Universidad Complutense de Madrid

Facultad de Informática

Desarrollo de una APP de autoadministración de test cognitivos para el diagnóstico precoz de la demencia

***

Development of an APP for the self-administration of cognitive tests for early diagnosis of dementia

Autores **Arantxa Patricia Brock Rabines**

Estephany Jhoselin Oscco Carbajal

Andrea Sthephanie Janampa Zanabria

Jaime Palazón Othon

Director de proyecto **José Luis Ayala Rodrigo**

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Junio de 2021

Índice general

Lista de Acrónimos

APP Aplicación informática. *Application*.

API Interfaz de programación de aplicaciones. *Application Programming Interfaces*.

BBDD Base de datos.

HTTP Protocolo de transferencia de hipertextos. *Hypertext Transfer Protocol*.

FK Clave foránea. *Foreign Key*.

MVC Modelo-vista-controlador.

EC2 Amazon Elastic Compute Cloud.

RDS Amazon Relational Database Service.

AWS Amazon Web Services.

SQL Lenguaje de consulta estructurada. *Structured Query Language*.

UI Interfaz de usuario. *User Interface*.

IDE Entorno de desarrollo integrado. *Integrated Development Environment*.

TIC Tecnologías de la información y la comunicación.

IA Inteligencia Artificial.

SO Sistema operativo.

iOS Sistema operativo iphone. *iPhone Operating System*.

ORM Mapeo objeto-relacional. *Object-Relational mapping*.

DCL Deterioro cognitivo leve.

EA Enfermedad de *Alzheimer*.

VOSP Percepción visoespacial de objetos. *Visual Object and Space Perception Battery*.

FCSRT Free and Cued Selective Reminding Test.

ROCF Test de la figura compleja del Rey. *The Rey–Osterrieth complex figure Test*.

TDAH Trastorno por Déficit de Atención e Hiperactividad.

Palabras clave

Palabras clave en Español

- Aplicación.
- Dispositivo móvil.
- Usuario.
- Evaluación cognitivo.
- Enfermedad neurodegenerativa.
- Autoadministración.
- Paciente.
- Deterioro cognitivo.

Keywords in English

- Application.
- Smartphone.
- User.
- Cognitive evaluation.
- Neurodegenerative disease.
- Autoadministration.
- Patient.
- Cognitive impairment.

Resumen

El presente trabajo de fin de grado nace de la necesidad de diseñar herramientas digitales que sean capaces de auto-administrar test de evaluación cognitiva para la detección precoz de enfermedades neurodegenerativas; estas herramientas deben tener como principal propósito que los usuarios puedan realizar los test desde sus propios dispositivos tras una recomendación previa por parte de un especialista.

Para ello se han definido una serie de objetivos, incluyendo, el análisis y estudio de los principales test utilizados actualmente en los departamentos de neurología, la adaptación digital de los test finalmente seleccionados, así como, la implementación y desarrollo de los mismos. En concreto realizaremos un desarrollo híbrido con la intención de diseñar una aplicación cuyo acceso esté disponible para la mayoría de personas, es decir, desarrollada para dispositivos de tipo smartphone con sistema operativo Android o iOS, pues estos destacan por ser los más utilizados.

Además, contaremos con la colaboración del departamento de neurología del Hospital Clínico San Carlos durante el proceso completo de elaboración del proyecto. Esta contribución nos aportará el criterio médico necesario para la adaptación de los test que conforman nuestra aplicación, así como la forma en la que presentaremos nuestro diseño gráfico de las pruebas al paciente y otras características para poder dotar a la aplicación de un sistema que permita almacenar y procesar las respuestas del paciente automáticamente.

Una vez realizado todo el proceso de desarrollo e implementación, realizaremos un despliegue de la APP en dispositivos iOS y Android, que se probará primeramente en población sana. Para esta administración de los test a los primeros usuarios contaremos con la colaboración del neuropsicólogo del departamento. De esta forma, se podrá validar el resultado de la especificidad del diagnóstico a través de esta herramienta, esperando gran similitud con la herramienta no auto-administrada; de igual manera podremos realizar un estudio y análisis de los primeros resultados obtenidos suministrando un test de usabilidad previamente diseñado, con las preguntas necesarias para la compresión de la interacción entre la interfaz y el usuario.

Abstract

The present project arises from the need to design digital tools that are capable of self-administering cognitive assessment tests for the early detection of neurodegenerative diseases. These tools should have as main purpose that users can run the tests from their own devices after a previous recommendation by a specialist.

In order to manage this purpose, some objectives have been defined, including the analysis and the study of the main tests currently used in the departments of neurology, the digital adaptation of the final tests selected, as well as the implementation and development of them.

Specifically, we will carry out a hybrid development with the intention of designing an application whose access is available to most people. An application developed for smartphone-type devices with Android or iOS operating system, which stand out for being the most used.

In addition, we will have the collaboration of the neurology department of the Clínico San Carlos Hospital during the entire process of this project. This contribution will provide us with the necessary medical criteria for the adaptation of the tests that make up our application, as well as the way in which we will present our graphical design of the tests to the patient and other features to be able to provide the application with a system that allows to store and process the patient's responses automatically.

Once the entire development and implementation process has been completed, we will deploy the APP on iOS and Android devices, which will first be tested on healthy populations. For this administration of the tests to the first users we will count on the collaboration of the neuropsychologist of the department. In this way, it will be possible to validate the result of the specificity of the diagnosis through this tool, expecting great similarity with the tool not auto-managed. In the same way, we will be able to carry out a study and analysis of the first results obtained by providing a previously designed usability test, with the required questions for the compression of the interaction between the interface and the user.

Capítulo 1

Introducción

El incremento de la esperanza de vida en el ser humano, junto con el aumento demográfico, ha propiciado la aparición de un mayor número de casos de enfermedades neurodegenerativas. Entre otras cosas, este aumento produce un atraso a la hora de poder obtener un diagnóstico por falta de asistencia médica. Por ello, la intención de este trabajo es ayudar, en la manera de lo posible, a mitigar dicho problema.

1.1. Enfermedades neurodegenerativas

Este tipo de enfermedades forman una parte extensa dentro del campo de la medicina, específicamente, en la rama de la neurología. Además, se caracterizan comúnmente por ser enfermedades de causa desconocida, con desarrollo progresivo de sus síntomas, que, a su vez, muestran la desintegración gradual de una o varias partes del sistema nervioso del ser humano. Asimismo, a estas características, se le suma una disminución de la funcionalidad e independencia personal.
Las personas que padecen alguna de estas enfermedades requieren atención total y continuada, que justifica la necesidad de que los pacientes sean tratados en unidades específicas. [1]

Estas enfermedades pueden clasificarse dependiendo de las manifestaciones clínicas que el paciente presenta; entre ellas se pueden distinguir las que muestran principalmente un síndrome demencial, como el Alzheimer; las que se exponen con trastornos de movimiento y postura, en el caso de Parkinson; las que presentan ataxia progresiva, es decir, falta de control muscular o de la coordinación de movimientos voluntarios, como comer, en el caso de la Ataxia Cerebelosa; o aquellas que muestran debilidad y atrofia muscular, por ejemplo la Esclerosis Lateral Amiotrófica; y otras muchas enfermedades con diferentes síntomas.

Todas estas enfermedades neurodegenerativas, no tienen un tratamiento que actúe sobre la causa que lo origina, sino más bien se aplican tratamientos paliativos, es decir, no alcanzan cura. [2]

1.2. Por qué se propone este TFG

La detección y diagnóstico precoz de las enfermedades neurodegenerativas, según diferentes estudios internacionales, se consideran, actualmente, un problema significativo de salud pública aún

no resuelto, ya que reflejan un retraso medio de 8 a 32 meses, entre la aparición de los primeros síntomas y el diagnóstico de la propia enfermedad.

Un diagnóstico adelantado supondría, entre otras muchas cosas, aumentar la compresión del problema teniendo la posibilidad de acceso a un tratamiento y apoyo social, sin embargo, hasta el momento, no se ha propuesto en ningún país la detección sistemática en pacientes que sufren este tipo de enfermedades en Atención Primaria, a pesar de que el porcentaje de casos no diagnosticados sea muy elevado. [3]

La propuesta de este TFG, en el contexto actual, pretende ayudar con los problemas descritos, esto es, permitir que la realización de test cognitivos no sea única y exclusivamente en un centro médico, ya sea centro de salud u hospital. Sino facilitar a los pacientes para la autoadministración de dichas pruebas desde sus propios domicilios, lo que también supondría una solución a la actual demora existente de obtener el diagnóstico de la enfermedad, por lo que aceleraría también la presión asistencial.

1.3. Objetivos de este TFG

El objetivo principal de este Trabajo de Fin de Grado es la implementación y desarrollo de una aplicación móvil híbrida, es decir, que esté disponible para dispositivos con sistema operativo Android o iOS, que englobe una serie de test de evaluación cognitiva.

Para ello, este trabajo se hace en colaboración con el departamento de Neurología del Hospital Clínico San Carlos, quiénes nos mostraron las diferentes pruebas con las que trabajaban y nos facilitaron su adaptación para dispositivos móvil.
Así pues, se almacenan las respuestas de cada una de las pruebas realizadas a los pacientes; y, a su vez, se obtendrán de ellas puntuaciones que posteriormente serán estudiadas por dichos especialistas. Estas puntuaciones proporcionarán una estimación al neurólogo a la hora de hacer el diagnóstico del paciente.
Asimismo, otro de los objetivos de esta aplicación es permitir la autoadministración de las pruebas a los pacientes de una forma cómoda y sencilla.

Por otro lado, cave remarcar el desafío que implica llevar a cabo un proyecto software en grupo desde sus inicios, pues esto conllevó una detallada planificación de todas sus fases, desde el estudio, hasta la implementación y las pruebas de este. Lo que ha hecho necesaria la organización del equipo completo, planificación en detalle del tiempo, los recursos disponibles y la puesta en práctica de los conocimientos adquiridos a lo largo de los años de la carrera.

1.4. Estructura de la memoria

El presente documento está estructurado en una serie de capítulos cuyos contenidos se describen a continuación:

- **Introducción:** describe el problema explicado en el apartado anterior, así como el principal objetivo por el cual se realiza este trabajo.
- **Estado del arte:** contiene una descripción detallada de las investigaciones y publicaciones realizadas en el ámbito de nuestro trabajo, es decir, dentro del campo de las enfermedades neurocognitivas y las herramientas utilizadas en la actualidad para la realización de test cognitivos.
- **Metodología:** describe el resultado final de la aplicación. A su vez, se explican detalladamente las tecnologías y herramientas utilizadas para el desarrollo de nuestra aplicación, así como los pasos y organización del equipo a la hora de realizar el trabajo.
- **Test y evaluación de usabilidad:** presenta un formulario de usabilidad completado por los usuarios tras el uso de la APP. Así como una recopilación de las diferentes respuestas que se han obtenido.
- **Conclusiones y líneas futuras:** capítulo en el que se presentan las conclusiones del trabajo realizado, así como las posibles mejoras que se podrían desarrollar en la aplicación tras ser presentada.

Capítulo 2

Estado del arte

Dentro de las enfermedades neurodegenerativas nos encontramos con el deterioro cognitivo leve o ligero. Este tipo de deterioro se presenta cuando hay disminución de las funciones cognitivas, pero estas no se consideran lo suficientemente severas como para ser consideradas demencia. Actualmente, el deterioro cognitivo leve está infradiagnosticado, por lo que el verdadero reto supone detectarlo en su fase inicial ya que es un proceso que en la mayoría de los casos evoluciona de forma gradual y evoluciona hacia la demencia, además en esta fase temprana puede resultar difícil de distinguir de otro tipo de situaciones como el envejecimiento normal, la depresión o un bajo nivel cultural.

