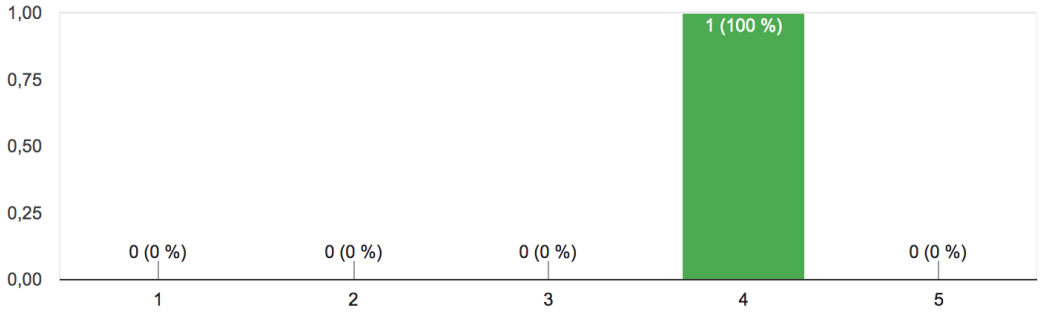


Figura 54: Resultado de la evaluación con usuarios (administradores) de la facilidad de uso de la interfaz.

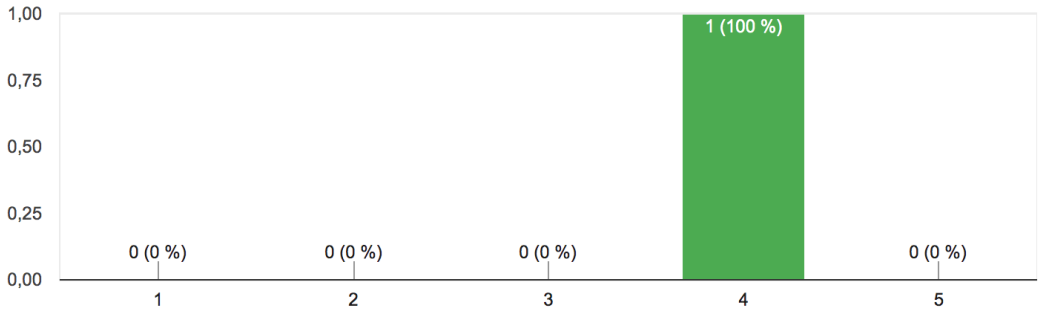


Figura 55: Resultado de la evaluación con usuarios (administradores) del aspecto de la interfaz.

Llegamos a la conclusión de que es una aplicación con aspecto atractivo, aunque se podría mejorar, fácil de usar y la recomendarían a otros profesores que no la hayan usado (Figura 56).

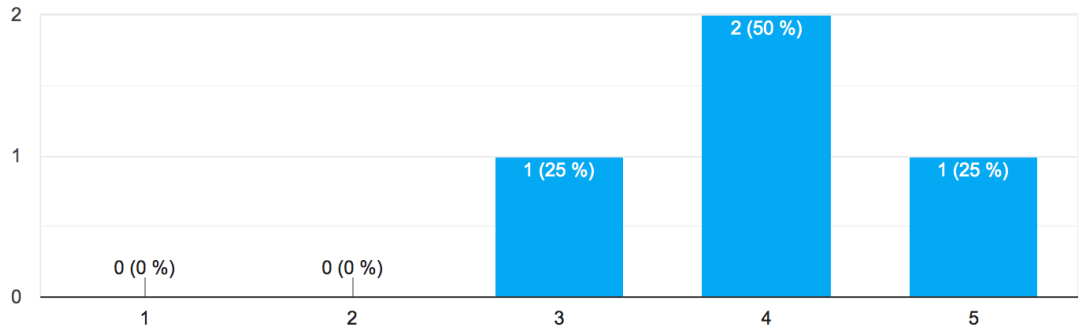


Figura 56: Resultado de la evaluación con usuarios (profesores) de la recomendación a otros profesores que no la hayan usado.

Las leyendas de los gráficos que tiene la aplicación se entienden bien ya que obtenemos puntuaciones entre 3 y 5 (Figura 57 y Figura 58).

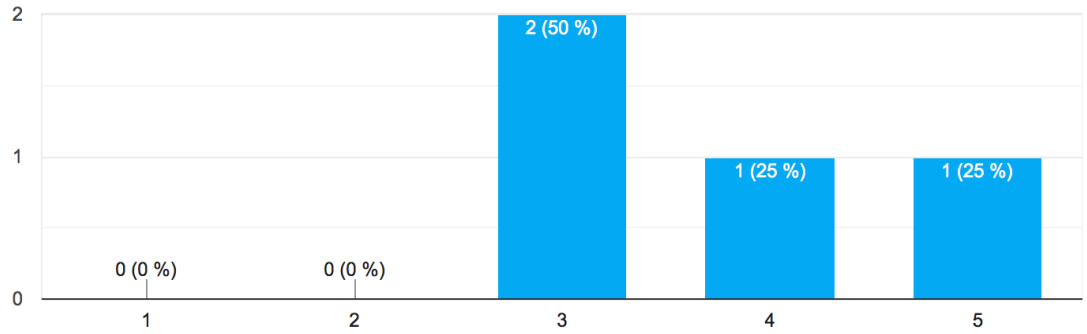


Figura 57: Resultado de la evaluación con usuarios (profesores) del entendimiento de los gráficos de la interfaz.

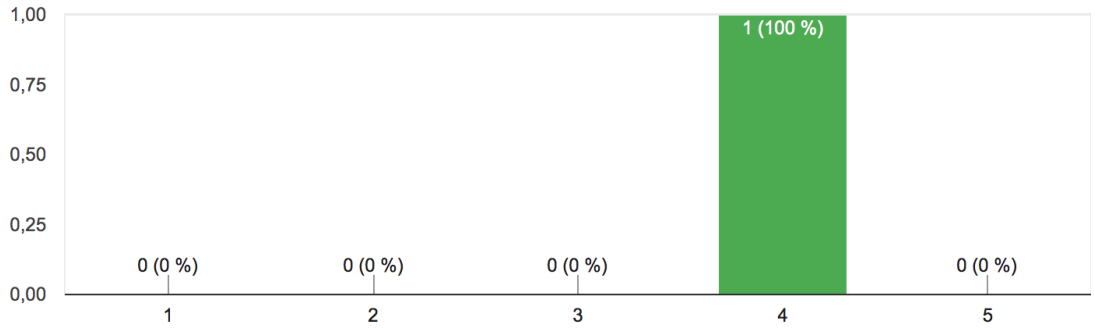


Figura 58: Resultado de la evaluación con usuarios (administradores) del entendimiento de los gráficos de la interfaz.

Además, han comentado que lo que más les ha gustado es que la aplicación es intuitiva y hay gran cantidad de datos para ver y poder analizar.

Por el contrario, lo que menos ha gustado es que el tiempo de carga de la aplicación es lento. Esta propiedad es algo que no ha gustado a la mayoría de los participantes y por tanto se tendría que solventar. Para solucionar este problema sería interesante poder tener la base de datos mejor estructurada de manera que tuviésemos tablas aplanadas con datos necesarios, en vez de tener un montón de tablas que tenemos que ir cruzando cada vez que necesitamos datos ya que esos cruces ralentizan la carga de la aplicación.

A algunos participantes les gustaría tener más formas de poder ordenar las tablas, lo que significa que no se ha transmitido correctamente que sí se puede

ordenar de la forma deseada. Para que se sepa que se puede ordenar de la forma deseada, se podría poner una anotación en la misma tabla comentándolo de forma que no hubiese dudas al respecto, También hay algún problema con los colores usados para los problemas intentados: rojo y verde, ya que para personas daltónicas sería complicado diferenciarlos, por tanto, cambiando esos colores, también se solucionaría.

Respecto al menú que hay ubicado en la parte izquierda nos sugieren que forme parte de la pestaña actual ya que de la forma que está puesto ahora mismo parece que jerárquicamente está por encima.

Otra de las cosas con la que hay diferentes respuestas es con la disposición de los botones (Figura 59 y Figura 60).

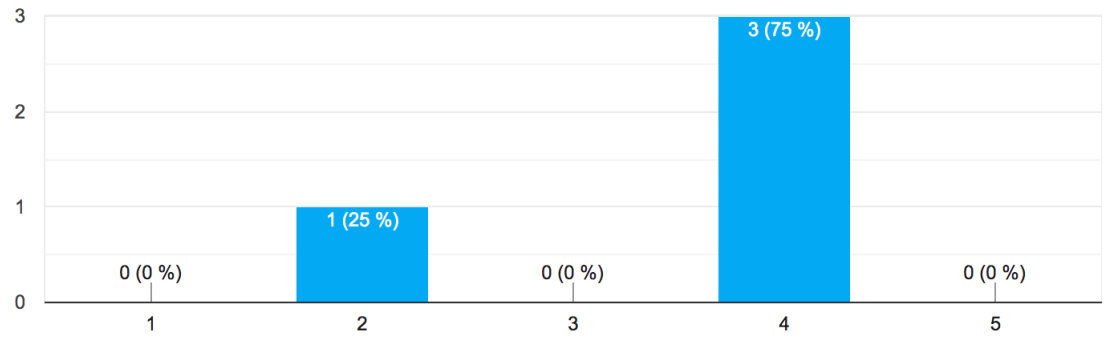


Figura 59: Resultado de la evaluación con usuarios (profesores) sobre la disposición de los botones en la interfaz.

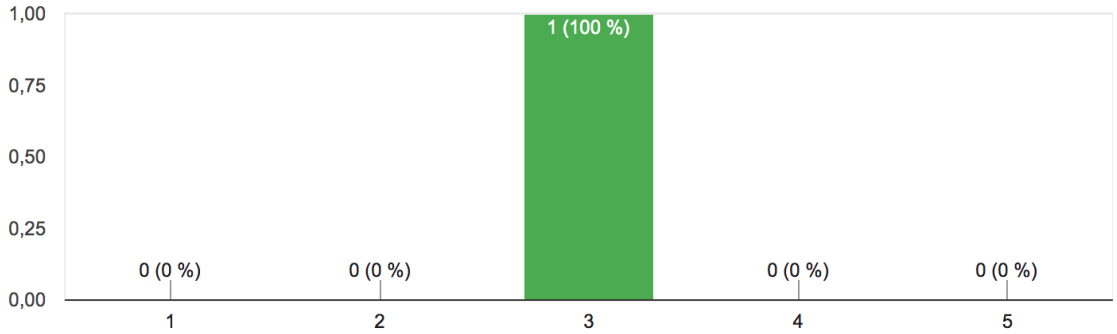


Figura 60: Resultado de la evaluación con usuarios (administradores) sobre la disposición de los botones en la interfaz.

Los administradores en esta evaluación han comentado que la información no llega a ser sensible de cara a su rol excepto la información sobre los nuevos usuarios y los Internal Errors. Para buscar soluciones acerca de esto, habría que volver a revisar todas las entrevistas a los administradores y sacar más información relevante sobre esa información que quieren ver y en el caso de no encontrarla, habría que rehacer preguntas para llegar a encontrar lo que buscamos.

A continuación, en base a todos los datos analizados, definiremos las conclusiones.

6.5 Conclusiones finales

Cuando definimos los objetivos para la realización de esta evaluación, queríamos saber si nuestra aplicación era fácil de usar, eficiente y el grado de satisfacción con la misma. Una vez que ya tenemos el informe de todas las evaluaciones realizadas, hemos visto que en general la aplicación la califican con un 4, siendo 5 la nota máxima, en cuanto a facilidad de uso y al aspecto de la interfaz. Respecto al tiempo de carga de la aplicación, es algo que por el contrario no ha gustado y algo que se debería solventar para que resultase más eficiente y sería lo más prioritario.