En la actualidad, el diagnóstico médico de demencia se fundamenta en criterios clínicos. Así pues, gracias a varios estudios se sabe que este tipo de enfermedades neurodegenerativas, como el EA, pasan por una fase previa silenciosa, llamada fase preclínica, y, que, además, los daños cerebrales empiezan años antes de que se manifiesten los primeros síntomas.

Los criterios de diagnóstico del DCL, usados en la práctica diaria, sobre todo, en los centros de atención primaria donde suele empezar el diagnóstico de pacientes con algún tipo de enfermedad neurodegenerativa, la historia clínica y la evaluación neuropsicológica, son las herramientas más eficaces [4].

Este tipo de evaluaciones requiere de un tiempo médico que choca con la realidad de casi todas las consultas de Atención Primaria y con gran parte de las consultas ambulatorias de neurología. El tiempo medio por consulta de Atención Primaria en Europa es de diez minutos, y alrededor de cinco minutos en España. De esta falta de tiempo médico surge la necesidad de elaborar test sencillos y rápidos que permitan detectar en pocos minutos, y con una medida objetiva, a aquellas personas que puedan padecer una demencia o DCL, o también a aquellas personas que sean dudosas de padecerlo y requieran de una derivación para una valoración especializada. La tarea de proporcionar en pocos minutos una valoración de las capacidades cognitivas de una persona es complicada, y, quizás, por eso el número de test propuestos y estudiados en los últimos años es enorme [3].

Estos test tienen como característica principal que están diseñados para la detección y cribado de

demencia y es importante recordar que no están hechos para sustituir el diagnóstico clínico. Otras características que deben tener estos test son:

- Ser fácil de administrar.
- Ser aceptado por los pacientes.
- Ser fácil de puntuar por los especialistas.
- Al evaluar debe hacerlo de forma independiente al idioma o nivel educativo.
- Tener una buena validez de criterio.

Para la elaboración de esta App hemos recabado información de los principales test cognitivos breves que se utilizan actualmente y que buscan medir de manera objetiva el rendimiento en una tarea concreta. Entre otras cosas se ha tenido en cuenta tanto la facilidad para comprender las instrucciones como el tiempo necesario para su realización (casi siempre mayor cuanto más grave es el deterioro cognitivo). Los test principales en los que se ha realizado el estudio son los mencionados en el siguiente punto (2.2).

2.1. Test de evaluación cognitiva

Las actuales baterías de test cognitivos son una herramienta valiosa en el diagnóstico de enfermedades neurodegenerativas. Estas pruebas utilizadas en medicina, especialmente en el campo de la neuropsicología, realizan un test de cribado cognitivo para la detección del deterioro, demencia y otras enfermedades neurológicas. Por otro lado, ayudan a evaluar su gravedad, y, además, se establecen los cambios cognitivos a lo largo del tiempo para poder documentar la respuesta del individuo al tratamiento.

Se tratan de pruebas que han de cumplir una serie de características y han de estar validadas tras la evaluación de un especialista. Asimismo, sirven de ayuda para la obtención de un futuro diagnóstico del paciente que las realiza. [3]

La finalidad de los test cognitivos consiste en medir objetivamente el rendimiento del paciente en las diferentes tareas que se le están evaluando. Se considera una parte de exploración neurológica, que aportan información importante sobre el paciente, como la atención, la facilidad para comprender instrucciones y el tiempo que le ha sido necesario para su realización.

Las diversas áreas cognitivas que se evalúan principalmente en estas pruebas son las siguientes:

- Orientación temporal y espacial.
- Memoria inmediata y diferida.
- Atención.
- Comprensión.

- Ejecución.
- Lenguaje y praxias.

Cada una de las pruebas tiene sus respectivas instrucciones que se le contarán al paciente o que él mismo deberá leer. Con ello, también, se puede saber el grado de atención del paciente.

A continuación, se explicará uno de los test utilizados actualmente en el Hospital Clínico San Carlos, el Test de la Figura Compleja de Rey.
Esta prueba consiste en la copia y reproducción de memoria de una figura geométrica compleja. Su objetivo es evaluar la organización perceptual y la memoria visual de personas con lesiones cerebrales, como deficiencia de memoria y capacidad visoconstructiva. Aunque actualmente también es utilizado para la evaluación del TDAH.

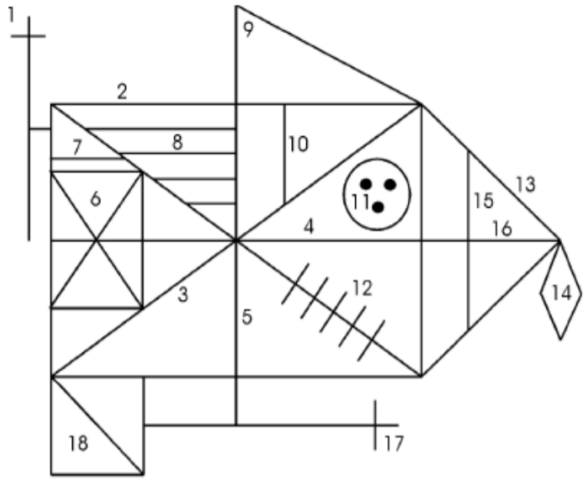


Figura 2.1: Figura geométrica del Test del Rey

El test es supervisado y cronometrado por un especialista. Se divide en dos tipos de tareas:

- **Fase de copia:** el sujeto debe copiar el modelo de la figura geométrica que se le muestra. Se le indica al paciente que la copia no ha de ser exacta, pero se le pide prestar atención a los detalles y las proporciones de la figura mostrada. En esta primera fase se le da al participante una hoja en blanco y un lápiz de color. Este empieza a copiar el dibujo, y cada vez que termina una parte del dibujo, el especialista le da otro lápiz de color para que continúe. A su vez, el especialista anotará el orden en el que el sujeto ha dibujado y el tiempo que tarda en realizar esta fase de copia. Si durante esta primera fase el paciente cambia la posición del modelo, el especialista deberá volverlo a poner en su posición inicial.
- **Fase de reproducción de memoria:** tras una pausa de máximo 3 minutos tras la ejecución de la primera fase, sacando de la vista el modelo y la copia, se le pide al sujeto que vuelva a reproducir el modelo mostrado anteriormente, sin volvérselo a mostrar, es decir, de memoria. Se le vuelve a dar al participante una hoja en blanco, pero esta vez un lápiz ordinario y se apunta el tiempo que tarda en realizar esta segunda fase.

CRITERIO	PUNTOS
Correcto y bien colocado	2
Correcto y mal colocado	1
Mal colocado pero reconocible	0.5
Elemento deformado o incompleto	0
Máximo total de puntos	36

Cuadro 2.1: Tabla de puntuaciones del Test del Rey

Tras la realización de las dos fases el especialista procederá a la evaluación de la prueba. Para ello se tienen en cuenta los siguientes elementos:

1. Cruz vertical.
2. Rectángulo grande.
3. Cruz diagonal.
4. Horizontal de rectángulo grande.
5. Vertical de rectángulo grande.
6. Rectángulo pequeño.
7. Pequeña horizontal encima del rectángulo pequeño.
8. Cuatro líneas paralelas.
9. Triángulo pequeño encima del rectángulo grande.
10. Pequeña vertical dentro del rectángulo grande.
11. Círculo con tres puntos.
12. Cinco líneas paralelas.
13. Lados del triángulo grande pegado al rectángulo grande.
14. Diamante.
15. Vertical dentro del triángulo grande.
16. Horizontal dentro del triángulo grande.
17. Cruz horizontal.
18. Cuadrado pegado al rectángulo grande.

Tras identificar los 18 elementos descritos anteriormente, se aplican a cada uno de ellos las puntuaciones siguientes:

Independientemente de la exactitud de los dibujos realizados por el paciente, los resultados obtenidos se pueden reproducir a los tipos siguientes:

I. **Construcción sobre el armazón:** el sujeto comienza por el rectángulo central, sobre el que se apoya y continúa con el resto de los elementos de la figura.

II. **Detalles englobados en un armazón:** el paciente empieza por algún detalle anexado al rectángulo central.

III. **Contorno general:** el sujeto comienza dibujando el contorno de la figura, continuando después con los detalles interiores.

IV. **Yuxtaposición de detalles:** el sujeto une los detalles de la figura, unos sobre otros.

V. **Detalles sobre un fondo confuso:** el sujeto reproduce un dibujo poco o nada estructurado, donde apenas se reconoce el modelo, pero sí ciertos detalles del mismo.

VI. **Reducción a un esquema familiar:** el sujeto asocia la figura a elementos que le resultan familiares.

VII. **Garabatos:** el sujeto reproduce un dibujo en el que no se pueden reconocer los elementos del modelo ni su forma global.

Como se muestra en [5]

2.2. Test Aplicados a nuestro estudio

Antes de comenzar con el desarrollo de nuestra aplicación, se realizó el estudio de los principales test con los que actualmente se trabaja en el departamento de neurología del Hospital Clínico San Carlos. Aunque una parte de las pruebas se siguen elaborando de modo tradicional, es decir, a papel, también utilizan el Vienna Test System, que es una herramienta TIC diseñada para la evaluación de habilidades cognitivas, memoria, atención, habilidades motrices, etc.

En nuestro caso hicimos uso de la herramienta para realizar un total de 13 test en un periodo de tiempo de dos semanas que se llevó a cabo con un ordenador con teclado especial para la realización de las diferentes pruebas. Esto sirvió para ponernos en contexto y entender la metodología que conlleva este tipo de pruebas; y, así, poder plasmarlo y adaptarlo a una aplicación para dispositivo móvil. Del estudio y realización de estos test, nos quedamos con las principales ideas de alguno de ellos para tomarlos como referencia y adaptación para nuestra aplicación. Los más importantes fueron:

2.2.1. VOSP Test

El VOSP Test sirve para explorar y evaluar la percepción de objetos y del espacio. Con este modelo se pueden reconocer 3 subtipos de déficit en la percepción de objetos: el trastorno de la discriminación sensorial visual, la agnosia aperceptiva, y la agnosia asociativa, que son otros tipos de trastornos que impiden indentificar o reconocer el nombre de objetos. [6].

Se administra un test inicial de cribado y, a continuación, una serie de subtest que establecen pruebas para evaluar la percepción de objetos. Los subtest más relevantes de esta batería son los siguientes:

1. **Decisión de objeto**: se le presentan 20 láminas al paciente con 4 siluetas en cada una (de esas 4 siluetas hay un objeto real en 2 dimensiones y 3 distractores). El sujeto debe identificar y señalar el objeto real.

2. **Siluetas progresivas**: compuesto de 10 siluetas situadas inicialmente en un ángulo de rotación máximo y revelando mayor número de detalles progresivamente. Se le pide al participante que identifique el objeto lo antes posible.

3. **Discriminación de la posición**: se muestran en 20 partes, dos cuadrados adyacentes y un punto situado exactamente en el centro de uno de ellos y desviado del punto medio en el otro [7]. Se pide que el paciente decida qué cuadrado es el que contiene el punto centrado.

4. **Localización del número**: se presentan 10 láminas, cada una de ellas con 2 cuadrados, uno situado en la parte superior que contiene números del 1 al 9, distribuidos aleatoriamente, y otro, situado en la parte inferior que presentaba un punto. Se solicitó al participante que identificara cuál era el número que se correspondía con la posición del punto.