Respecto a que hay usuarios en el ranking de usuarios del profesor que aparecen repetidos, ya lo hemos solventado. Así mismo, lo que han comentado con respecto a los botones de volver al Dashboard del administrador desde las interacciones del profesor en vez de volver al Dashboard del profesor, también ha quedado solventado.

También hemos observado que hay algunas gráficas que no se entienden correctamente como ocurre con la gráfica de envíos y de la evolución de envíos y al haber reducido las consultas de la base de datos, hay totales que aparecen que no coinciden y por tanto llaman la atención al igual que pasa con determinados usuarios.

Una vez resuelto el problema de la eficiencia, se podrían cargar mayor número de datos y por tanto todos ellos serían congruentes entre sí.

En general estamos contentos con los resultados obtenidos en estas evaluaciones y de cara al trabajo futuro, una de las principales tareas sería dar solución al tiempo de carga.

Capítulo 7 - Conclusiones y trabajo futuro

Este capítulo contiene las conclusiones, tanto en español como en inglés, acerca del proyecto que se ha realizado. Se explica, cómo se ha trabajado en el desarrollo del proyecto resultante y también se trata el trabajo futuro con diferentes puntos interesantes para llevar a cabo en un futuro.

7.1 Conclusiones

Para este proyecto, se ha creado un Dashboard para jueces en línea para el juez en línea de *¡Acepta el reto!*, el cual se creó para que principalmente usuarios de la Universidad Complutense de Madrid pudiesen usarlo bien para aprender a programar o mejorar sus habilidades de programación, en el caso de los alumnos, o bien para tener un sistema de enseñanza controlando los ejercicios de programación propuestos por los profesores.

Comenzamos investigando que son los jueces en línea y cuáles existen en la actualidad y qué son los paneles de control y cómo se construyen. Estas dos primeras investigaciones nos iban a resultar de mucha ayuda para ir teniendo las primeras ideas a la hora de diseñar nuestro panel de control. Esta parte fue una de las que más tiempo me llevó ya que, había que buscar mucha información y contrastarla para más tarde poder hacer unas buenas preguntas en las entrevistas a los usuarios que iba a tener nuestra aplicación y poder extraer sus necesidades a partir de ellas. Estas entrevistas fueron muy cómodas de realizar a través de Google Forms tanto para los entrevistados como para el entrevistador.

Realizar estas entrevistas fue una de las primeras fases del Diseño Guiado por Objetivos: la investigación. Esta fase, también contenía el apartado de analizar a la competencia, obtener los requisitos funcionales de los administradores y de los

profesores y diseñar la aplicación en papel en base a las necesidades obtenidas en ambos grupos.

La parte del diseño en papel también fue larga y muy exhaustiva ya que se tenían que estudiar cada uno de los diseños e ir mejorándolos en cada una de las iteraciones que realizamos. De esta manera la aplicación que más tarde íbamos a crear sería sencilla y usable y la experiencia del usuario satisfactoria.

Una vez que se llegó al diseño más próximo de las necesidades de ambos grupos, se llevó a cabo el estudio e investigación de las herramientas de trabajo para la implementación de la aplicación que íbamos a programar. Estas herramientas debían de ser sencillas para nuestro uso y potentes para que nuestra aplicación cumpliese con todas las necesidades creadas en el diseño final en papel.

La implementación de la aplicación requirió de esfuerzo y tiempo ya que la estábamos creando en base al diseño en papel, pero al programarlo, había elementos en los paneles de control que carecían de sentido o de usabilidad y necesitábamos otros elementos en los cuáles no habíamos pensado durante el diseño en papel.

Los elementos que íbamos cambiando no eran muy sustanciales en cuanto a la implementación, pero sí al diseño.

Una vez que la implementación quedó concluida y se llevaron a cabo varias pruebas en la aplicación por nuestra parte, llegaba la hora de hacer la evaluación con los usuarios finales para que siguiendo unos objetivos que habíamos fijado, nos diesen su *feedback* en base a la aplicación creada.

Con las conclusiones extraídas de esa evaluación, hemos visto que la aplicación creada resulta fácil de usar y el aspecto de la interfaz es bueno, sin embargo, el tiempo de respuesta es lento y se debe mejorar.

El trabajo futuro fue definido según implementábamos la aplicación, ya que íbamos descubriendo algunas mejoras sobre la marcha, y también a partir del feedback dado por los participantes de nuestra evaluación con usuarios.

7.2 Conclusions

In this project, we have created a Dashboard for online judges more exactly for the online judge "*¡Acepta el reto!*" which was mainly created to be used by the users of the University Complutense of Madrid. On one hand the students would be able to use it to learn how to program and improve their programming skills, on the other hand the lecturers would be able to use it as a teaching system, where they can check all the programming exercises offered.

Firstly, there were a research about what are online judges and which ones already exist. We will also research about what dashboards are and how to create them. These two researches will help to get a global idea to start designing the dashboard. This part required a big amount of time, as we had to collect a lot of information and contrast, to afterwards create good interview´s questionnaire to the users of our app, the main propose was to gather their needs. These interviews were done using Google Forms, which was easy to use for both the interviewed users and the interviewer.

The interviews were one of the first parts of the Goal-Directed Design: the Investigation. This phase also had a part to analyse the competence, get all the functional requirements from the administrators and lecturers and design the app on paper with all the needs from both groups.

The paper design step was long and exhaustive as it was needed to study each design and improve them in each of the iterations made, as a result the app would be easier to use and a great experience for the user. This ensures that the app that will be launched later will be easy and usable and the users will have a satisfactory experience.

Once we achieved the best design to cover every group's needs there was a research of which tools will be used to implement the app. These tools required to be easy to be used but with a big potential to fulfil all the necessities named on the last design on paper.

The app implementation demanded effort and time, as we designed it based on the paper design, but when we started to program it, we discovered that there were some dashboard elements not longer needed or did not make sense, so there was a need to include new elements that we did not think were needed during the paper design.

The elements that we changed did not have a big impact in the implementation but on the design.

Once the implementation was concluded and a few app trials were made, it was time for the evaluations with the final users, where following the goals we have set they will give us their feedback about this app.

From the conclusions drawn, we have found that the app created is easy to use and the visual design of the interface layout is good, however, the response time is slow and should get better.

The final work was defined while we were implementing our app because we were noticing some possible improvements and also using the feedback collected from the users' evaluation.

7.3 Trabajo futuro

Una vez implementada toda la aplicación, ha habido una serie de propuestas extraídas a través de las entrevistas realizadas a los usuarios en el Capítulo 3 - Diseño, que no se han llegado a implementar ya que no vimos que fueran de carácter primordial para este proyecto, pero que sí nos gustaría tener en cuenta para futuros desarrollos a mejor de la aplicación .

Para tener constancia de ellos y poder rescatarlos, vamos a describirlos de forma detallada:

1. **Tiempo de carga.** Redefinir la base de datos de forma que tengamos tablas aplanadas que nos disminuyan el tiempo de acceso a los datos y por tanto solucionen el tiempo de carga general de la aplicación.
2. **Ver el código que han mandado los alumnos.** Esta información en el momento de crear este panel de control no lo vimos relevante ya que el proyecto está basado en los paneles de control y no con lo las funcionalidades de *¡Acepta el reto!*. Es algo que se podría implementar en un trabajo futuro, ya que muchos de los jueces online existente a día de hoy sí que disponen de esta funcionalidad y ayudaría mucho a los profesores para poder ver el código enviado por el alumno.
3. **Saber si las soluciones de los usuarios son originales o copias.** Relacionado con la propuesta anterior, si se puede ver el código, también se puede llevar a cabo esta propuesta. Algunos de los jueces en línea investigados disponen de un algoritmo que es capaz de saber si una solución es original o es copia. A los profesores también les resultaría de gran ayuda.
4. **Comparar código.** Sabiendo las soluciones que mandan los usuarios, se podría ir comparando los códigos para ver los diferentes cambios que va haciendo un usuario hasta llegar a la solución que recibe el **AC**. Este comparador, podría funcionar de forma similar a algunos de los controles de versiones que existen a día de hoy en aplicaciones como Github, Gitlab o Mercurial.
5. **"*Heatmaps*".** Añadiendo *heatmaps* (mapas de calor) en determinados puntos del panel de control ayudaría a los usuarios (profesores o administradores) a saber cuándo se hacen la mayoría de los envíos: Día, mes, hora…
6. **Dificultad de los problemas.** Actualmente, en la aplicación no disponemos de un nivel de dificultad de cada uno de los problemas. Esta funcionalidad resultaría muy interesante para el rol de profesor y también para los usuarios ya que dependiendo del nivel que vayan adquiriendo a la hora de programar, podrán elegir unos u otros problemas. Este grado de dificultad también lo tenemos disponible en algunos de los jueces en línea existentes en la actualidad.
7. **Recomendaciones de problemas.** Relacionado con el punto anterior, estaría también esta funcionalidad. Sabiendo el grado de dificultad de los problemas y la categoría, se podría establecer un recomendador de problemas para los usuarios e incluso para los profesores.

8. **Porcentaje de intentos.** Refiriéndonos a los problemas, resultaría interesante poder tener una métrica del porcentaje de intentos que realiza un usuario sobre un problema antes de rendirse.
9. **Evolución de envíos.** En este proyecto, la evolución de los envíos ya lo hemos implementado para ambos roles. En el caso del rol del administrador, hemos pensado que resultaría interesante poder añadir un filtro de fechas por si el usuario desea ver una evolución en unas determinadas fechas ya que ahora mismo solo podría ver la evolución anual de esos envíos.
10. **Resolución responsive.** Adaptar la aplicación web a otras resoluciones, de forma que, si se quiere acceder a la aplicación desde otro dispositivo que no sea un ordenador, no haya problema en la visualización de los diferentes paneles de control.

Apéndice A - **Entrevistas Acepta el reto 2018**

En este anexo, está la recopilación de todas las entrevistas realizadas tanto a profesores como a los administradores que usan www.aceptaelreto.com y http://ucm.aceptaelreto.com.

La información acerca de estas entrevistas se encuentra en: https://docs.google.com/forms/d/1VdUPzEiRyv7cmiPsXYGv_pePFo5JJw61thWHY9oj8q0

1. ¿Cuánto tiempo llevas utilizando la herramienta?

Unos tres años

1 año con el soporte de Marco, otro año más de manera informal y varios años más sin prestarle mucha atención

Un curso académico

2 años (no está claro si "la herramienta" es Acepta el reto o la página ucm.aceptaelreto.com, he contestado como si fuera lo segundo)

1 año

Desde antes de que se lanzara

Desde los principios 2012

1 curso

2. ¿Cuál es el objetivo de los ejercicios propuestos?

Proponer problemas a los alumnos. Los problemas se utilizan para que los alumnos practiquen, no forman parte de la evaluación

Que los alumnos aprendan a diseñar algoritmos y a utilizar estructuras de datos, poniendo a prueba su funcionamiento más allá de la solución teórica en papel

Que los alumnos practiquen más el diseño de algoritmos

Aprender a utilizar las estructuras de datos vistas en clase y diseñar algoritmos eficientes que las utilicen

Que los alumnos tengan una gran variedad de ejercicios con los que practicar y yo pueda visualizar fácilmente quienes han hecho que problemas

Para la docencia: clase de EDA o FLI (sus asignaturas) y por otro lado para concursos de programación.

Para que los alumnos practiquen

Que los alumnos hagan más ejercicios y puedan comprobar si su solución funciona.

Que la asignatura (EDA) les motive más.

3. ¿Cómo te riges para seleccionar los problemas que propones a los alumnos?

Problemas semejantes a los aprendidos en clase

Que sean similares a los que resolvemos en clase, aunque de complejidad variable

Que se correspondan con el tema/contenido que se está impartiendo en clase.

Que sean de dificultad variada y puedan resolverse con lo visto en clase

Según el tema de la asignatura

Mirando las categorías (fundamentos de programación)

Por las categorías y los temas que estoy dando en el momento

Hablo con los gemelos o propongo ejercicios que ya he resuelto yo.

4. ¿Con qué frecuencia se plantean estos ejercicios?

Se plantean al empezar cada tema de la asignatura

1 conjunto de problemas para cada tema

Un conjunto de ejercicios para cada tema.

Todas las semanas

Una vez por cada tema

Todas las semanas (2)

Planteo un conjunto de ejercicios cada vez que empiezo un tema de la asignatura.

Aproximadamente cada 3 semanas

5. ¿Con que frecuencia consultas la herramienta?

Es muy variable

Al menos semanalmente, pero si veo que están trabajando llego a mirar varias veces al día

Una vez a la semana

Cada 2 o 3 días

Todas las semanas para animarlos a que realicen los ejercicios antes de que finalice el periodo establecido

Todas las semanas

Varias veces por semana

Depende. A veces cada semana. A veces cuando termina un tema de la asignatura.

6. ¿Piensas que la herramienta debería elaborar informes de forma desatendida y enviárselos de forma periódica? En caso afirmativo, ¿con qué frecuencia? ¿qué tipo de informes?

Podría ser interesante. Se debería preguntar si se quiere recibir información y solo mandarla en caso de que se pida. Se podría informar de los nuevos problemas que van apareciendo. De acontecimientos como los concursos. O proponer problemas similares a los últimos resueltos por el alumno. El tema de estadísticas ya lo puede consultar él en la página y las pistas es mejor que se den bajo demanda.

Podría ser útil. La frecuencia podría depender del número de problemas en cada conjunto de ejercicios y del tiempo disponible para resolverlos (es decir, periodos variables). Particularmente, me interesa saber cuántos alumnos están intentando

resolverlos y si hay algún problema que esté dando algún problema concreto.

Sería útil. Una vez por semana o quincena. Participación /éxito de los alumnos.

Estaría bien. Cada 2 días, con los problemas intentados, cuántos se han resuelto

No

No / Administrador: Para ver que funciona todo bien. Para ver los nuevos usuarios y los envíos que se han hecho. Por semana.

No me lo había planteado, pero, no estaría mal. Una vez a la semana. Para ver el tipo de actividad de los alumnos y problemas más resueltos

Estaría bien. Quizás la frecuencia de los informes se debería poder configurar, pero como 1 por semana. Seguramente algo parecido a lo que ya se muestra en ACR: ranking de alumnos, número de envíos y soluciones a cada problema, etc.

7. ¿Qué lenguajes de programación se proponen para los ejercicios?

C++ (6)

C++ (me gustaría poder utilizar Python)

Actualmente C++. Pero estaría bien incorporar Python o Javascript a ACR.

8. ¿Para qué usas la web http://ucm.aceptaelreto.com?

Para que los alumnos resuelvan problemas de la asignatura. Estos problemas no se usan para calificación porque no se puede consultar el código utilizado por el alumno.

Para ver cómo marchan los alumnos de mi grupo

Para seleccionar los ejercicios.

Para ver los envíos que han realizado mis alumnos

Para comprobar el trabajo de los alumnos

Para buscar ejercicios y ver que los alumnos están haciendo los ejercidos.

Monitorización del grupo de alumnos.

Para hacer el seguimiento de los alumnos, ver que ejercicios resuelven y quien resuelve más ejercicios.

Para proponer nuevos ejercicios a los alumnos y ver el ranking.

9. ¿Qué datos de los que aparecen en la web http://ucm.aceptaelreto.com te resultan interesantes?

Todos (2)

La agrupación de problemas por temas. El número de personas que ha resuelto el problema.

La clasificación de los problemas según su categoría.

Ejercicios resueltos, numero de intentos en cada uno, fechas

El número de problemas resueltos de cada alumno y en el ranking ver los problemas que más se resuelven y menos (más o menos entendidos)

Los problemas que ha resuelto cada alumno y el orden de los alumnos que han resuelto más o menos

¿Todos?

10. ¿Qué pasos sigues en la web http://ucm.aceptaelreto.com cada vez que la consultas?

Introduzco mi usuario y contraseña. Consulto los problemas a veces por temas, a veces por orden. Selecciono el problema que quiero resolver, veo el enunciado...

Miro la clasificación general, luego la del tema actual y, si veo algún ejercicio no intentado o con muchos envíos, intento ver qué problema tienen en ese ejercicio

Voy a la pestaña de problemas y busco en las categorías que me interesan.

Ver los últimos envíos, qué problemas un alumno ha intentado y todavía no ha resuelto correctamente

Comprobar la cantidad de ejercicios resueltos por cada alumno/equipo, comprobar

los errores que han cometido (run error, wrong answer ...)

Asignatura>tema (que se está dando) y ver los problemas enviados por los alumnos

Juego de problemas activo

Normalmente ver el ranking general de alumnos y ver cómo van los problemas del tema que actualmente estoy impartiendo.

11. ¿Qué información individual extraes de cada alumno?

El número de problemas que ha resuelto

Los ejercicios intentados y resueltos

Los problemas que ha resuelto.

Cuántos problemas ha resuelto correctamente

Número de ejercicios resueltos de los que he propuesto.

Ver los problemas que realizan y el número de intentos

Ver los problemas enviados

Qué ejercicios ha intentado y cuáles ha resuelto

12. ¿Qué información agregada extraes de cada ejercicio?

Los problemas que resultan más difíciles o más fáciles

los alumnos que han sido capaces de resolverlo, además de adivinar problemas habituales

No entiendo la pregunta.

Si les ha resultado difícil o no, viendo cuántos envíos han necesitado antes de resolverlo

Cantidad de alumnos que lo han intentado o no, y cantidad de alumnos que lo han resuelto

Ninguna. No se sabe el tanto por ciento de la gente que acierta o no

Cuanta gente lo ha resuelto.

13. ¿Qué datos echas de menos si nos referimos a datos individuales?

Poder ver el código de los alumnos

Los que tendría si pudiese entrar como juez

Ninguno

Poder ver el código de soluciones incorrectas para ayudar a los alumnos

Casos de prueba que les han fallado y su código

La parte de los porcentajes. Saber qué problema ha sido más o menos resuelto. No se sabe en la HOME cuantos ejercicios ha resuelto cada alumno, solo salen los ejercicios que han resuelto. La tabla está bien porque más o menos sabes los ejercicios que se han resuelto

Los fallos más comunes del grupo en http://ucm.aceptaelreto. El código que han mandado los alumnos solo se puede ver en www.aceptaelreto.com y no en http://ucm.aceptaelreto

El número de intentos y soluciones

Saber si su solución es original o una copia. Visualizar cuándo ha resuelto cada ejercicio.

14. ¿Qué datos echas de menos si nos referimos a datos agregados?

Poder buscar problemas con palabras clave

los que tendría si pudiese entrar como juez

Ninguno

Igual

Lo mismo

https://ucm.aceptaelreto.com: esta ok.

www.aceptaelreto.com: los envíos suelen ser en C++. No se saben los índices de aceptación en cuestión del lenguaje

No hay estadísticas por grupo, ni se sabe el número general de aceptados o rechazados de todos los problemas juntos, para saber un poco la evolución de enviados en el tiempo

Conocer la distribución de las soluciones en el tiempo. Número de soluciones que son copias.

15. ¿Cómo accedes a los diferentes apartados de la Aplicación?

No se entiende la pregunta

pinchando en los enlaces

¿¿¿¿

A través de los menús

Clics (2)

¿Aplicación? Para mi es una página web. Pulsando en los enlaces

16. ¿Qué dispositivo utilizas para consultar la Aplicación?

El ordenador

PC. Muy pocas veces accedo con móvil o tableta

Un Mac

Ordenador, móvil, tablet

Ordenador (3)

Mi ordenador de trabajo (Macbook Pro).

17. ¿Cómo localizas un problema dentro de la Aplicación?

O por numero o por tema

No entiendo muy bien la pregunta ¿En qué circunstancias? Normalmente, entro en la página donde he puesto los problemas y pincho en él

Por el número.

En el listado por volúmenes o por categorías. Conozco casi todos

Están divididos en pestañas y se puede pinchar sobre cada uno.

Categorías (2)

¿La aplicación es ACR o ucm.aceptaelreto.com? En el primero buscando por las categorías, en el segundo buscando en los grupos de problemas que yo he propuesto.

18. ¿Qué soporte encuentras más cómodo a la hora de estudiar los datos de sus alumnos? ¿Soportes digitales? ¿Soportes físicos?

Si quiero ver estadísticas de las cosas que van haciendo soporte digital. Si quiero corregir el código soporte físico.

Ya me he acostumbrado a los digitales. Hay tantos datos que consultar en sitios diferentes que el soporte físico no me resulta cómodo

Soportes físicos.

Soportes digitales

Digitales (3)

Digital. Mejor en un formato que pueda modificar. Estaría bien poder exportar las tablas y rankings a Excel.

19. ¿Qué tipo de gráficas y diagramas encuentras más útiles a la hora de valorar a sus alumnos en conjunto? ¿Y de forma individual?

No los he utilizado

Como tengo un grupo pequeño, los números me valen

Depende de la información.

Conjunto: por un lado, la evolución de los envíos, como ha avanzado la clase a lo largo del curso, ver una progresión temporal a través de envíos o problemas resueltos. Individual: ver el porcentaje de problemas resueltos, progresión de problemas hechos/resueltos. Parte temporal un rollo GitHub

Conjunto: diagramas de sectores. Estadísticas de aceptados o errores. Individual: pistas o información de los casos en los que se equivocan y los cambios que hacen de un envío a otro para ver la evolución.

La forma de visualizar la información depende de la información que se quiera visualizar. Creo que esta pregunta está mal planteada. ¿A la hora de valorar qué parámetro de los alumnos?

20. ¿Qué tipo de seguridad consideras necesaria a la hora de acceder a los resultados de los problemas propuestos?

No lo se

Vuelvo a no entender la pregunta. Una seguridad muy segura pero que me permita verlo todo.

¿¿¿¿

Debería estar más protegida no hay anonimato. a través del id sabes los nombres y apellidos

Debería estar protegido, la url la conoce poca gente, pero debería ser privada para los profesores, por asignatura y grupo.

No sé si hay algún problema con que esa información sea pública...

ENTREVISTA ADMINISTRADORES:

21. ¿Cuál es tu interés particular en el uso de la herramienta?

Por un lado, quiero que los profesores puedan hacer esa página sin tener que montársela (ucm.aceptaelreto.com) y en (www.aceptaelreto.com) que puedan crear problemas ellos solos. Que el administrador no tenga que dar mucho soporte

Dar uso a los problemas que tenían en un cajón y no saldrían (dar salida a los problemas de los concursos, los alumnos mejoran en los concursos porque tienen

estos problemas para practicar) y para investigar

22. ¿Analizas los datos que la Aplicación recoge, o únicamente te preocupa que ésta funcione correctamente?

Si recojo datos y no están voy a la base de datos a buscarlos

Ambas cosas, no se analizan tanto como querría

23. ¿Dispones en la Aplicación de todas las herramientas necesarias para evaluar las soluciones propuestas por los alumnos?

Sí, pero, son ad hoc. Los administradores pueden ver el código y eso les pone a ellos en una situación ventajosa, otros profesores no pueden verlo

No, no está lo del div (no hay diferencias entre cada envío) y no hay comparación entre gente (no se sabe si se están copiando o no)

24. ¿Cuáles son los problemas o dificultades que encuentran los alumnos a la hora de enviar sus soluciones?