2.2.2. Trail Making Test

El objetivo de este test es evaluar distintas funciones cognitivas, entre ellas la atención, flexibilidad cognitiva y la velocidad psicomotora. [8]. El TMT Test consta de dos partes:

- La parte A, en la cual se pretende realiza mediante líneas una conexión de forma consecutiva. Esta conexión se realizará entre 25 números que se encontrarán distribuidos al azar en una hoja.

- En la parte B, la conexión tiene que seguir la misma lógica, pero uniendo números y letras de forma alterna.

2.2.3. FCSRT Test

El FCSRT es un test utilizado en neurología para evaluar la capacidad de aprendizaje y memoria verbal. Permite valorar, entre otras la capacidad de retención. Esta prueba en concreto ha sido ampliamente utilizada para la evaluación del déficit de memoria en diferentes enfermedades neurológicas, como la EA, y en otros trastornos, también bastante conocidos como son el estrés pos-traumático, depresión o esquizofrenia. [9].

2.2.4. ROCF Test

Este test consiste en una prueba de copia y reproducción de memoria de figuras geométricas.Se le debe pedir al paciente que realice dos tipos de tareas: [10]:

- **Fase de copia**: El sujeto debe copiar el modelo de la Figura de Rey (basado en figuras geométricas), indicando que la reproducción no necesariamente debe ser exacta, pero que debe poner atención en los detalles y las proporciones.
- **Fase de reproducción de memoria**: Tras un cierto tiempo de la primera fase se le pide al paciente que reproduzca la figura sin tenerla a la vista y sin recibir ninguna ayuda verbal que le permita identificar la forma o la situación de ninguno del total de elementos que conforman la figura.

2.3. Evaluación cognitiva con tecnologías digitales

Las enfermedades neurodegenerativas han estado presentes siempre en nuestras vidas, sin embargo, con el paso del tiempo y la evolución tecnológica cada vez se disponen de un mayor número de herramientas y aplicaciones para realizar pruebas de evaluación cognitiva.Igualmente, estas pruebas siguen siendo, en su mayoría, explicadas y supervisadas por un especialista.

Los test mencionados anteriormente son utilizados hoy en día en consultas médicas, pero estos suelen hacerse de forma dirigida y supervisada por un médico. Gracias a los avances tecnológicos y de forma alternativa a estos test cognitivos surgen varios tipos de herramientas TIC (como la mencionada anteriormente, Vienna Test System) que realizan adaptaciones de este tipo de pruebas con el objetivo de la detección precoz de enfermedades como la demencia o la EA. Entre ellas destacan las aplicaciones web y las aplicaciones para otros dispositivos digitales.

2.3.1. Herramientas digitales para evaluación cognitiva

El software desarrollado e implementado para alguna de estas aplicaciones se realiza con objetivo de investigación médica y otras se realizan para uso comercial, aunque estas últimas, están más orientadas al entrenamiento cognitivo para la mejora de la agilidad mental. Las aplicaciones encontradas hasta la fecha han sido:

- Neuronation [11]
- Cognifit [12]
- iMentia [13]
- BrainTest [14]
- MyCognition [15]
- Neotiv [16]
- MoCa [17]

2.3.2. Problemas encontrados

Alguno de los inconvenientes principales del software utilizado actualmente son los siguientes:

1. Pago: La mayoría de las aplicaciones son de pago. Aunque hay algunas que tienen un pequeño test de prueba gratuito, luego si se quieren seguir utilizando se tiene que pagar la suscripción.
2. Licencias: aplicaciones como iMentia, necesitan una licencia para poder utilizarse.

3. Supervisión: aplicaciones como iMentia que también realizan un test para valorar un posible deterioro cognitivo a través de diferentes test de cribado, necesita supervisión para poder realizarlos.

4. Disponibilidad: Algunas aplicaciones están únicamente disponibles en un sólo sistema operativo (iOS o Android). O no son compatibles con todos los dispositivos (solo apto para tablets etc...).

En las siguientes tablas hacemos un pequeño detalle de las aplicaciones encontradas y sus características.

TEST	**¿QUÉ EVALÚAN?**	**SUPERVISIÓN**
NeuroNation	Entrenamiento cerebral	No
BrainTest	Videojuegos de acertijos para neuroestimulación	No
myCognition	Velocidad de procesamiento, memoria de trabajo y episódica, atención y funciones ejecutivas	No
Neotiv	Memoria	No
MoCa	Detección de enfermedades neurodegenerativas, desorden del sueño, depresión, esquizofrenia	Sí
Cognifit	Memoria a corto plazo y de trabajo, percepción auditiva y visual, atención y tiempo de respuesta	No
iMentia	Memoria, orientación, lenguaje, atención, razonamiento, funciones ejecutivas y comprensión	Sí

Cuadro 2.2: Tabla I de aplicaciones

TEST	SONIDO/VOZ	PAGO	OBSERVACIONES
NeuroNation	No	Pago con evaluación gratuita	La prueba gratuita valora la función cognitiva a mejorar.
BrainTest	No	Gratuita	Juegos para neuroestimulación, como juegos de palabras, sudokus, etc.
myCognition	No	Pago	Programa de 15 semanas dividido en 3 niveles (Avanzado, Intermedio e Introducción) con diferentes ejercicios para cada semana.
Neotiv	No	Pago con evaluación gratuita	Total de 5 test de memoria, se hace uno cada 24h. Además, incluye un cuestionario de satisfacción al finalizar cada test.
MoCa	No	Pago	Dispone de dos formatos de evaluación, mediante APP y mediante papel.
Cognifit	Sí	Pago con prueba gratuita	Aparte de las evaluaciones tiene entrenamientos disponibles para adultos y niños.
iMentia	Sí	Pago y gratis con licencia	Cada área de trabajo tiene hasta 6 niveles de complejidad.

Cuadro 2.3: Tabla II de aplicaciones

2.3.3. Machine Learning en la evaluación cognitiva

La realización y desarrollo de este tipo de aplicaciones, así como el uso activo de las mismas nos puede ayudar a desarrollar ideas innovadoras y eficaces capaces de no discriminar a los individuos de riesgo, identificar a los pacientes en fase preclínica o temprana de la enfermedad, establecer estrategias individualizadas y preventivas para mejorar el pronóstico y la calidad de vida de los pacientes. Una de estas ideas innovadoras es un concepto bastante nuevo de aplicar técnicas de Machine Learning a las herramientas TIC de evaluación cognitiva.

En medicina, el análisis de datos está basado en técnicas estadísticas, las cuáles cada vez se van quedando más obsoletas debido al crecimiento de la información almacenada. Por ello, actualmente se ha comenzado a utilizar Machine Learning como herramienta de apoyo que permite analizar datos sintomatológicos de pacientes, para así, poder obtener un diagnóstico lo más acertado posible, y un tratamiento adecuado a seguir.
Gracias al desarrollo de las tecnologías y con ayuda de técnicas de Machine Learning, se ha podido hacer una estimación y predecir el estado cognitivo de la persona que se está evaluando, de manera bastante precisa.

Entre los diferentes algoritmos utilizados, destacan aquellos basados en árboles de decisión y clasificación. [18]
Los árboles de decisión son un modelo de predicción cuyo principal objetivo es el aprendizaje inductivo a partir de observaciones y construcciones lógicas. Este tipo de modelos se construyen

a partir de un problema, en el que se especifican las variables evaluadas, las acciones a tomar y el orden en el que se tomará una decisión. Quedando estructurado de la siguiente manera:

- Nodo principal o raíz: atributo a partir del cual se inicia el proceso de clasificación.
- Nodos internos: correspondientes a cada una de las preguntas acerca del atributo en particular del problema.
- Nodos hijos: corresponden a las diferentes respuestas de los nodos internos.
- Nodos finales u hoja: corresponden a una decisión, que coincide a su vez, con una de las variables de clase del problema a resolver.

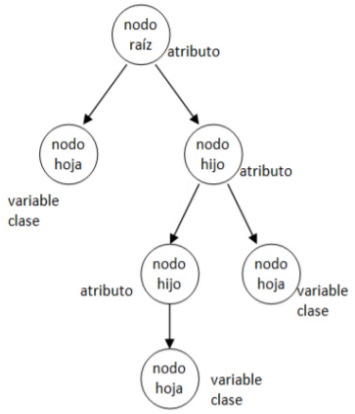


Figura 2.2: Estructura árbol de decisión.

Tras generar el modelo, sólo un camino será seguido dependiendo del valor de la variable evaluada en cada momento.[19]

2.3.4. Requisitos en herramientas de evaluación cognitiva

No solo el desarrollo e implementación de nuevas ideas aplicadas a herramientas TIC son importantes para el desarrollo de las mismas. También hay que considerar una serie de requisitos a cumplir por aquellas herramientas enfocadas a la evaluación cognitiva y que además debimos tener en cuenta para este trabajo. Son los siguientes:

1. **Accesibilidad**: los SO que predominan entre las personas con dispositivos móviles son iOS y Android, por lo que se desarrolló una APP híbrida para poder alcanzar a el mayor número de personas.
2. **Portabilidad**: hacer la aplicación para un dispositivo smartphone permite al usuario la posibilidad de hacerlo desde prácticamente cualquier lugar en el que disponga datos.
3. **Seguridad**: que los resultados de las pruebas del paciente solo estén a disposición del especialista.
4. **Facilidad de uso**: una APP sencilla e intuitiva y fácil de entender.

Capítulo 3

Metodología

El objetivo de este TFG, como ya se ha mencionado anteriormente, ha sido el desarrollo de una aplicación móvil en la que, por recomendación médica, las personas puedan realizar una serie de test cognitivos, para que, así, posteriormente, puedan ser evaluados por especialistas en el campo de la neurología.

Los principales requisitos de la APP consistían en que sea intuitiva y fácil de usar, además de estar disponible tanto para dispositivos con sistema operativo iOS como Android. Esta aplicación la hemos llamado BrainApp y en ella se pueden diferenciar tres partes:

Test de cribado

En medicina, el cribado, es la realización de una serie de pruebas que sirven de ayuda para la detección precoz de una determinada enfermedad. En nuestra aplicación incorporamos al comienzo un test de cribado. Este test consta de una serie de preguntas que sirven para poner en situación al paciente y al especialista, que luego evaluará cada una de las pruebas. Dentro de este test de cribado podemos encontrar tres tipos de preguntas:

1. Preguntas al paciente: son cuestiones que se realizan al paciente de cómo se ha encontrado en los últimos meses.
    - ¿Le cuesta recordar cosas?
    - ¿Tiene dificultad para encontrar las palabras al hablar?
    - ¿Se ha desorientado últimamente?
    - ¿Tiene dificultades para utilizar el dinero?
    - ¿Tiene dificultades para recordar eventos recientes?
    - ¿Le piden últimamente que repita las cosas?
    - ¿Tiene dificultades en decir el nombre de los objetos?
    - ¿Tiene dificultades en recordar los nombres de otras personas?
    - ¿Nota mayor ansiedad?
    - ¿Se encuentra bajo de ánimo o decaído?

Cada una de estas preguntas admite una única respuesta, afirmación o negación.

Figura 3.1: Pregunta al paciente 1.

Figura 3.2: Pregunta al paciente 9.

2. Preguntas sobre datos demográficos: sirven para saber la edad del paciente, así como su género, además de poder evaluar su nivel cultural.

   - ¿En qué año ha nacido?
   - ¿Hasta qué edad estudió?
   - ¿Cuál es su género?

   Este tipo de preguntas permiten al paciente introducir manualmente cada una de las respuestas con ayuda de un teclado exclusivamente numérico. En el caso de la última pregunta, simplemente, se le muestran dos botones al paciente para que seleccione la respuesta que considere oportuna.

Figura 3.3: Pregunta de datos demográficos.