Pues que no tienen ni idea de programar. No entienden el concepto de juez virtual y tiene ciertas cosas que son tal cual, imprimir entrada con ci en vez de con scan o print

Crean programas muy básicos, comenten error de la gestión de la entrada porque no conocen como funciona el juez. Hay algunos enunciados que no se entienden bien o no se han escrito bien y quieren tener un ejemplo para ver que tiene que dar ese problema. La app no tiene que se pueda reenviar, por ejemplo, si se ha equivocado en un ';', tienen que copiar el código completo y volverlo a mandar, y hay veces que se equivocan al copiar. La gente se deja que no hay muchos ejemplos, pero eso, va en contra de nuestros principios

25. ¿Cuáles son los problemas o dificultades que encuentran los profesores a la hora de consultar los resultados?

Lo de los datos agregados, el ranking y el no poder acceder al código. No hay login diferente para alumno/profesor. Está a medias en la base de datos

No pueden ver el código. ucm se ha montado de forma local. Hay gente que no puede tener acceso a esos rankings

26. ¿Disponen tanto alumnos como profesores de algún canal para enviar sugerencias técnicas a la plataforma?

Hay un contáctanos, pero la gente no suele mandar nada. El otro día mandaron un email de que no funciona el ese formulario. El contáctanos es muy abierto, pero podría tener sentido que no fuese un buzón común, sino que fuese especifico

Hay un formulario para hacer envíos. Sobre papel sí sugieren, en la práctica no

27. ¿Existe algún sistema de soporte para los alumnos que encuentren problemas durante el uso de la plataforma?

El contáctanos. Van a verlos si son de la Universidad Complutense de Madrid

Escriben emails de vez en cuando.

Apéndice B - **Entrevistas 2019**

La información acerca de estas entrevistas se encuentra en: https://docs.google.com/forms/d/1m3sAEAz9ULdiot-LWDvEBSW-j5W9q_qhlzqr1pdwP5A

1. ¿Cuánto tiempo llevas utilizando la herramienta (aceptaelreto.com)?

3 años

3 años

Desde 2011

Desde 2014 y antes :)

8 años

6 años de forma personal. 3 años de forma docente

2. ¿Crees que aceptaelreto.com es una página fácil de usar? ¿Por qué?

En esencia sí, pero la búsqueda de problemas algunas veces me ha resultado difícil.

Sí, los ejercicios están clasificados por diferentes criterios y el registro se hace sin requerir demasiados datos.

Sí, no tiene mucha complicación

Si, se centra en lo esencial darte el problema y mostrarte donde enviarlo. La solución se actualiza sola.

Sí, por la categorización de los problemas

Sí. Muy sencilla. Pocas posibilidades y bien clasificadas, por lo que es imposible perderse

3. ¿Con que objetivo propones los ejercicios?

Objetivo de hacer prácticas o también de realizar pruebas de evaluación con un tiempo acotado.

Prepararse para el concurso de Programame, practicar estructuras de programación y para qué los alumnos se acostumbren a realizar los ejercicios cumpliendo estrictamente con las especificaciones dadas

Tareas evaluables dentro de mi programación didáctica

Diversión

Con el de estimular y motivar a mis alumnos, que le cojan el gusto por la programación.

(1) Inicio a la programación competitiva. (2) Implementación de algoritmos/paradigmas específicos explicados con anterioridad en clase

4. ¿Cómo te riges para seleccionar los problemas que propones a los alumnos?

En general, por la temática que se esté usando en ese momento en cuanto a programación.

Que sean accesibles y permitan practicar los contenidos que estemos viendo. A los alumnos que van mejor les propongo ejercicios más complejos.

Según las categorías de los problemas: Arrays, Strings, pilas, etc.

En función de la solución

Me rijo por la categoría de los problemas y su nivel de dificultad, es decir, el porcentaje de envíos y AC.

Cada bloque de problemas suelen estar centrados en un único algoritmo. O son problemas que ya tengo hechos y clasificados, o que se encuentran ya clasificados en acr.

5. ¿Cada cuánto tiempo planteas los ejercicios propuestos?

Las sesiones prácticas en las asignaturas suelen ser cada semana, las pruebas de evaluación cada más tiempo.

En los dos primeros trimestres continuamente, al menos una vez a la semana. Los contenidos del tercer trimestre se alejan más de lo que podríamos practicar con Acepta el reto.

Semanalmente

1 vez a la semana

Lo realizo fundamentalmente al final del primer trimestre cuando ya he visto con los alumnos estructuras de control y arrays. Lo suelo hacer una o dos veces a la semana

como "retos findes". A principios del segundo trimestre, mediados de enero organizamos la Fase Local de Programame, y luego ya avanzo en POO y no lo usamos tanto.

1 semana

6. ¿Con que frecuencia consultas la aceptaelreto.com?

Una vez a la semana mínimo.

Semanalmente

Varias veces durante el curso escolar, coincidiendo con los contenidos de mi programación

3-4 veces a la semana

Sobre todo, en los momentos que he mencionado antes. Diciembre y enero, casi a diario

1 mes. La mayoría de accesos los realizo a través de ucm.aceptaelreto.com para ver el progreso

7. ¿Piensas que la herramienta debería elaborar informes de forma desatendida y enviárselos de forma periódica? En caso afirmativo, ¿con qué frecuencia? ¿qué tipo de informes?

Quizás vendrían bien las estadísticas de intentos globales, aceptados... y qué porcentaje de personas de entre todas las posibles que podían hacer envíos los ha hecho y con qué tasa de éxito

No me lo había planteado, pero podría ser buena idea.

Sí, sería muy interesante contar con informes sobre Feedback de los casos de prueba no superados, por ejemplo. Por usuario, ver la cantidad de envíos y veredictos, etc. Los informes podrían ser bajo demanda.

Si generarlos, no necesariamente enviarlos. Accesibles en el perfil, en caso de ser enviados que pueda configurar la frecuencia dado que a veces hay entregas semanales y otras con más tiempo

Tal y como la uso no es necesario...

Veo interesante la generación de informes, pero no de forma automática y frecuente, sino solamente cuando se solicite. Sería interesante ver número de AC y número de

intentos, con relación a los envíos de mis alumnos, y en relación al global de todo acr. El número de intentos sería interesante desdoblarlo entre el número de intentos antes de un AC, y el número de intentos sin AC (como métrica de número de intentos antes de rendirse).

Además, una comparativa de TLE vs WA para cada problema puede ser útil para determinar de forma burda si se está usando el algoritmo correcto, o no .

8. ¿Qué lenguajes de programación propones para los ejercicios?

Java

Java

C++ o python

C++ y me gustaría python

C++

Java

9. ¿Echas en falta algo que pienses que es importante para ti en aceptaelreto.com? ¿El qué?

PYTHON!

Sugerencia de problemas

"Amistades" de usuario

En el listado de problemas, marcar de forma visual (colores?) los problemas intentados y los problemas con AC.

Estadísticas generales de la plataforma similares a las de mi perfil (a parte de los últimos envíos, lenguajes usados, número de envíos, nº AC / nº submissions, etc).

Un buscador más fácil para los problemas

Estrategias de resolución, más casos de prueba o pistas sobre los envíos que no consiguen el AC

No

Más pistas para los alumnos ante los RTE ;-))

10. ¿Hay algo en aceptaelreto.com que no veas útil? ¿El qué?

No

No

No

Nada

Todo me parece muy útil, me encanta la página

No... lo único que cada vez hay más gente subiendo los problemas resueltos...

ENTREVISTAS UCM ACEPTA EL RETO

La información acerca de estas entrevistas se encuentra en: https://docs.google.com/forms/d/1Twleb8pgQJX-rgu13mz4PVHIEt235178GgQXFMVhRzw

1. ¿Para qué usas la web http://ucm.aceptaelreto.com?

Pruebas de evaluación y sesiones prácticas de docencia

Desarrollo de sesiones de entrenamiento. Normalmente cada sesión enfocada en un algoritmo en concreto

Entrenamientos grupo UCppM

2. ¿Qué datos de los que aparecen en la web http://ucm.aceptaelreto.com te resultan interesantes?

Sobre todo, la parte de la documentación con las explicaciones.

Envíos en tiempo real. Clasificación de cada sesión. Recopilatorio de problemas en el lado derecho. Posibilidad de añadir un texto al principio de cada pestaña

Problemas, sesiones, espacio explicativo de la sesión y usuarios...

3. ¿Qué datos de los que no aparecen en la web http://ucm.aceptaelreto.com echas de menos?

La búsqueda de problemas por título también estaría muy bien

De forma más visual e intuitiva clasificar los problemas propuestos en cada sesión por dificultad. Quizás en vez de mostrar de forma numérica el número de ac y de envíos, mostrar alguna otra métrica más representativa de la dificultad en formato visual (eg: ac/numEnvíos, dacu, etc.)

Poder programar las sesiones sin revelar los problemas. Estadísticas de la sesión. Acceso al código de un envío

4. ¿Cada vez que accedes a http://ucm.aceptaelreto.com sigues los mismos pasos de consulta? Si no es así, ¿en qué se basan tus cambios?

Sí, suelo seguir los mismos pasos

Sí. Normalmente las consultas a la página se dirigen a la sesión actual, y rara vez a la clasificación global

Si, URL guardada en el navegador

5. ¿Qué información individual extraes de cada alumno?

Los envíos e intentos, sobre todo el último que hayan hecho

Número de ACs de cada sesión. Número de intentos de cada sesión. Nº de TLE de cada alumno.

Número de envíos y sus resultados

6. ¿Qué información resumida extraes de cada ejercicio?

El veredicto para después analizar el código del ejercicio

Nivel de dificultad basado en el número de soluciones correctas, y número de intentos por cada problema. El número de intentos normalmente lo distingo entre el número de intentos de los alumnos con un AC, y el número de intentos de alumnos sin AC (es decir, número de intentos antes de rendirse). Nº TLE

La dificultad del mismo

7. ¿Qué datos echas de menos si nos referimos a datos individuales?

Actividad en el tiempo

-

El código.

8. ¿Qué datos echas de menos si nos referimos a datos resumidos?

Estadísticas descriptivas de tasas de éxito

Clasificación de la dificultad de cada problema en relación con el resto de problemas de la sesión y con el general de todo el sitio o del resto de sesiones. Datos generales de cada problema agrupados en cada columna (nº ac, nº envios, etc)

Estadística resumen de la sesión

9. ¿Qué dispositivo utilizas para consultar la Aplicación?

Ordenador

Navegador escritorio. Android

Android y PC

10. ¿Cómo localizas un problema para añadirlo a tu espacio en http://ucm.aceptaelreto.com?

Buscando las categorías

O bien ya lo tengo hecho y clasificado, o bien utilizo la clasificación de problemas propia de acr.

Buscando en mi repositorio o utilizando la exploración por categoría

11. ¿Qué soporte encuentras más cómodo a la hora de estudiar los datos de sus alumnos? ¿Soportes digitales? ¿Soportes físicos?

Digitales

Soporte digital si es interactivo. Si los datos son estáticos (un pdf por ejemplo), soporte en papel

Digitales, tablas o excels.

12. ¿Utilizas algún tipo de ayuda en forma de gráfico o de diagrama a la hora de evaluar a tus alumnos en conjunto? ¿Y de forma individual? Si es así, ¿de qué tipo?

Prefiero los elementos de resumen como tablas o gráficos, pero también se realiza evaluación individual basada en veredictos y código enviado.

No. Pero si se generaran automáticamente los utilizaría.

No, pero me gustaría. Por ejemplo, gráfico circular con número de envíos y sus resultados.

13. ¿Qué tipo de seguridad consideras necesaria a la hora de acceder a los resultados de los problemas propuestos resueltos por los alumnos?

La que hay ahora mismo de solo poder acceder el profesor me parece correcta

Posibilidad de proteger con contraseña todo el curso. Una vez dentro del curso, la información puede ser vista por todo el mundo (como es ahora).

Que otro alumno no pueda acceder al código. Verificación de la autoría del envío.

Apéndice C - **Base de datos**

Inicial

Nombre	**Descripción**
id	ID de la categoría de los problemas
parent	ID de la categoría padre de la que derivan, puede ser NULL
numOfProblems	Número de problemas de los que se compone la categoría
totalAC	Total de aciertos en la categoría
totalC	Total de problemas resueltos en C
totalCE	Total de errores en la compilación (COMPILATION ERROR)
totalCPP	Total de problemas resueltos en C++
totalIR	Total de errores internos (INTERNAL ERROR) en la categoría
totalJAV	Total de problemas resueltos en JAVA
totalML	Total de límites de memoria (MEMORY LIMIT) en la categoría
totalOL	Total de límites en tiempo de salida(OUTPUT LIMIT) en la categoría
totalPE	Total de errores en presentación (PRESENTATION ERROR)
totalRF	Total de funciones restringidas (RESTRICTED FUNCTION)
totalRTE	Total de errores durante la ejecución (RUN-TIME ERROR)
totalSubs	Total de entregas en la categoría
totalTL	Total de límite de tiempo (TIME LIMIT) en la categoría
totalWA	Total de respuestas erróneas (WRONG ANSWER) en la categoría

Tabla 6: Category (tabla con toda información relativa a cada una de las categorías de los problemas).

Nombre	Descripción
id	ISO del país

Tabla 7: Country (tabla con los ISO de los países)

Nombre	Descripción
id	ID de la institución
country_id	ISO del país

Tabla 8: Institution (tabla que relaciona las instituciones con cada uno de los ISO de los países existentes).

Nombre	Descripción
id	ID incremental del último problema propuesto en la semana
problem	ID del problema
startDate	Fecha de inicio de la propuesta
endDate	Fecha de fin de la propuesta

Tabla 9: PastProblemOfTheWeek (tabla con los problemas propuestos cada una de las semanas)

Nombre	Descripción
internalId	ID del problema
totalAC	Total de aciertos del problema
totalPE	Total de errores de presentación (PRESENTATION ERROR)
totalWA	Total de respuestas erróneas (WRONG ANSWER)
totalTL	Total de límites de tiempo (TIME LIMIT)
totalML	Total de límites de memoria (MEMORY LIMIT)
totalOL	Total de límites de salida (OUTPUT LIMIT)
totalRF	Total de funciones restringidas (RESTRICTED FUNCTION)
totalRTE	Total de errores durante la ejecución (RUN-TIME ERROR)
totalCE	Total de errores en la compilación (COMPILATION ERROR)
totalIR	Total de errores internos (INTERNAL ERROR)
totalSubs	Total de intentos
totalDistinctUsers	Total de usuarios diferentes que han intentado hacer el problema
totalDACU	Total de usuarios diferentes que han acertado la solución
problemType	Tipo del problema
publicationDate	Fecha de publicación en la aplicación

Tabla 10: Problem (tabla con todos los problemas que componen la aplicación)

Nombre	Descripción
categoryId	ID de la categoría
problemId	ID del problema

Tabla 11: ProblemCategories (tabla que relaciona las categorías con los problemas)

Nombre	Descripción
id	ID de la entrega
problem_id	ID del problema de la entrega
user_id	ID del usuario que ha hecho la entrega
language	Lenguaje en el que se ha hecho la entrega
ranking	Ranking de la entrega
comment_id	ID del comentario
status	Estado del problema de la entrega
submissionDate	Fecha de entrega

Tabla 12: Submission (tabla con las entregas que realiza cada uno de los usuarios)

Nombre	Descripción
id	ID del comentario de entrega

Tabla 13: SubmissionComment (tabla con los ids de los comentarios de entrega)

Nombre	Descripción
id	ID del usuario
gender	Género del usuario
registrationDate	Fecha de registro del usuario
country_id	ID del país del usuario
institution_id	ID de la institución del usuario

Tabla 14: Users (tabla con los usuarios de la aplicación)

A partir de estas tablas, se creó el modelo relacional, Figura 61. Una vez que ya se tenía, el prototipo de los paneles de control exigía otra serie de entidades y atributos, por lo que se añadieron nuevas tablas a las anteriores ya citadas, dando lugar a un nuevo modelo relacional, Figura 62.

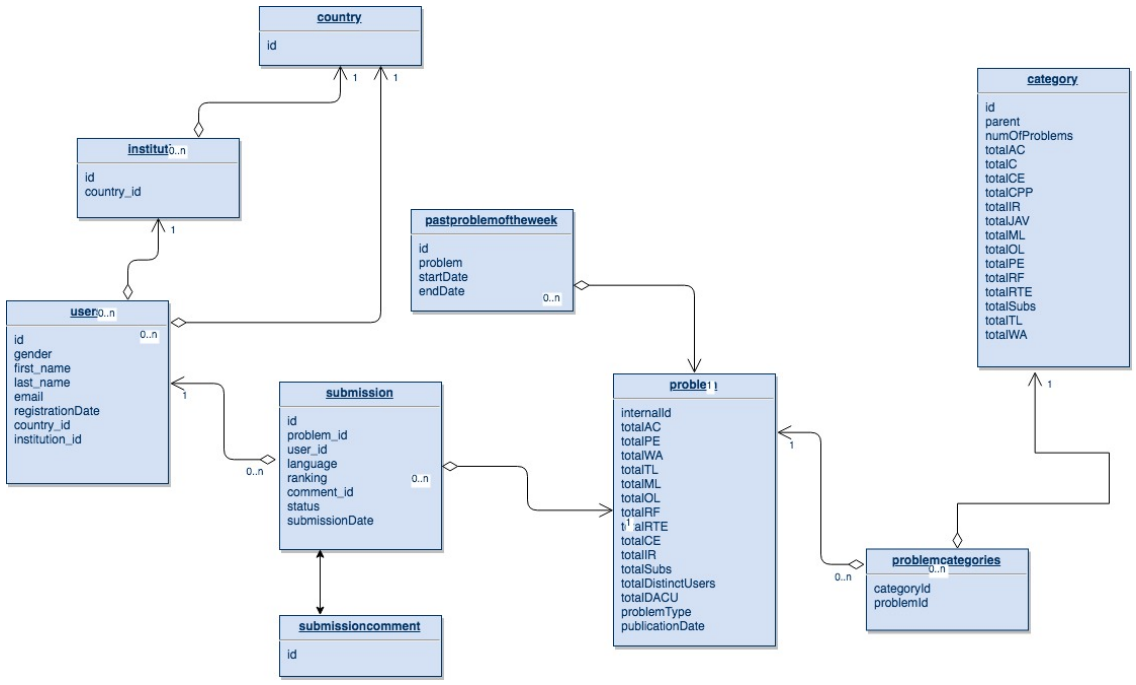


Figura 61: Modelo relacional inicial

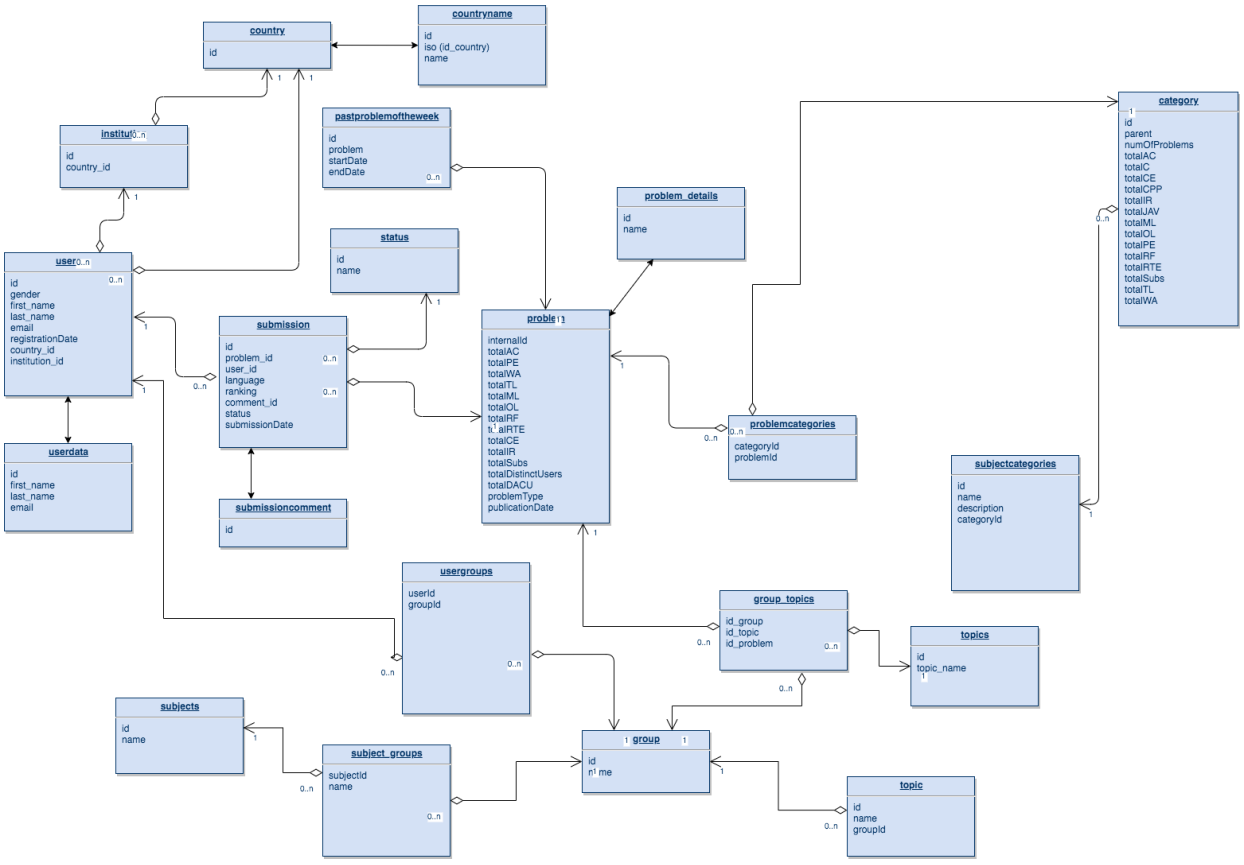


Figura 62: Modelo relacional final

Final:

La estructura final de las tablas, se compone de las anteriores definidas y las siguientes añadidas:

Nombre	**Descripción**
id	ID del país
iso	ID ISO del país (country_id)
name	Nombre del país

Tabla 15: Countryname (tabla con los países)

Nombre	Descripción
id	ID del grupo
nombre	Nombre del grupo de alumnos

Tabla 16: Group (tabla con los grupos de clases)

Nombre	Descripción
id	ID del problema
name	Nombre del problema

Tabla 17: Problem_details (tabla con los nombres de cada uno de los problemas)

Nombre	Descripción
id	ID del estado (Iniciales del estado)
name	Estado completo

Tabla 18: Status (tabla con los diferentes resultados que puede tener un problema)

Nombre	Descripción
id	ID de los temas de las asignaturas
name	Nombre de las categorías de los problemas
description	Descripción de las categorías
categoryId	ID de las categorías de los problemas

Tabla 19: SubjectCategories (tabla para relacionar los temas de cada una de las asignaturas)

Nombre	Descripción
id	ID de la asignatura
name	Nombre de la asignatura

Tabla 20: Subjects (tabla con las asignaturas)

Nombre	Descripción
subjectId	ID de las asignaturas
groupId	ID de los grupos

Tabla 21: Subject_groups (tabla que relaciona las asignaturas con los grupos existentes)

Nombre	Descripción
id	ID del tema
name	Nombre del tema
groupId	ID de los grupos

Tabla 22: Topic (tabla que relaciona los temas de cada grupo)

Nombre	Descripción
id	Id del usuario
first_name	Nombre del usuario
last_name	Apellidos del usuario
email	Email del usuario

Tabla 23: Userdata (tabla con las características de los usuarios)

Nombre	Descripción
id	ID del usuario
gender	Género del usuario
first_name*	Nombre del usuario
last_name*	Apellidos del usuario
registrationDate	Fecha de registro del usuario
country_id	ID del país del usuario
institution_id	ID de la institución del usuario

Tabla 24: Users (tabla con los usuarios de la aplicación. Se añaden nuevos campos a la tabla inicial)

Nombre	Descripción
id	ID del tema
topic_name	Nombre del tema

Tabla 25: Topics (tabla que tiene los temas de una asignatura)

Nombre	Descripción
id_topic	Id del tema
id_group	Id del grupo
id_problem	Id del problema

Tabla 26: Group_topics (tabla que relaciona los temas del grupo C con los problemas)

*nuevos campos de la tabla.