Figura 3.4: Pregunta de datos demográficos.

3. Preguntas de orientación: son preguntas que sirven para saber si el paciente está bien orientado en el día actual en el que está realizando la prueba, así como de la edad que tiene.

- ¿Cuántos años tiene?
- Escriba la fecha de hoy: día actual, día de la semana, mes y año.

Para cada una de estas respuestas al igual que las anteriores, en el caso de ser datos numéricos, al paciente solo se le da la posibilidad de un teclado numérico. En caso contrario, si la respuesta no requiere de números se le muestra el teclado normal del dispositivo.

Figura 3.5: Pregunta de orientación 1.

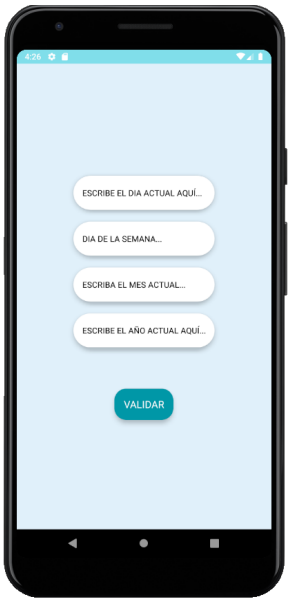


Figura 3.6: Pregunta de orientación 2.

Todas estas preguntas no tienen ningún tipo de validación instantánea, es decir, si el paciente se equivoca al introducir el día actual o los años no corresponden con el año de nacimiento introducido, la aplicación no muestra ningún tipo de error. El único requerimiento que hay en este test de cribado es que todas las preguntas tengan respuesta. Esto se debe a que todas las respuestas serán guardadas en la BBDD y posteriormente cada una de las respuestas será evaluada por el especialista.

Test de destreza

Esta parte consta de dos pequeñas pruebas que sirven para saber cómo se desenvuelve el paciente con el dispositivo desde el que está realizando el test. En la primera prueba se muestra una pequeña frase con una palabra resaltada; aquí el paciente deberá escribir dicha palabra. Por otro lado, en la segunda prueba se muestran una serie de figuras geométricas donde se indica al participante que debe pulsar únicamente sobre los círculos.

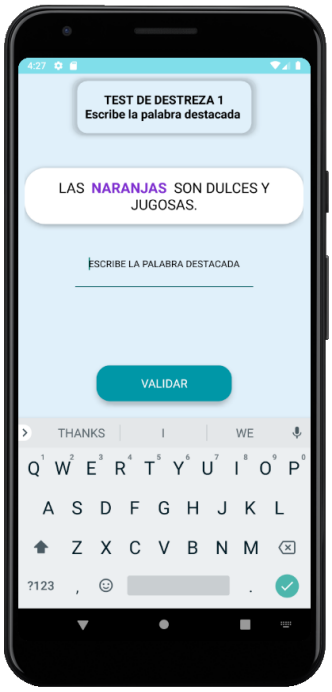


Figura 3.7: Test de habilidad 1.

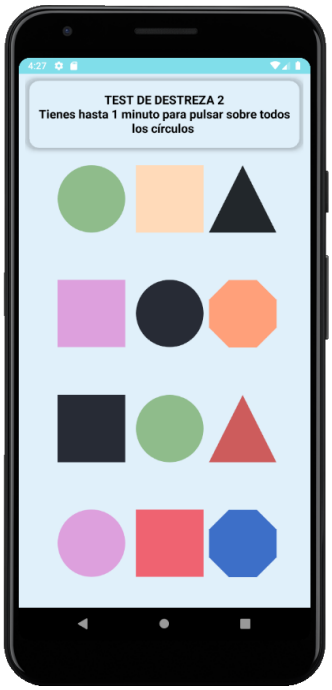


Figura 3.8: Test de habilidad 2.

Las respuestas de las dos pruebas de destreza no son guardadas en la BBDD ya que sólo ayudan a que el paciente se desenvuelva a la hora de resolver los test que deberá realizar posteriormente.

Test de evaluación cognitiva adaptados

Como se ha mencionado anteriormente, este proyecto está desarrollado en colaboración con el grupo de neurología del Hospital Clínico San Carlos. Los neurólogos nos mostraron las pruebas con las que trabajaban actualmente, en las cuales se basaron proponiendo cuatro pruebas adaptadas que pudieran ser realizadas con facilidad desde un dispositivo móvil.

Las cuatro pruebas están contextualizadas en el mismo asunto. El tema elegido corresponde con un supermercado ya que lo consideramos un tema con el que todo el mundo puede sentirse familiarizado y, para así, darle cierta armonía al conjunto global de nuestra aplicación. Al comienzo de cada una de estas pruebas mostramos al paciente las instrucciones de esta, así como una pequeña demostración de cómo debe realizarse el test.
A continuación, se procede a explicar cada una de las pruebas que contiene nuestra aplicación haciendo referencia a la prueba original en la que está basada.

1. **Test de memoria**

   Esta primera prueba de memoria está basada en el FCSRT test y consta de dos partes; una primera parte en la que se evalúa la memoria a corto plazo y libre del paciente; y, una segunda en la que se evalúa la memoria a largo plazo, también conocida como memoria diferida.

   Nuestra adaptación consiste en mostrarle al paciente nueve palabras que tendrá que recordar,

cada una de ellas perteneciente a una categoría semántica. Estas son la lista de palabras que se muestran, agrupadas por sus respectivas categorías:

- Alimentación
  - Fideos
  - Levadura
  - Azúcar
- Aseo personal
  - Gasas
  - Acondicionador
  - Servilletas
- Limpieza del hogar
  - Detergente
  - Esponja
  - Lavavajillas

Cada una de las palabras las mostramos en un color diferente, dependiendo de la categoría a la que pertenecen. En el caso de las palabras pertenecientes a alimentación se muestran en rojo; las que corresponden a aseo personal en verde y las de limpieza del hogar en azul. La clasificación por colores podría proporcionar al paciente alguna ayuda a la hora de recordar las palabras dado que podría asociar cada color a una categoría semántica, resultándole, así, más fácil el desarrollo de la prueba.

Al comienzo de esta prueba se le muestran al paciente las nueve palabras seguidas durante un tiempo de cinco segundos por palabra. Esta lista de palabras sólo se mostrará una única vez. Tras mostrar al paciente estas nueve palabras, se le pedirá que escriba todas las palabras que recuerde, dándole nueve espacios en blanco para que pueda escribir las que recuerde.

Seguidamente se le pedirá al paciente que escriba las palabras que recuerde por categoría, siendo la primera categoría alimentación, seguida de limpieza del hogar y por último aseo personal. En cada una de estas peticiones se le facilitarán tres espacios en blanco por categoría.

La segunda parte -para poder evaluar la memoria diferida del paciente- es mostrada cierto tiempo después, por lo que, tras terminar la primera parte, el paciente pasa a realizar otro tipo de test para distraerle. Una vez terminado, realizará la segunda parte del test de memoria, es en esta segunda parte donde se le vuelve a pedir al paciente que escriba todas las palabras que recuerde, esta vez sin mostrarle la lista de palabras.

2. **Test de localización**
   Este test está basado en uno de los subtest del VOSP, descrito en apartados anteriores, en concreto en el Test de Localización del número.

En este caso, la adaptación que hemos implementado es muy similar al test original, la única diferencia es el uso de frutas, en vez de números. Se muestran al paciente diez pantallas diferentes, cada una de ellas con dos cuadrados, uno situado en la parte superior que contiene diferentes piezas de fruta, distribuidos aleatoriamente, y, otro situado en la parte inferior que presenta un punto.

En cada una de las pantallas se pedirá al participante que identifique cuál es la pieza de fruta que se corresponde con la posición del punto. Dicho punto va cambiando de posición en cada una de las diferentes pantallas.

3. **Test de fluencia de diseño**
   Esta tercera prueba resulta del test original Trail Making Test que ha sido descrito anteriormente.
   En la adaptación desarrollada se le muestran al paciente cinco fresas distribuidas como los puntos del número cinco de un dado. En esta prueba se le pedirá al participante que haga quince diseños diferentes y únicos o todos aquellos que pueda realizar en un minuto. Cada diseño se hará tras la unión de cada una de las fresas dispuestas en la pantalla.

4. **Test de denominación del lenguaje**
   La última prueba de nuestra aplicación se basa en dos de los subtest del VOSP, en concreto en el Test de Decisión de Objeto y en el Test de Siluetas Progresivas.

   La adaptación de estos dos test, propuesta por el departamento de neurología, consiste en que se le muestre al paciente veinte pantallas, cada una de ellas con un objeto perteneciente al supermercado y un espacio para que pueda introducir el nombre de cada uno de los objetos. En cada una de las pantallas el participante deberá reconocer, si puede, el objeto que se le muestra y escribirlo en un periodo máximo de veinte segundos.
   Los objetos son los siguientes:

   - Huevo
   - Chocolate
   - Fresa
   - Zumo
   - Cereales
   - Manzana
   - Queso
   - Pera
   - Zanahoria
   - Guisantes
   - Jamón
   - Sandía
   - Tomate
   - Uvas
   - Mantequilla
   - Aceite
   - Miel
   - Kiwi
   - Brócoli
   - Plátano

   Para el desarrollo de cada uno de estos test se tuvo en cuenta que entre las diferentes pruebas no se repitiesen objetos, esto es, no se reproducen las mismas palabras entre los test de

denominación y memoria. Así como, el orden de realización de cada uno de ellos. Ya que esto podría suponer alguna dificultad para el paciente a la hora de realizar cada una de las pruebas.

3.1. Tecnologías Aplicadas

A continuación, se explicarán las principales tecnologías y herramientas en las que nos hemos apoyado para llevar a cabo tanto el desarrollo frontend y backend de la aplicación, así como los servicios cloud, que se ha utilizado para desplegar nuestra API y nuestra BBDD. En primer lugar, nos centraremos en la tecnología utilizada para el frontend de la aplicación, el cual fue llevado a cabo con React Native.

3.1.1. React Native

La principal razón por la que escogimos este framework para las vistas de nuestra aplicación es que queríamos desarrollar la aplicación en los dos principales sistemas operativos para dispositivos móviles, iOS y Android. Estas dos aplicaciones pueden tener el mismo diseño y la misma lógica, pero varios componentes de la UI iban a ser diferentes y las propias aplicaciones debían desarrollarse en dos lenguajes diferentes. Es decir, nuestro objetivo requería que se creasen dos versiones diferentes una en lenguaje java para Android y otra en lenguaje nativo para iOS [20].

Por esto mismo, React Native se consideró como la mejor candidata ya que se considera un framework de programación de aplicaciones nativas multiplataforma. Además, se basa en JavaScript (un lenguaje de programación interpretado) y ReactJS (una librería de JavaScript) y su principal característica se orienta en ayudar a crear aplicaciones que puedan ser ejecutadas tanto en iOS como Android simultáneamente con el mismo código base. Igualmente, en lugar de que las aplicaciones sean ejecutadas en navegador, estas corren directamente sobre las plataformas móviles nativas, en este caso iOS y Android. Este desarrollo híbrido, aunque no nos eximió de realizar pequeños cambios entre una versión y la otra, sí nos permitió reutilizar la estructura y gran parte del código implementado para todas las pantallas.

3.1.2. Android Studio y XCode

Tanto Android Studio como XCode pueden clasificarse como herramientas IDE, es decir, se usan para el desarrollo y creación de aplicaciones móviles y, además, proporcionan un conjunto de herramientas para construir y probar aplicaciones. XCode se utiliza para crear aplicaciones para iOS, MacOS, etc., y Android Studio para dispositivos móviles con sistema operativo Android.