Apéndice D - **Instalación**

Este apéndice se compone de toda la información necesaria para instalar la aplicación.

Como se ha recogido en el capítulo referente a la arquitectura y a la implementación, el sistema se instala sobre un entorno LAMP.

El proyecto se compone de dos partes: los ficheros fuente que corren la aplicación y su base datos. Es por ello que en el entregable figuren éstos distribuidos en la carpeta instalación:

- Una con los ficheros fuente comprimidos en un .zip

- Otra con el archivo .sql necesario para montar la base de datos.

El proceso de instalación es sencillo y se resume en cuatro pasos:

-PASO 1: Descomprimir el .zip con los ficheros fuente. Una vez descomprimido el .zip, el resultado serán varias carpetas y ficheros (adjuntar captura con el filesystem)

-PASO 2: Subir las carpetas descomprimidas a la raíz del servidor web. Por defecto, cada carpeta incluye un fichero .htaccess para prevenir un acceso directo vía navegador.

-PASO 3: Desplegar la base de datos en el servidor. La base de datos (MySQL), se facilita en un fichero de tipo .sql, por lo que puede desplegarse con cualquier herramienta de gestión (SGBD) como phpMyAdmin. Para conocer los detalles sobre el modelo relacional de la base de datos, se puede consultar el Apéndice C - Base de datos.

-PASO 4: Configurar los "ficheros application/config/config.php" y "application/config/database.php" según los parámetros del servidor. Una vez conozcamos los detalles del servidor que alojará el proyecto, es necesario configurar los dos ficheros arriba señalados para su correcta ejecución. Ambos ficheros son auto explicativos, pero se enumeran a continuación los cambios más importantes que deben realizarse.

- config.php: Este fichero multipropósito permite configurar los aspectos más pormenorizados del proyecto, siendo la mayoría opcionales. Se encuentran aquí recogidos temas como la seguridad, la codificación de caracteres o las cookies. Para nuestro propósito, el único campo que deberemos comprobar es el relativo a la URL base del proyecto:

$config['base_url'] = $root;

Donde deberemos sustituir la raíz del servidor, por una URL personalizada si así lo deseamos. Por ejemplo:

$config['base_url'] = 'http://example.com/';

- database.php: Este fichero es el encargado de realizar la conexión con la base de datos por lo que su correcta configuración es esencial para poder levantar el sistema. Los campos relevantes son:

```
$db['default'] = [
...
'hostname' => 'localhost',
'username' => 'root',
'password' => 'yourpassword',
'database' => 'tfg',
...
];
```

Donde 'hostname', 'username', 'password', y 'database' deben corresponderse con los datos de nuestro servicio MySQL previamente desplegado.

Para cualquier configuración adicional, así como para la búsqueda de soluciones a problemas eventuales, se remite a la Guía Oficial de Instalación del framework base, el cual se mantenido inalterado para la realización de este proyecto:

https://www.codeigniter.com/user_guide/installation/index.html#

NOTA: El sistema está idealmente planificado para ejecutarse sobre un servidor Apache. Si se desea utilizar un Nginx, se recomienda adaptar el recetario que figura en la Wiki del proyecto redactado a tal efecto: https://www.nginx.com/resources/wiki/start/topics/recipes/codeigniter/

Tras la configuración anterior, el proyecto debería estar disponible y navegable a través de la URL, ya sea local o externa, definida por el servidor.

Apéndice E - **Evaluación con usuarios**

Administradores

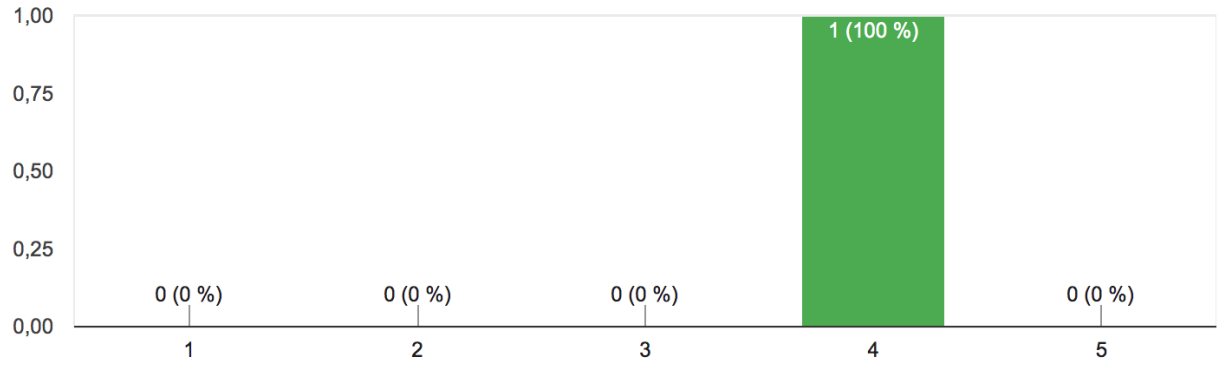
La información acerca de estas evaluaciones se encuentra en: https://docs.google.com/forms/d/1zCSNF_euPq7xKhz2id7U8bbOjgN39cI_MDfOTSQvtBI/

1- ¿Cómo accedería al listado de usuarios y al Detalle de un problema desde el Dashboard del Administrador?

Al listado de usuarios dando a la pestaña "Listado de usuarios" en la parte de arriba. A la información de un problema, depende. Si es a un problema cualquiera, yendo a la pestaña "Listado de problemas" y buscándolo por título. Si es a un problema de los mencionados en alguna lista, pulsando sobre su nombre.

Con las pestañas correspondientes y siguiendo luego el enlace del problema.

2- ¿Cómo calificaría la forma de acceder a las diferentes páginas que forman el dashboard?

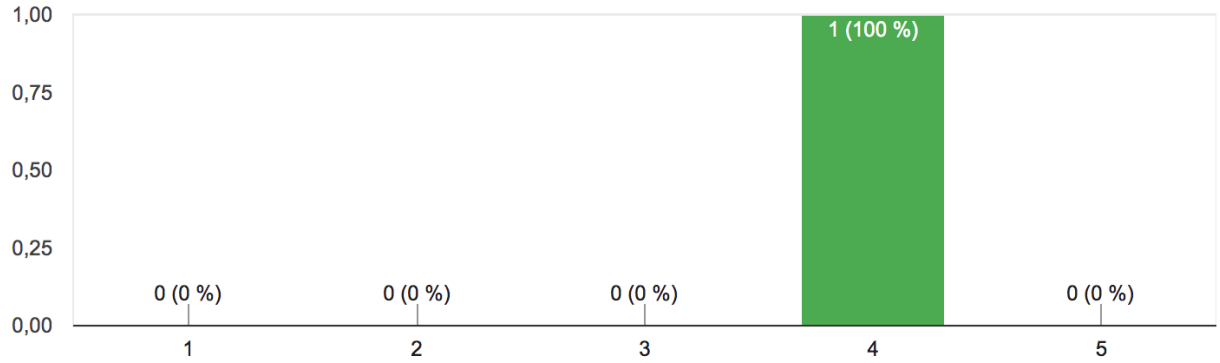


3- ¿Sabría decirme cuántos WA se han producido en todos los envíos realizados en el Dashboard del Administrador?

El 30'3% de todos los envíos (44'7 * 67'8)

En número absoluto tendría que calcularlo a mano. En porcentaje, 44.7%

4- ¿Se entienden bien las leyendas de los gráficos?

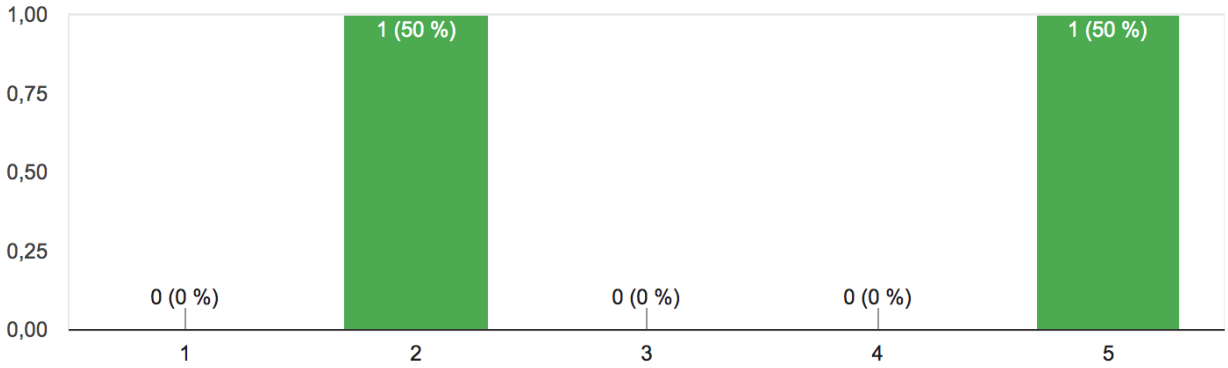


5- ¿Cuántos participantes tiene Perú?

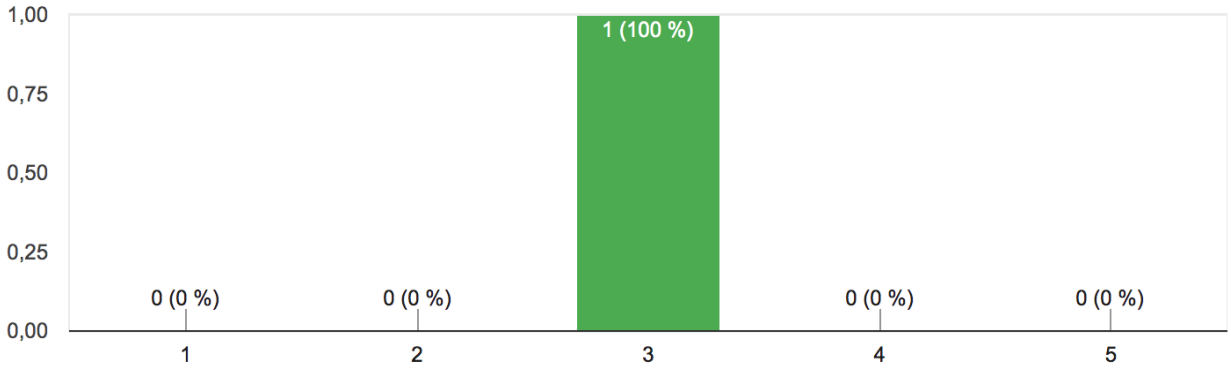
81

8

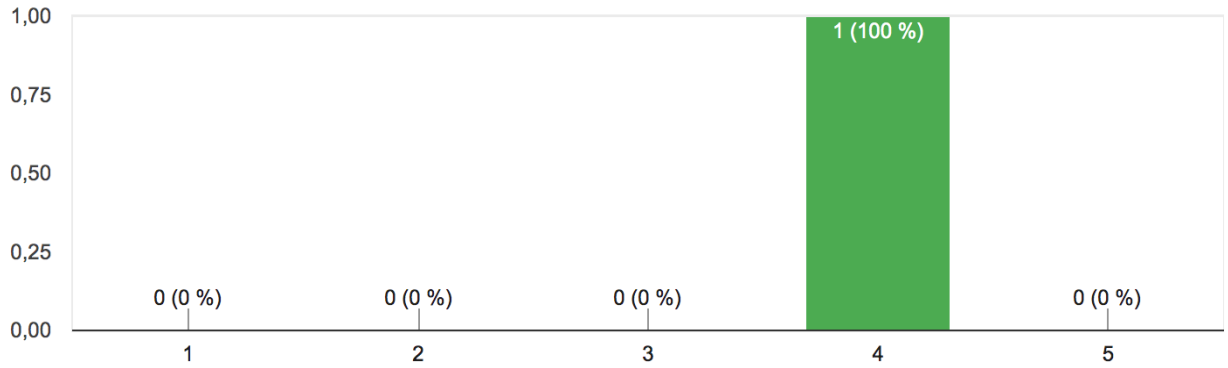
6- ¿Sabría acceder a ver el número de participantes de cada país?