Así pues, como nuestro objetivo consiste en desarrollar para ambos sistemas operativos, utilizamos ambos IDE para la simulación de la aplicación durante toda la implementación del proyecto. Esto nos ayudó a ir probando las vistas de la aplicación, emulando nuestro programa como si fueran en dispositivos móviles. Aunque Android Studio se puede instalar prácticamente en computadores con cualquiera de los SO más utilizados, como Windows, MacOS o Linux, el IDE XCode se puede

instalar solo en MacOS, por lo que tuvimos que utilizar dos SO para la simulación y desarrollo de la aplicación desde el inicio. Estos fueron Windows (el SO que tenía la mayoría de nuestros computadores) y en MacOS. [21].

Para nuestro desarrollo backend, servidor, API y BBDD utilizamos las siguientes herramientas que mencionaremos en cada uno de los siguientes puntos.

3.1.3. API REST en Laravel

También necesitaremos del uso de una API, que es una interfaz de programación de aplicaciones, es decir, un conjunto de servicios ofrecidos desde un servidor, que una aplicación puede usar en remoto para enviar o recibir información, y así comunicarse con el servidor central. Las ventajas de contar con una API son los siguientes:

- Permiten la reutilización de código, reduciendo los tiempos en el desarrollo de aplicaciones.
- Incrementan la interoperabilidad entre aplicaciones, ya que las máquinas se comunican información entre sí más rápido que los humanos.

En nuestro caso elegimos crear una API REST [22].

La API REST se basa en un protocolo sin estado de cliente-servidor que nos ayudó a conseguir una aplicación que nos permitiese trabajar con la base de datos y después consumirla mediante cualquier otro lenguaje o framework frontend como React Native. Decidimos crear nuestra API REST con Laravel que es un framework de PHP [23].

Laravel nos ayuda entre otras cosas a desarrollar una aplicación, por medio de su sistema de paquetes y se caracteriza por ser un framework de tipo MVC, y por ser capaz de instanciar clases y métodos para usarlos en muchas partes de nuestra aplicación sin la necesidad de escribirlo y repetirlos muchas veces con lo que eso conlleva a la hora de modificar algo en el código.

Otra ventaja de Laravel, que hizo que destacara y, por tanto, la eligiésemos para crear nuestra API, es su facilidad de uso y aprendizaje, es decir, la curva de aprendizaje es suave además de ser bastante utilizado en el mercado. Además, su documentación se considera otra gran ventaja, ya que Laravel cuenta con muchos ejemplos de uso junto con los paquetes que se debe usar en los controladores para utilizar esa clase, sabiendo que las llamadas serán devueltas, si un string, un array, un booleano, etc.

3.1.4. SQL

La sintaxis escogida para crear nuestra BBDD fue SQL, un sistema de gestión de bases de datos relacional. Aunque se barajó la posibilidad de realizarlo frente a una BBDD NoSQL, sin embargo, finalmente, esta opción fue descartada.

La principal diferencia que existe entre SQL y NoSQL es la resolución de escenarios completamente diferentes y excluyentes el uno del otro, es decir, para lo que resulta ideal SQL, no lo

es NoSQL y viceversa. Aunque las principales diferencias que se tuvieron en cuenta, entre ellas, fueron:

- SQL permite realizar combinaciones de forma fácil y eficiente entre diferentes tablas para extraer información relacionada, mientras que NoSQL no se permite o se permite de forma muy limitada.
- NoSQL nos permite trabajar con grandes cantidades de información; mientras que SQL facilita la gestión en bases de datos relacionales.

Como NoSQL está pensado principalmente para manejar grandes cantidades de información (Big Data); y como nuestro objetivo era que nuestra BBDD mantuviera una relación jerárquica de los datos, nos decantamos finalmente por SQL. El servidor en el que se llevó a cabo el desarrollo local de la BBDD fue en MySQL Server.

3.1.5. Postman

Esta herramienta es conocida porque permite crear peticiones sobre APIs de una forma sencilla y de esta manera, poder probarlas. Un desarrollador puede utilizar Postman para comprobar el funcionamiento de una API que esté en desarrollo. Por ello escogimos utilizarla, para realizar testing exploratorios para nuestra API REST y comprobar su correcto funcionamiento [24].

3.1.6. Amazon Web Services

Para desplegar nuestra API y la BBDD en la nube se utilizaron los recursos de Amazon Web Services. AWS es una plataforma que ofrece servicios, como por ejemplo, infraestructuras de las TIC para empresas, que nos permite disponer de almacenamiento, recursos de computación, aplicaciones móviles, bases de datos entre otros en forma de lo que hoy conocemos como Cloud Computing [25].

Esta nube (Cloud) nos da la posibilidad de utilizar servicios en la red, como por ejemplo guardar información sin disponer de la estructura necesaria que hace falta. Hoy en día el almacenamiento de información en la nube es más demandado por los usuarios de internet.

3.1.7. Visual Studio Code

Actualmente los sistemas operativos viene por defecto con una herramienta que nos permite editar texto, archivos o tomar notas fácilmente. Aunque este programa cumple su función, la verdad es que está infinitamente limitado en todos los sentidos. Por ello, se necesitan características más avanzadas como por ejemplo para programar cualquier tipo de software, a menudo se recurren a otras alternativas mucho más completas y profesionales, como es el caso de Visual Studio Code.

El entorno Visual Studio Code fue el escogido para desarrollar todo el software de la aplicación. [26].

3.1.8. Repositorios GitHub

Para tener un control de versiones del código y poder programar de forma simultánea por parte de todos los integrantes del grupo, hicimos uso de la herramienta GitHub. Esta herramienta se caracteriza por ser una de las principales plataformas para crear proyectos, y destaca sobre todo por sus funciones colaborativas que ayudan a que el código de los repositorios que sean públicos pueda ser descargado y revisado por cualquier usuario, lo que ayuda a mejorar el software y crear ramificaciones a partir de él. Y en caso de preferir que tu código no se vea, también pueden crearse proyectos privados.

En nuestro caso se crearon dos repositorios esn GitHub, uno para el desarrollo de la **aplicación en React Native** [27] y otro para montar la **API** [28]

3.2. Arquitectura

En esta sección, explicaremos en detalle, como se relacionan las tecnologías empleadas durante el desarrollo de este proyecto. A continuación podemos ver un gráfico en el que se muestra cómo se relacionan dichas tecnologías.

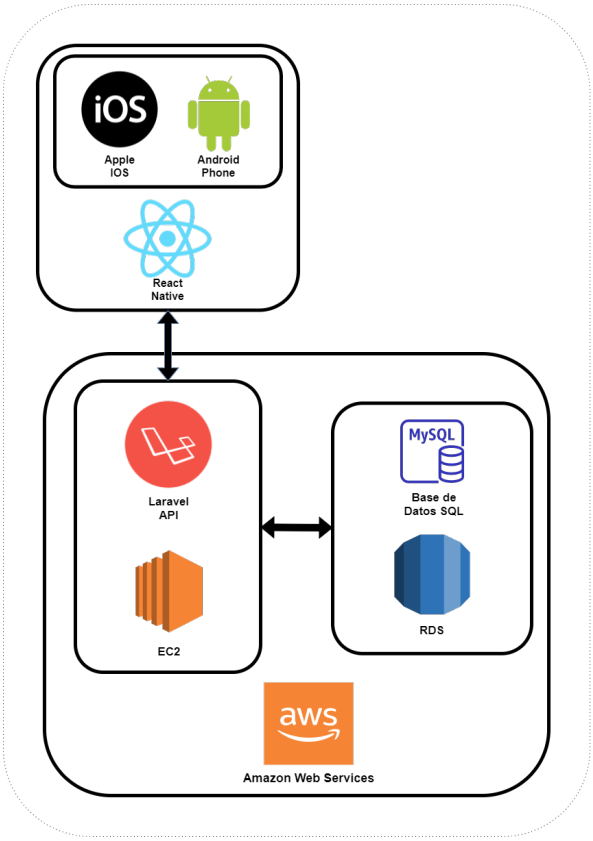


Figura 3.9: Relación entre las tecnologías utilizadas.

3.2.1. Frontend

Para iniciar el diseño frontend de nuestra aplicación primero nos apoyamos en la herramienta de diseño gráfico Adobe XD. Esta herramienta nos ayudó sobre todo para visualizar los primeros bocetos de las principales vistas a desarrollar para la APP creando Mockups para ellas. Estos Mockups son maquetas que se utilizan sobre todo para ayudar a visualizar las primeras ideas en cuanto al diseño móvil que teníamos en mente. Fuimos creando los bocetos de las vistas que teníamos claro que iban a ser las primeras en desarrollarse como es el caso de las pantallas de Login y Register, y de forma posterior el resto de las pantallas.

Los siguientes Mockups muestran las maquetaciones iniciales de las pantallas.

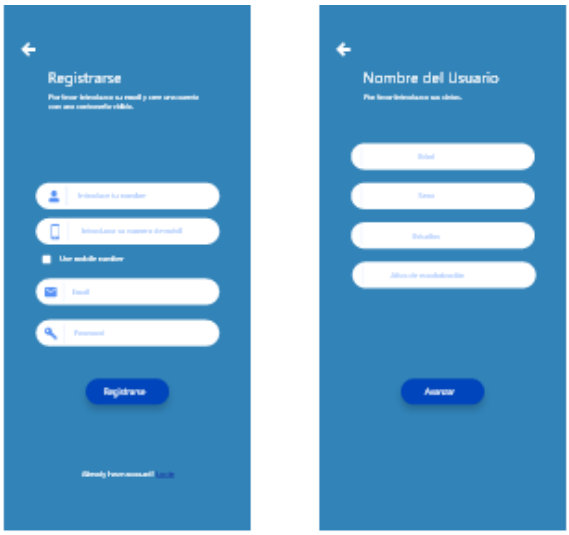


Figura 3.10: Mockups de las pantallas iniciales.

Figura 3.11: Mockups de las pantallas de transición y test de usabilidad.

Una vez entendido el concepto e idea inicial que teníamos a priori para empezar a desarrollar nuestras vistas, escogimos el entorno de Visual Studio Code para programar y desarrollar el software de toda la parte frontend. El framework utilizado fue el explicado en el punto anterior, React Native, que se basa en el lenguaje de programación Java Script. Para poder estilizar la visualización de nuestros componentes también hicimos uso del lenguaje de diseño gráfico css.

Las primeras vistas a desarrollar fueron las que estaban relacionadas con todas las pantallas donde realizamos el test general, este test se realiza para saber la situación y orientación actual del usuario que está haciendo uso de la aplicación, y en su mayoría todas las vistas se basan en preguntas al usuario. El esquema principal de todas las vistas desde momento en el que el usuario abre la aplicación hasta que el usuario termina el test general previo a los test de evaluación cognitiva se encuentran ordenados en el siguiente esquema:

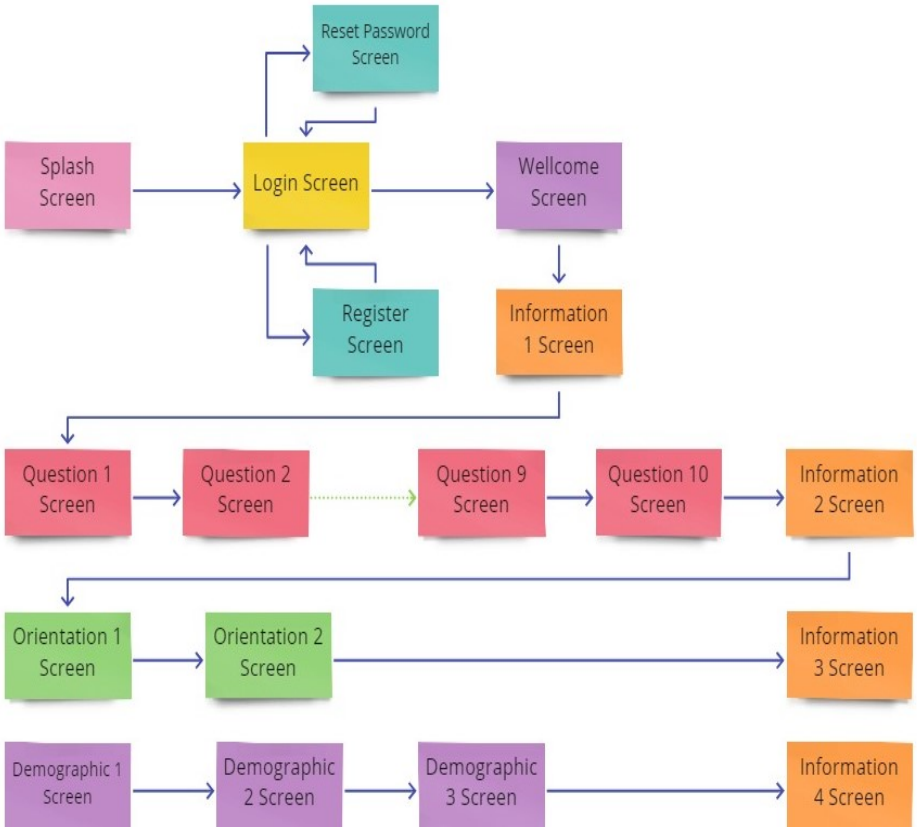


Figura 3.12: Estructura de la vistas de la aplicación.

El uso de la aplicación se hace de forma secuencial , pues no permitimos que una vez iniciada la prueba se pueda volver hacia atrás.

Esto se diseño específicamente así para que no se pudiesen retroceder y bien cambiar la respuesta o volver a repetir algún otro tipo de prueba de los test cognitivos. Las siguientes imágenes muestran el diseño que finalmente dimos a nuestras tres pantallas iniciales:

1. Splash Screen

2. Login Screen

3. Register Screen

(Según hemos listado de arriba, abajo, se muestran de izquierda a derecha).

Figura 3.13: Spash Screen

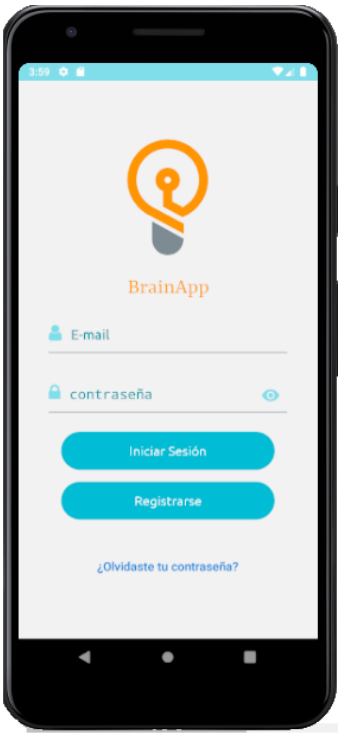


Figura 3.14: Login Screen

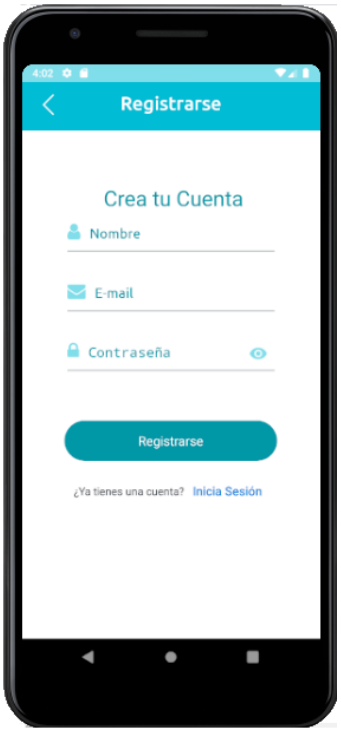


Figura 3.15: Register Screen

De estas tres pantallas, destacaremos la actividad de la principal, que es la vista Login Screen. Esta pantalla se encarga sobre todo de la acción de Inicio de Sesión del usuario, y es por ello que partir de esta pantalla comienza de forma secuencial las preguntas generales y los test de evaluación cognitiva así como la interacción del usuario con la aplicación. La interacción cliente-sistema para el inicio de sesión se describe en la imagen 3.16 en cinco pasos.

1. Acceder a la aplicación: primero mostramos la vista Splash Screen una vez el usuario accede a la aplicación, esta vista muestra nuestro logo durante un par de segundos y redirige directamente a la pantalla de Login.

2. Pedir datos: la vista Login Screen pide directamente los datos , como hemos visto en la imagen ,

3. Ingresar datos: el usuario tiene que introducir un correo y contraseña válidos.

4. Verificar usuario: los datos introducidos por el usuario se envían a través de una llamada a la API y se cotejan con nuestros datos de la BBDD que tenemos desplegada en AWS.

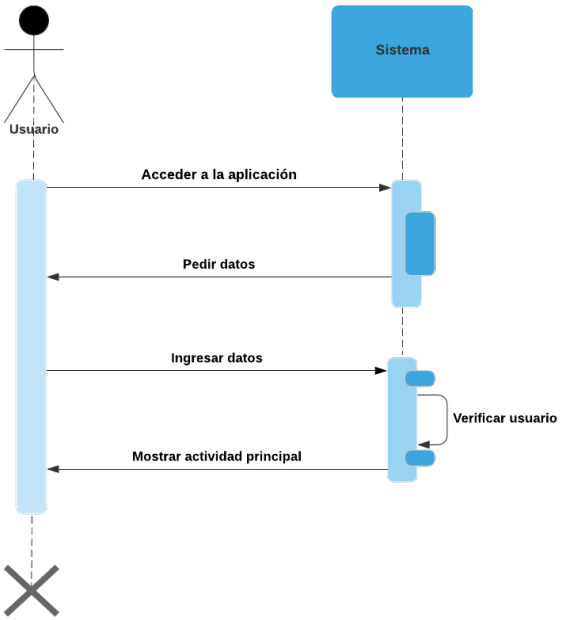


Figura 3.16: Diagrama de secuencia de Inicio de Sesión.

5. Mostrar actividad principal: una vez se ha comprobado que el usuario y contraseña se corresponden con un usuario registrado previamente se redirige al usuario a la página Wellcome Screen, a partir de está pantalla daremos dos opciones al usuario , o bien salir e iniciar la prueba o bien iniciar la actividad y comenzar con las preguntas generales y test.

3.2.2. Backend

Para realizar una comunicación entre nuestro Frontend y nuestra base de datos hemos tenido que desarrollar una API en el entorno de programación Laravel. Esta decisión fue tomada porque Laravel es un framework del lenguaje de programación php y nuestro nivel de familiaridad con este lenguaje es alto, también hemos tenido en cuenta su fácil acceso a la base de datos, pero esto será explicado en unos apartados más adelante.

Laravel es un framework basado en el patrón arquitectónico MVC (Modelo Vista Controlador) que incluye en sus modelos un sistema de datos relacional llamado ELOQUENT ORM que nos facilita la creación de modelos y conectarlos con nuestra base de datos. También incluye Controladores, es la parte del backend que dota nuestra aplicación de funcionalidad y nos permite organizar el código en clases [23].
Laravel también incluye un sistema de vistas llamado Blade, que en este proyecto no se ha utilizado porque para desarrollar el API no era necesario.

1. **Eloquent**
   Laravel trabaja con Eloquent ORM, que son los encargados de almacenar la información y los accesos a esta.
   Eloquent ORM, es un sistema que nos facilita la conexión con la base de datos, nos permite

manejar los datos por medio de objetos (modelos), de esta manera, realizamos consultas a la base de datos desde nuestros modelos.
Para nuestra aplicación, hemos creado doce modelos que están relacionadas con doce tablas en la BBDD.

Para poder entender mejor Laravel Eloquent, a continuación mostraremos el modelo `Memory.php`

```
<?php
    namespace App\Models;

    use Illuminate\Foundation\Auth\User as Authenticatable;
    use Illuminate\Notifications\Notifiable;
    use Laravel\Passport\HasApiTokens;

    class Memory extends Authenticatable
    {
        use Notifiable, HasApiTokens;

        protected $table = 'memory_test';

        /**
         * The attributes that are mass assignable.
         * @var array
         */
        protected $fillable = [
            'user_id',
            'word_1',
            'word_2',
            'word_3',
            'word_4',
            'word_5',
            'word_6',
            'word_7',
            'word_8',
            'word_9',
            'fails',
            'points'
        ];
    }
```

Como podemos observar en el código mostrado, el modelo Memory almacena en el campo fillable los datos que se van a almacenar en nuestra tabla memory_test. Este modelo se encaga de conectarse con nuestra tabla memory_test y almacenar las palabras, el número de fallos y el número de aciertos que el usuario ha escrito en nuestra APP.

2. **Rutas y Middleware**
   Laravel permite utilizar un controlador de rutas muy sencillo.
   Al crear un nuevo proyecto, el propio entorno de programación nos crea una sección de rutas para controlar la funcionalidad que va a tener nuestra API.
   A continuación se mostrara un ejemplo de rutas de nuestra API en Laravel `api.php`

```php
<?php
use Illuminate\Http\Request;
use App\Http\Controllers\Api\AuthController;
use App\Models\Demographics;
use App\Models\InitialEvaluatiton;

/*
|--------------------------------------------------------
| API Routes
|--------------------------------------------------------
|
| Here is where you can register API routes for your application. These
| routes are loaded by the RouteServiceProvider within a group which
| is assigned the "api" middleware group. Enjoy building your API!
|
*/

//AUTENTICATION (logIn and logOut)
Route::group([
    'prefix' => 'auth'
], function () {
    Route::post('login', [AuthController::class, 'login']);
    Route::post('signup', [AuthController::class,'signUp']);

    Route::group([
      'middleware' => 'auth:api'
    ], function() {
        Route::get('logout', [AuthController::class, 'logout']);
        Route::get('user', [AuthController::class,'user']);
    });
});

//Initial test
Route::post('addQuestions', [InitialEvaluationController::class, 'addQuestions'])->middleware('auth:api');


//Demographics Test
Route::post('addAnswers', [DemographicsController::class, 'addAnswers'])->middleware('auth:api');
```

Como podemos observar, para definir una ruta es importante indicar el método HTTP que esta va a tener, por ejemplo get, post, update o delete. También será necesario indicar el controlador que se encargará de realizar la funcionalidad a la que queremos direccionar mediante la ruta creada.

La sección rutas de Laravel también nos ofrece una funcionalidad muy útil para la gestión de

acceso (guardar información o modificar datos), esta gestión de acceso son los middleware. Los middleware, son archivos o controladores que se ejecutan cuando se realiza una petición a nuestra API, este mecanismo sirve para filtrar las peticiones HTTP que entran en nuestra aplicación. Por ejemplo en nuestro proyecto hemos utilizado el middleware que Laravel nos ofrece por defecto (auth:api), este middleware verifica que el usuario que está realizando la petición este autentificado en nuestro proyecto. Si el usuario no está autentificado, nuestra API devolverá un mensaje de error, pidiendo al usuario que se identifique (iniciar sesión). Sin embargo, en caso de que el usuario se haya autentificado previamente, el middleware permitirá que la petición continúe.