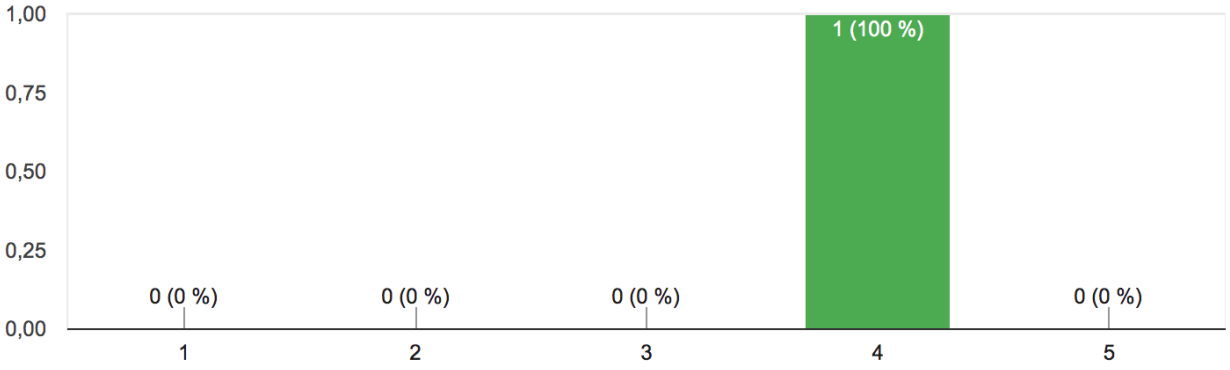
7- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Cómo valoraría la disposición de los botones?



8- En una escala del 1 (no me gusta) al 5 (me encanta), ¿Cómo valoraría el aspecto de la interfaz?



9- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Cómo valoraría la facilidad de uso de la interfaz?



10- ¿Qué es lo que más le ha gustado de la aplicación?

Ver tantos datos :-)

Lo mismo que la de la vista del profesor rellenado en el otro formulario. De todas formas, hay poco de "administrador" en el dashboard. Prácticamente todo es de interés para un visitante cualquiera. Excepto la parte de arriba del número de usuarios nuevos y sobre todo las estadísticas de los Internal Errors, lo demás puede ser curioso de ver para cualquiera.

11- ¿Qué es lo que menos le ha gustado de la aplicación?

Es un poco lenta

Que dice "Dashboard admintrador" pero no hay información sensible del sistema útil para el administrador. Lo mismo que la de la vista del profesor rellenado en el otro formulario.

12- ¿Qué mejoraría?

Me resulta extraña la relación entre las pestañas ("Dashboard administración", "Listado usuarios", "Listado problemas") y las opciones de navegación de la izquierda (IR A). Visualmente está fuera de las pestañas, por lo que parece que jerárquicamente está por encima. Dado que el lado izquierdo es navegabilidad de la pestaña seleccionada (y desaparece, de hecho, si la pestaña no lo incluye) creo que debería estar dentro de la pestaña actual, y no fuera.

Añadir información para el administrador :) Y lo mismo que la de la vista del profesor rellenado en el otro formulario.

Profesores

1- ¿Cómo accedería al listado de alumnos y al Detalle de un problema del curso desde el Dashboard del profesor?

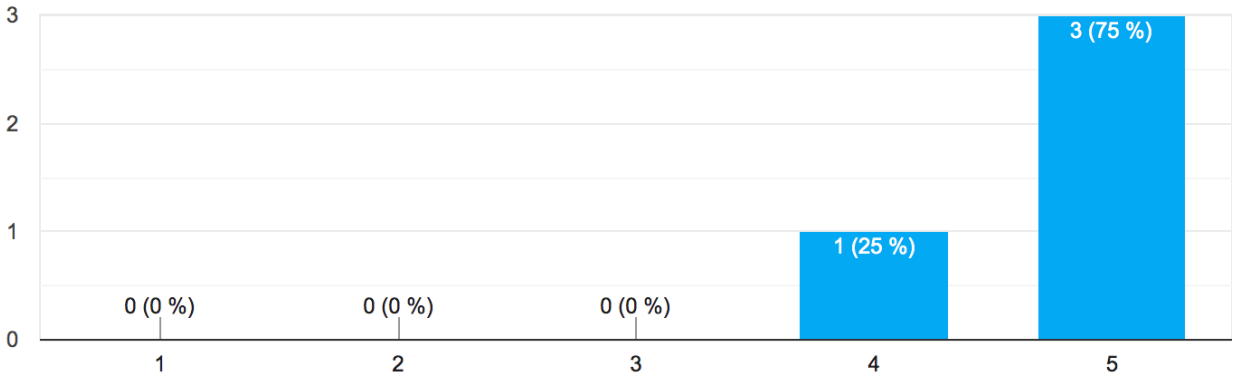
Pulsando en la pestaña de Listado de alumnos y de Listado de problemas

Desde las pestañas de listado de alumnos y listado de problemas. En el último caso usaría despues el buscados

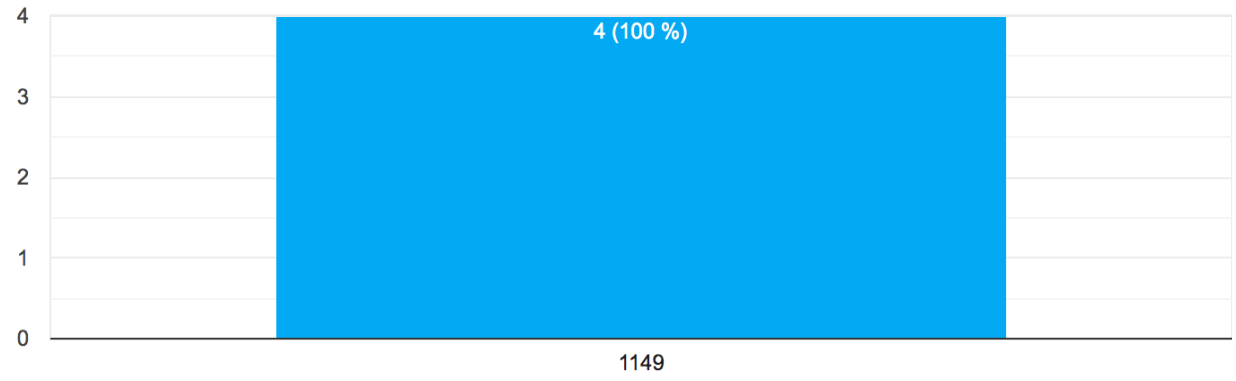
Con la pestaña Listado de alumnos, y con la pestaña Listado de problemas > Enlace del problema correspondiente.

Con las pestañas correspondientes

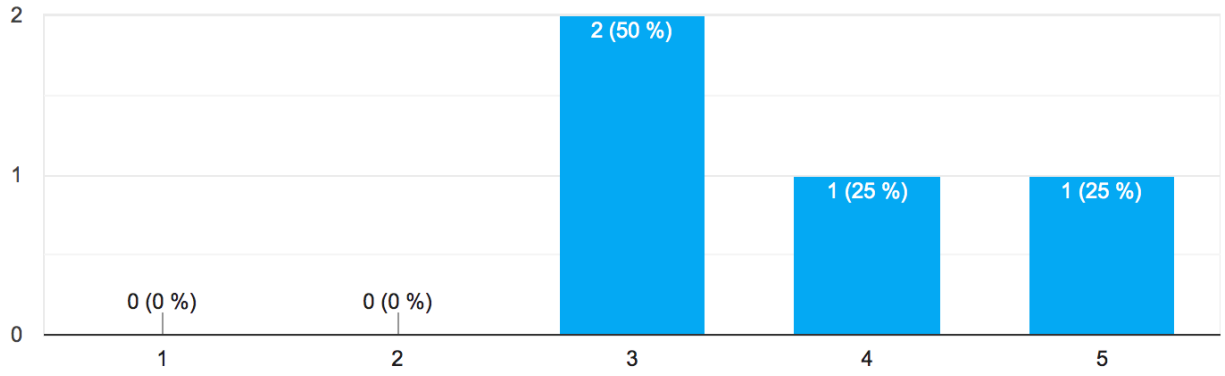
2- ¿Cómo calificaría la forma de acceder a las diferentes páginas que forman el dashboard?



3- ¿Podría decirme cuántos TLE se han producido en los envíos de nuestros alumnos?



4- ¿Se entiende bien la leyenda de Aceptados/Fallidos?



5- Acceda al Detalle del alumno y vaya al área de Problemas Intentados, ¿qué significa cada uno de los colores usados en esa área?

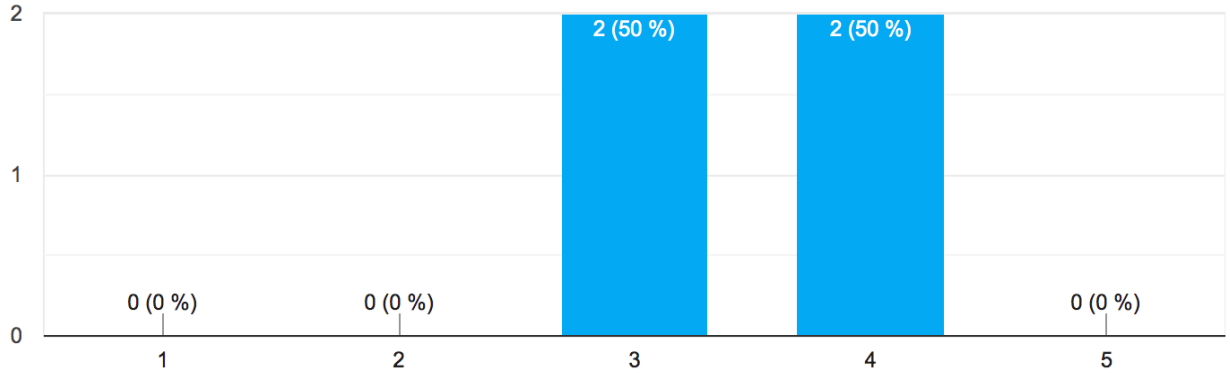
Entiendo que el color rojo es que ninguna de las entregas del ejercicio es AC y verde supondría que es que existe alguna entrega del ejercicio que está AC: pero tengo dudas porque mas abajo en el ranking de problemas hay un envío de un problema, el 39, que está en verde pero tiene un PE pero no hay ningún otro envío AC (Eudora Altenwerth).