3. **Controladores**
   Los Controladores son las clases que se encargan de añadir la parte funcional a nuestra aplicación. En nuestro proyecto tenemos doce controladores :

   - AuthController: Gestiona el registro y el inicio de sesión del API.
     Cuando un usuario inicia sesión correctamente, este controlador se encargará de crear un token para poder mantener la sesión activa, de esta forma se le permite al usuario continuar con la aplicación.
     Un token es una clave de acceso personal que se utiliza para autenticar las solicitudes HTTP que se realizan a nuestra aplicación.
   - InitialEvaluationController: Este controlador recibe las respuestas que el usuario ha realizado en la primera parte de la aplicación.
   - OrientationController, DemographicController: Estos controladores se encargan de comprobar y validar que las respuestas obtenidas después de que el usuario complete el test de Orientación y el test de datos demográficos. Estos controladores utilizan los modelos Orientation y Demographic para almacenar los datos obtenidos en las tablas orentation_test y demographic_test.
   - AlimentationController, AseoController, CleanningController y Memorycontroller: Estos controladores gestionan y validan las respuestas que el usuario ha ingresado al realizar el test de memoria, este test consiste en mostrar 9 palabras distintas pertenecientes a tres departamentos de un supermercado. Después de que se muestren las palabras, el usuario deberá introducir las palabras que este recuerde.
     Estos controladores se encargan de analizar las palabras que el usuario ha introducido, para realizar las comprobaciones entre las palabras correctas y las palabras que introduce el usuario, nos decantamos por utilizar la función similar_text() de php.
     La función similar_text calcula la similitud entre dos strings (palabras) y nos devuelve la similitud en porcentaje, para el Test de Memoria decidimos que el mínimo porcentaje de similitud entre las palabras debería ser el 75 %.
     De esta forma almacenamos todas las respuestas que el usuario ha introducido, el número de palabras correctas, que es este caso serían las palabras que tengan un porcentaje de similitud superior al 75 %, y el número de palabras incorrectas, las cuales serían aquellas que no lleguen al 75 %.

- FluentController: Este controlador almacena las respuestas que el usuario ha introducido en el test de Afluencia, los datos que este controlador se encargará de almacenar utilizando el modelo Fluent, son el número de diseños únicos, el número de repeticiones, y el número total de diseños.
- LocationController: Este controlador procesa los datos recibidos del usuario al realizar el test de localización.
- DenominationController: Este controlador valida las respuestas que el usuario introduce en el test de denominación, este test consiste en mostrar una imagen al usuario, y este deberá escribir el nombre del objeto que se le está mostrando.
  Para realizar las comprobaciones necesarias entre el nombre del objeto y la palabra que introduce el usuario, decimos que para este caso lo mejor sería utilizar la función de levenshtein(). La función de Levenshtein calcula la distancia entre dos strings. Por ejemplo si tenemos una palabra coche y otro carro, la distancia de Levenshtein sería 4 porque se necesitan cambiar 4 caracteres para que la palabra coche sea igual a carro. Para este test, se tomarán como correctas las palabras que tengan una distancia menor al 30 %.
- PasswordController: Este controlador se encarga de realizar el cambio de contraseña en caso de que el usuario lo solicite, para ello hemos creado tres funciones principales.
  - create: esta función se encarga de verificar que el usuario que ha solicitado el cambio de contraseña exista en la BBDD, en caso afirmativo, esta función se encarga de crear un nuevo token que será reenviado al usuario vía email.
  - find: Esta función se encarga de verificar que el token que se le ha enviado al usuario sigue siendo válida ya que tienen un periodo de validez de dos horas.
  - reset: Después de verificar que el token previamente enviado al usuario, es válido, esta función recibe como argumento la nueva contraseña del usuario y se encarga de encriptar la contraseña utilizando la funcion bcrypt() de php para luego guardarla en nuestra base de datos.

A continuación mostraremos un ejemplo de Controlador en Laravel `OrientationController.php`

```php
<?php

namespace App\Http\Controllers\Api;

use App\Http\Controllers\Controller;
use Illuminate\Support\Facades\Auth;
use App\Models\User;
use App\Models\Orientation;
use Carbon\Carbon;
use Illuminate\Http\Request;

class OrientationController extends Controller
{
    /**
```

```
15      *   Add users answers to table orientation_test
16      */
17      public function addAnswersOrientation(Request $request)
18      {
19          $request->validate([
20              'year' => 'required',
21              'month' => 'required|string',
22              'weekDay' => 'required|string',
23              'monthDay' => 'required',
24              'user_years' => 'required'
25          ]);
26
27          Orientation::create([
28              'user_id' => Auth::user()->id, //get user id that is
                    autenticated
29              'year' => $request->year,
30              'month' => $request->month,
31              'weekDay' => $request->weekDay,
32              'monthDay' => $request->monthDay,
33              'user_years' => $request->user_years
34          ]);
35
36          return response()->json([
37              'message' => 'Successfully evaluation filled!'
38          ], 201);
39      }
40
41  }
```

Como podemos observar en el código de ejemplo, el controlador OrientationController, solo tiene un método, addAnswersOrientation, este método se encarga de comprobar que todos los datos recibidos por la petición existan y sean válidos. En caso de que recibamos una petición con datos incorrectos, Laravel automáticamente devolverá un mensaje de error informando los datos que faltan para realizar la validación correctamente, por otro lado si los datos se han validado correctamente, esta función se encargará de llamar al Modelo `Orientation`, creado previamente, para almacenar los datos en la BBDD.

3.2.3. Base de Datos

Para el desarrollo del API, decidimos que la mejor opción para poder almacenar los datos sería utilizar una Base de datos open source, por ese motivo nos decantamos por utilizar Mysql.
La arquitectura de la APP está basada en el modelo cliente-servidor. Esto quiere decir que la base de datos se encuentra en un servidor externo, en nuestro caso, la BBDD se encuentra en AWS. A ella se accede a través del API del servidor mediante llamadas HTTPS. Esta base de datos almacenará toda la información de la aplicación, desde la información de los usuarios registrados en ella, así como las respuestas introducidas por los mismos.

Actualmente la base de datos tiene la siguiente estructura 3.17:

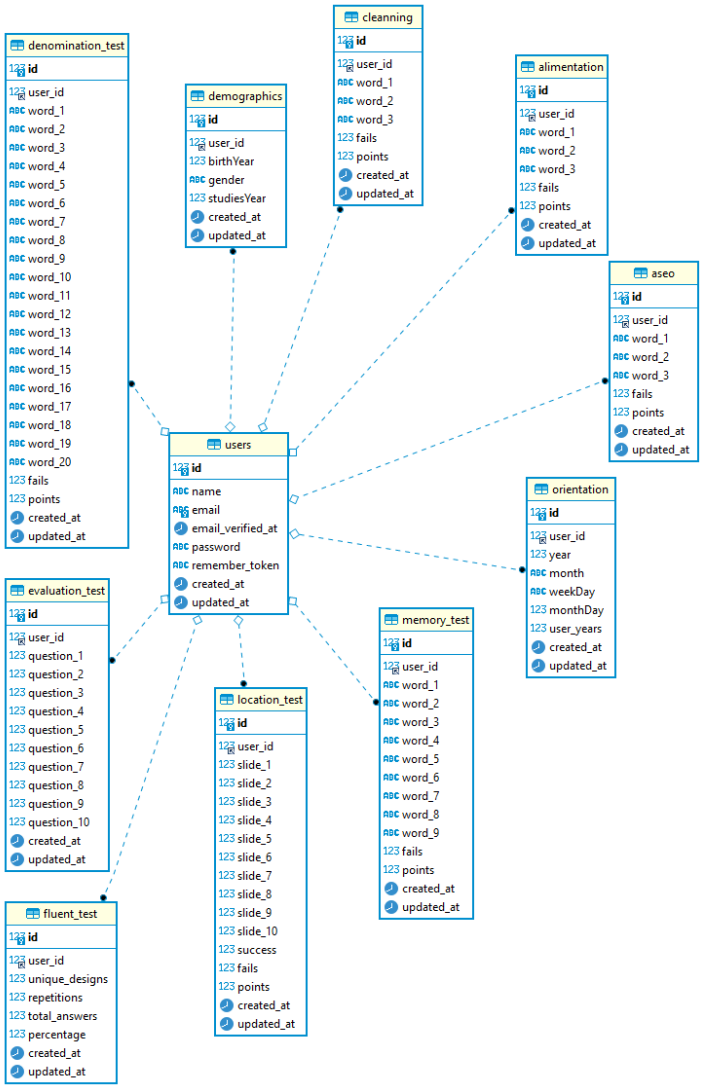


Figura 3.17: Estructura BBDD.

La BBDD consta de 12 tablas en las que cada una almacena las respuestas de los usuarios y los datos personales de estos.

- Tabla users: almacena los datos de los usuarios registrados en la APP.
- Tabla password_resets: esta tabla almacena el token junto con la fecha y la hora en la que se genera. Esta tabla se modifica cada vez que un usuario solicita un cambio de contraseña.
- Tabla evaluation_test: en esta tabla se almacenan las respuestas de las 10 primeras preguntas del test de cribado.
- Tabla demographics_test: guarda las respuestas del participante correspondientes segundo grupo de preguntas del test de cribado
- Tabla location_test: guarda por cada una de las pantallas del test de localización un booleano (true/false) en caso de haber acertado o no, así como el número total de aciertos, fallos y puntos al finalizar el test.

- Tabla fluent_test: almacena el número de diseños únicos, número de repeticiones, número total de diseños y el porcentaje de aciertos del test de fluencia de diseño.
- Tabla denomination_test: almacena cada una de las respuestas introducidas por el paciente a las 20 imágenes mostradas en el test de denominación. En esta tabla también se almacenan los puntos obtenidos y los fallos calculados mediante el Algoritmo de Levenshtein.
  Por último, para almacenar las respuestas del usuario al test de memoria decidimos implementar 4 tablas adicionales.
- Tabla memory_test: almacena las respuestas correspondientes a la lista completa del test.
  - Tabla aseo: guarda las respuestas de la categoría de aseo.
  - Tabla cleanning: almacena las respuestas de la categoría de limpieza del hogar.
  - Tabla alimentation: reúne las respuestas de la categoría de alimentación.

La tabla principal es la tabla de usuarios, en ella se almacenarán los datos de cada uno de los usuarios que se registren en la APP, otorgándoles a su vez un identificador único.
Este identificador único es la FK anteriormente mencionada del resto de las tablas de la BBDD. Cuando un usuario se elimine de la BBDD, se borrarán por consiguiente todos los datos del resto de tablas relacionadas con el identificador del usuario eliminado.
Además del identificador, y las respuestas del paciente, también se registrarán en cada una de las tablas la fecha y hora en la que se han registrado los datos. Esto nos ayudará a saber cuánto ha tardado el paciente en realizar cada uno de los test y también saber si ha intentado hacer más de una vez alguna de las pruebas.