Verde que al menos hay un AC, rojo ningún AC

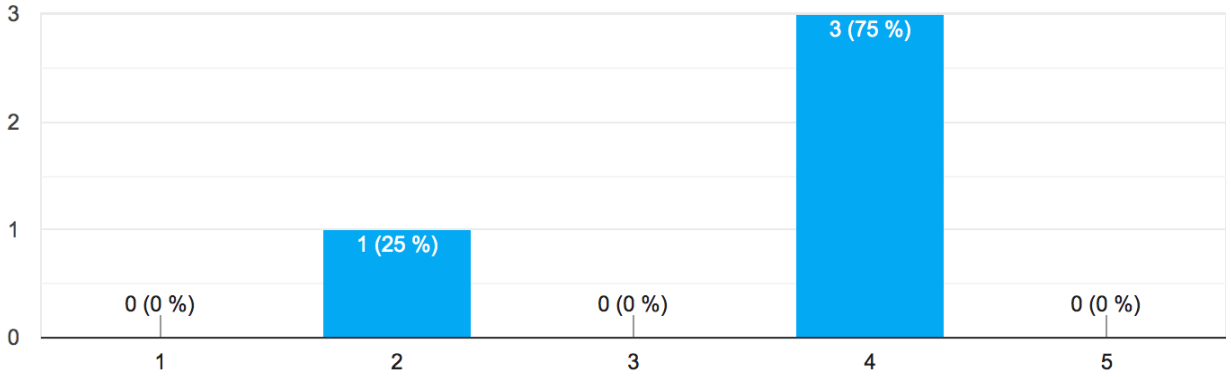
Cuáles ha conseguido y cuales no. Aunque los colores elegidos no son muy amables con los usuarios daltónicos.

Rojo que ha fallado y todavía no lo ha conseguido y verde que sí ha conseguido resolver el problema.

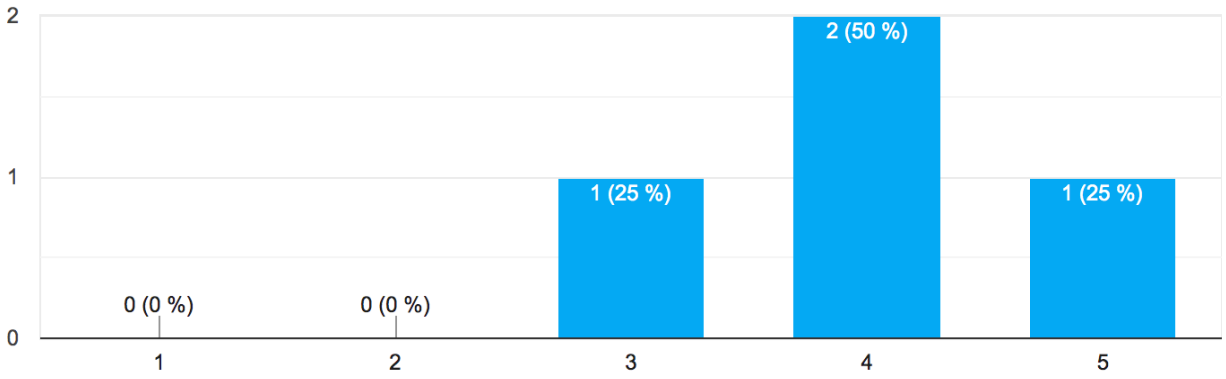
6- ¿Se entiende el área de problemas intentados?



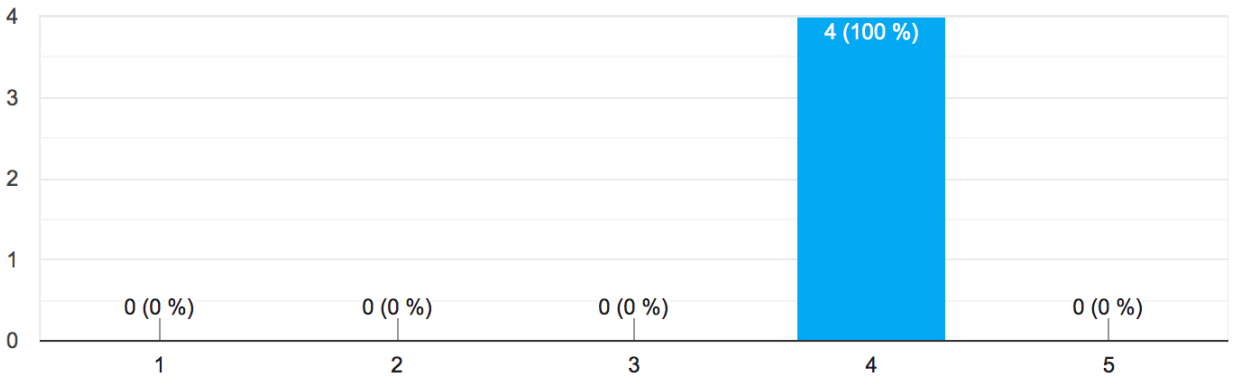
7- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Cómo valoraría la disposición de los botones



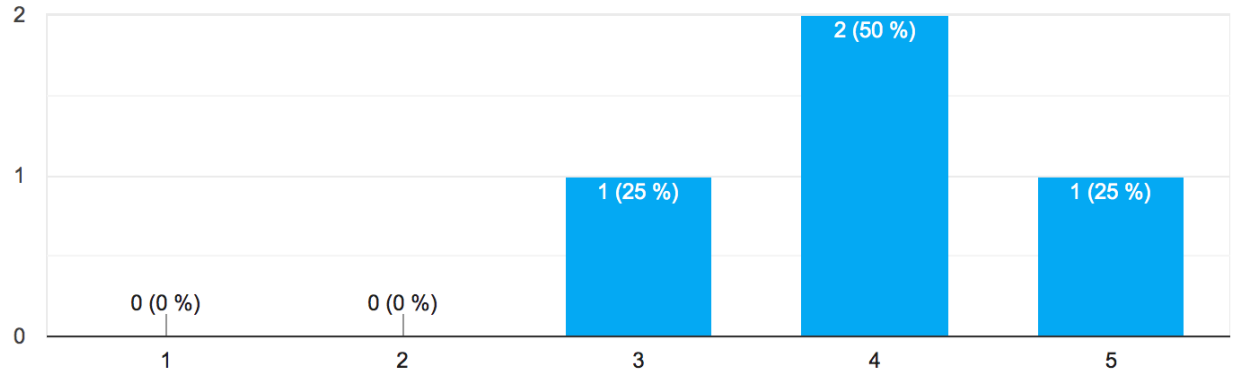
8- En una escala del 1 (no me gusta) al 5 (me encanta), ¿Cómo valoraría el aspecto de la interfaz?



9- En una escala del 1 (muy difícil) al 5 (muy fácil), ¿Cómo valoraría la facilidad de uso de la interfaz?



10- En una escala del 1 (nunca) al 5 (siempre), ¿Recomendaría esta aplicación a otros profesores?



11- ¿Qué es lo que más le ha gustado de la aplicación?

La disposición del menú a la izquierda para ir accediendo a los distintos apartados de la pestaña

El análisis de los datos que realiza que permite ver el trabajo de los alumnos fácilmente. La interfaz es muy intuitiva.

Es bastante intuitiva, aunque tiene algunas cosas que desconciertan demasiado. La parte de búsqueda en las listas de problemas, aunque desconcierte que si buscas "255" salga también el problema 858 porque hay un 255 en las estadísticas...

Toda la información a la que se puede acceder, y que esté separada tanto por alumnos como por problemas.

12- ¿Qué le ha gustado menos en la aplicación?

Tiempo de carga

Ver mejoras

Los botones de la izquierda que aparecen en algunas páginas (IR A, VOLVER A). No se entiende qué es. Hay cosas que no se entienden. Ejemplo: en el ranking de problemas de un usuario, ¿qué es el verdicto, si la fila representa todos los envíos al problema? La velocidad El tamaño en muchas ocasiones es demasiado grande; una paginación de 10 problemas puede ser demasiado poco. Se pueden añadir más, sobre todo si cada fila ocupa menos. Los colores para daltónicos

Que en el gráfico de Aceptados/Fallidos los diferentes segmentos no estén ordenados como en la leyenda.

13- ¿Qué mejoraría?

Hay alumnos duplicados Tarda mucho en cargar Los alumnos del ranking no coinciden con el listado de alumnos No entiendo AC Totales/Envios Totales. Si aparece por ejemplo 1/2 espero ver dos envíos, uno AC y otro no AC pero no los veo. No se si es porque no se ha pretendido que haya datos congruentes en la prueba. Tampoco entiendo la evolución de envíos del alumno. Algunos tienen más de 4000 envíos pero solo aparecen 6 en el listado de últimos envíos.

Hay que añadir un botón de salida bien visible. Las aplicaciones de las que el usuario no sabe cómo salir son muy estresantes. Comprobar el correcto funcionamiento de: 1. En el ranking de usuarios de la pantalla principal la primera y la tercera alumna aparecen repetidas 2. En la gráfica de envíos de la pantalla principal los aceptados acaban en diciembre cuando en los últimos envíos hay aceptados más recientes 3. En la caja de AC totales de la pantalla principal aparecen 2159, pero si vamos al ranking de usuarios y sumamos los AC totales que aparecen suman más. 4. He entrado como profesor y he ido a ver la información de un alumno en el listado de alumno. En la parte izquierda aparece un botón (en la parte inferior) en la que puedes acceder al área de administrador. No pide nada simplemente apareces allí. Pulse por equivocación pensando que era para volver al panel principal del profesor. Debería estar mas claro, quizá separando en esta parte lo que corresponde a profesor de lo que corresponde a administrador y quizá pedir confirmación o algo así.

Todo lo de la pregunta anterior. Poder ordenar las tablas por distintos criterios. Por ejemplo, en el ranking de problemas, ordenar por fecha del último envío, por intentos, por id del problema, ...

La velocidad de respuesta

BIBLIOGRAFÍA

Acepta el reto. (n.d.). From https://www.aceptaelreto.com/

Bootstrap. (n.d.). From https://getbootstrap.com/

Codeforces. (n.d.). From https://codeforces.com/

Codeigniter. (n.d.). From https://codeigniter.com/

Cooper, A., Reimann, R., & Cronin, D. (2007). *About face 3.* Wiley Publishing, Inc.

Few, S. (2006). *Information dashboard design.* O'Reilly.

Font Awesome. (n.d.). From https://fontawesome.com/

Github. (n.d.). From https://github.com/

Google. (n.d.). *Formularios Google*. From https://www.google.es/intl/es/forms/about/

Gothelf, J. (2013). *Lean UX: Applying lean principles to improve user experience.* O'Reilly.

Gulp. (n.d.). From https://gulpjs.com/

Gómez-Martín, P. P., & Gómez-Martín, M. A. (n.d.). ¡ Acepta el reto!: juez online para docencia en español. *Actas de las Jornadas sobre Enseñanza Universitaria de la Informática, 2017, vol. 2, p. 289-296* .

JQuery. (n.d.). From https://jquery.com/

Kattis. (n.d.). From https://www.kattis.com/

Krug, S. (2006). *Don't Make Me Think! A Common Sense Approach to Web Usability*. New Riders.

Laravel. (n.d.). From https://laravel.com/

Myer, T. (2008). *Professional Codeigniter.* Wiley Publishing.

Nielsen, J. (2001). *First rule of usability? Don´t listen to users.*

Parallels. (n.d.). From https://www.parallels.com/es/

ProgramaMe. (n.d.). From http://www.programa-me.com/2020/reg/

Revilla, M., Manzoor, S., & Liu, R. (2010). *A New Learning Paradigm: Competition Supported By Technology*. Sello Editorial.

Sass. (n.d.). From https://sass-lang.com/

Selectize. (n.d.). From https://selectize.github.io/selectize.js/

Sparkline. (n.d.). From https://omnipotent.net/jquery.sparkline/#s-about

Spoj. (n.d.). From https://www.spoj.com/

Symfony. (n.d.). From https://symfony.es/

Tabler. (n.d.). From https://tabler.io/

Tablesorter. (n.d.). From https://mottie.github.io/tablesorter/docs/

Twig. (n.d.). From https://twig.symfony.com/

uHunt UVa. (n.d.). From https://uhunt.onlinejudge.org/

URI Online Judge. (n.d.). From https://www.urionlinejudge.com.br/judge/en

UVa - onlinejudge. (n.d.). From https://onlinejudge.org/

Vectormap. (n.d.). From https://jvectormap.com/