Para la creación de cada una de las tablas se ha utilizado Laravel, herramienta explicada anteriormente. Desde Laravel lo primero que se creó fue un modelo de datos y una migración por cada una de las tablas de la BBDD.
Las migraciones son ficheros que nos permiten crear tablas, establecer relaciones entre ellas, modificarlas y eliminarlas. En la migración de cada una de las tablas se especifican a parte de su propio nombre, los campos y sus respectivos tipos.

```
1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  class MemoryAlimentantionTable extends Migration
8  {
9      /**
10      * Run the migrations.
11      * @return void
12      */
13     public function up()
14     {
15         Schema::create('alimentation', function (Blueprint $table) {
```

```
16                $table->id();
17                $table->unsignedBigInteger('user_id');
18                $table->string('word_1')->nullable();
19                $table->string('word_2')->nullable();
20                $table->string('word_3')->nullable();
21                $table->unsignedInteger('fails');
22                $table->unsignedInteger('points');
23                $table->timestamps();
24                $table->foreign('user_id')->references('id')->on('users')->
                      onDelete('cascade');
25            });
26        }
27
28        /**
29         * Reverse the migrations.
30         *
31         * @return void
32         */
33        public function down()
34        {
35            Schema::dropIfExists('alimentation');
36        }
37    }
```

Como podemos observar, crear una tabla utilizando el sistema de migraciones es bastante sencillo, solo tenemos que definir los campos necesarios y las relaciones entre las diferentes tablas.
Los modelos son los componentes principales de las aplicaciones que siguen el patrón MVC. Se encargan de acceder a los datos, crearlos, modificarlos y eliminarlos.
Una vez creadas las migraciones y sus modelos correspondientes, se creó un controlador por cada uno de los modelos. Estos controladores son los encargados de agrupar la lógica de peticiones HTTP en una sola clase.

3.3. Control de errores

La fase de control de errores es un punto muy importante dentro del desarrollo software de una aplicación. En nuestro caso, realizamos esta fase después de obtener la primera prueba funcional en archivo ejecutable de la app. Durante el periodo de prueba realizamos reiteradas veces la ejecución normal del transcurso de la aplicación para intentar anticiparnos a posibles errores con los que se podrían encontrar los usuarios.
Los diferentes errores que fueron surgiendo durante el desarrollo de estas pruebas y las soluciones aplicadas han sido las siguientes:

- Ampliación de pantalla o zoom: desactivamos la posibilidad de hacer zoom en cada una de las pantallas de la APP. Ya que esto podría facilitar al usuario la realización de las diferentes pruebas. Como por ejemplo, en los casos de reconocimiento o localización de objetos.
- Giro automático de pantalla: esta funcionalidad se desactivó ya que la APP no está diseñada para realizar los test horizontalmente. Esto se debe, a que algunas de las vistas de nuestros

test de evaluación requerían de un espacio mínimo de pantalla para que el usuario pudiese visualizarlos correctamente y con el modo horizontal nos hubiésemos visto muy limitados en cuanto al espacio del diseño.

- Retroceso: se anuló el evento de ir hacia atrás para no perturbar el flujo de la APP e impedir el posible envío erróneo de datos al servidor. Para deshabilitar esta opción se tuvo que hacer de dos maneras diferentes, dependiendo del sistema operativo del dispositivo.

    - Para Android se hizo uso del BackHandler, recurso del framework de React Native, que funciona como un manejador de eventos. Un evento es cualquier acción que realiza el usuario al interactuar con nuestra aplicación. En concreto, el BackHandler es capaz de detectar cuando el usuario pulsa, en este caso sobre el botón que permite la opción de retroceso. Reconfiguramos este manejador para deshabilitar la función de este botón.
    - Para iOS se accedió al archivo de navegación en el que se encuentran todas las vistas de la APP. Cada una de las pantallas tiene ciertos atributos, como el nombre de la vista, etc. Uno de estos atributos es gestureEnabled, que permite habilitar o deshabilitar la opción de retroceso. En nuestro caso añadimos en cada una de las vistas gestureEnabled a false, para que la opción quedase deshabilitada.

- Exceso de tiempo o tiempo de inactividad: durante la fase de prueba de la APP nos anticipamos a un posible escenario en el que el usuario dejase de utilizar la aplicación o se encontrase en una misma pantalla durante un periodo de tiempo prolongado. Para controlar esta situación, hicimos uso de la función setTimeOut, otro recurso de React Native, disponible para ambos sistemas operativos. Esta función nos permitía, tras detectar 1 minuto de inactividad por parte del usuario, mostrarle un mensaje de alerta y darle la opción de continuar o salir de la aplicación.

- Error en la conexión de la BBDD: esto ocurre cuando no hay conexión a internet, ya que la BBDD se encuentra alojada en un servidor en AWS, por lo que si no hay internet no hay acceso a la BBDD y las respuestas del paciente no pueden ser enviadas ni guardadas en el servidor. En el caso de que el usuario se encuentre en esta situación, se le mostrará un mensaje de error de conexión y se le redirigirá para volver a realizar el test correspondiente, ya que sus respuestas no habrán podido ser guardadas.

- Control de tipos: este tipo de control recoge todos los posibles escenarios en los que el usuario pudiera introducir una respuesta diferente a la esperada. Por requerimiento de nuestra aplicación hemos tenido en cuenta los siguientes casos:

    - Por petición de los especialistas, todos los datos alfabéticos introducidos en la APP por los usuarios debían ser en mayúsculas. Para ello, en todos los campos donde se esperaba una entrada de texto, se configuró para que el teclado apareciese directamente en letras mayúsculas.

  - En el caso de que los datos de entrada que se piden al usuario sean específicamente numéricos se le mostrará un teclado numérico, en caso contrario, el teclado por defecto, el alfanumérico.
  - En los dos casos descritos anteriormente resolvemos los escenarios posibles donde el usuario introduce una entrada de un tipo diferente al esperado, pero también es posible que el usuario no introduzca ninguna respuesta, bien porque se haya olvidado o porque no sepa qué responder. Hemos configurado las preguntas y los test, de forma que todas las entradas sean requeridas obligatoriamente. Por lo que si el usuario intenta avanzar de pantalla sin haber respondido primero, le mostraremos un mensaje de alerta que le recuerde que debe completar el campo, o en el caso de que haya más de una entrada, los campos que haya dejado vacíos.

- Recuperación de contraseña: en el caso de que el usuario quiera acceder a la aplicación y no recuerde su contraseña, se ha habilitado una opción para que pueda recuperarla. Esta opción aparece en la misma pantalla de inicio de sesión y hemos implementado esta nueva funcionalidad utilizando la API. Primero el usuario deberá introducir su correo electrónico, nuestra API se conectará con la base de datos, verificará que el correo electrónico exista, y después de hacer esta comprobación, se generará un token de un solo uso, válido por dos horas, para que el usuario pueda cambiar su contraseña. Después de que la contraseña haya sido restablecida, se enviará un mensaje al usuario para confirmar el cambio de contraseña a su correo electrónico. Y se le redirigirá a la página de inicio de sesión de la APP para que pueda acceder con la nueva contraseña.

3.4. Planificación del proyecto

Para la creación o actualización de software de calidad, es imprescindible basarse en una metodología de desarrollo. Estas metodologías son definidas como un conjunto de métodos coherentes y relacionados por unos principios comunes para alcanzar un objetivo que requiere habilidades y conocimientos específicos; con el objetivo, de formalizar y optimizar las actividades que engloban el completo desarrollo, así como facilitar la organización de los integrantes del equipo. [29]

Existen principalmente tres tipos de metodologías, las clásicas, evolutivas y ágiles. Entre ellas, nos decantamos por usar una metodología ágil por las siguientes características que las definen:

- **Mejora de la calidad del producto:** fomenta la constante mejora de las propiedades del producto, Además, la interacción entre el cliente y los desarrolladores facilita el cumplimiento de las necesidades del cliente, lo cual conlleva a un buen resultado final.

- **Mayor motivación e implicación de los trabajadores:** interacción continua entre los miembros del equipo de desarrollo.

- **Entregas con mayor velocidad y eficiencia:** cada cierto periodo corto de tiempo establecido previamente, se hacen pequeñas entregas que muestran los avances del proyecto en

versiones funcionales, lo que permite la constante revisión del proyecto, corregir posibles errores e implementar mejoras.

3.4.1. Metodología SCRUM

De entre todos los tipos de metodología ágil, la más adecuada nos pareció la metodología Scrum. Este tipo de metodología ágil sirve para gestionar proyectos que implican desarrollar productos, organizar a los integrantes por roles y controlar el tiempo.

Su propósito principal es realizar a tiempo todos aquellos requisitos que se van a llevar a cabo en el proyecto. Es por esto, que al inicio del proyecto se define, como si fuera una lista de deseos, las características y requisitos que debe cumplir el producto a desarrollar; a esta lista se le llama "Product backlog". De esta lista se seleccionan los requisitos que se quieren cumplir para la fecha de entrega prevista, que sean factibles. A esta lista actualizada se le llama "Product release" y será el objetivo a cumplir.

La gestión de los roles es muy importante, Scrum abarca a todas las personas implicadas en un proyecto y les asigna un rol. Existen roles esenciales en Scrum como el llamado "Product owner", tiene una visión general de que limitaciones técnicas existen y también que necesita el cliente en el producto. Otro rol importante es el llamado "Scrum Master", es decir, el jefe de proyecto, que monitoriza el proyecto y controla que el plan se cumpla y la entrega se haga a tiempo. Otros roles a destacar son los desarrolladores y los testeadores que no tienen nombre especial, pero que son tan importantes como los otros y sin ellos el proyecto no saldría adelante.

Una vez definidos estos roles, falta estimar las tareas del "Product release". Scrum estima las tareas por horas, días o meses dependiendo del tamaño de las mismas. Las más grandes se pueden descomponer en subtareas de menor tiempo.

Tras la estimación de las tareas, aparece la idea del "Sprint", que es un periodo de tiempo de entre dos y treinta días en el cual se van a realizar determinadas tareas, seleccionadas al inicio del mismo de las existentes en el "Product release".

Por último, y con el objetivo de tener un seguimiento del progreso de los "Sprints", se realizan reuniones periódicas en las que todos los integrantes del proyecto pueden hablarse sin jerarquías, donde se comparten dudas, opiniones, se comentan los avances y se contraponen ideas. Estas reuniones pueden ser diarias "Daily meeting", semanales "Weekly meeting" o con la frecuencia que se considere, pero en su esencia no deben sobrepasar los quince minutos.

A continuación, explicaremos cómo hemos integrado estos conceptos en nuestro proyecto.
Debido a que es un equipo formado por cuatro integrantes, lo cual requiere división de tareas y organización. Y con la intención de tener nuestro proyecto controlado y cumplir con el tiempo de entrega, se ha seguido esta metodología. En el cual el "Scrum Master" era el tutor, el "Product owner" en este caso era cada integrante sobre el trabajo en común y los 4 alumnos hemos pasado por severas fases de desarrollo y testeo.

Desde que se presentó el proyecto se declaró un “Product backlog”, definido por cuáles y cómo iban a ser las características de nuestra aplicación. De este emanó el llamado “Product release”, que vinieron a ser, cuáles características íbamos a conseguir y cuales vamos a cumplir a la hora de entrega.

Con ánimo de cumplir todas estas tareas, se estableció una “Weekly meeting” con el tutor, en la cual se compartían los progresos, se contrastaban ideas y se resolvían dudas. Así como, se definían qué tareas se iban a cumplir para la semana siguiente. Por lo que de esta forma definíamos nuestros “Sprint” de un mes de duración.

Por último, acordábamos reuniones para avanzar sobretodo tareas en equipo de desarrollo. En ocasiones en grupos de dos o tres personas. Además, los domingos se realizaba el “día de control” como "Weekly meeting", en el cual se comentaban los avances internos y organizaban las tareas restantes para finalizar el Sprint sin ítems pendientes.

3.4.2. Tabla de tareas Kanban en Trello

Aunque la metodología Scrum es muy completa, otra herramienta visual de trabajo que nos ha resultado muy útil es la tabla Kanban que se basa en dividir el trabajo en tareas concretas en forma de tarjetas. Estas pasan por tres estados: “Por Hacer”, “Haciendo”, “Hecho”. Esta visualización, nos ha servido para tener siempre una imagen clara y global de en qué estado estaba tanto el proyecto, así como las diferentes tareas o trabajos y a quien estaban asignadas.

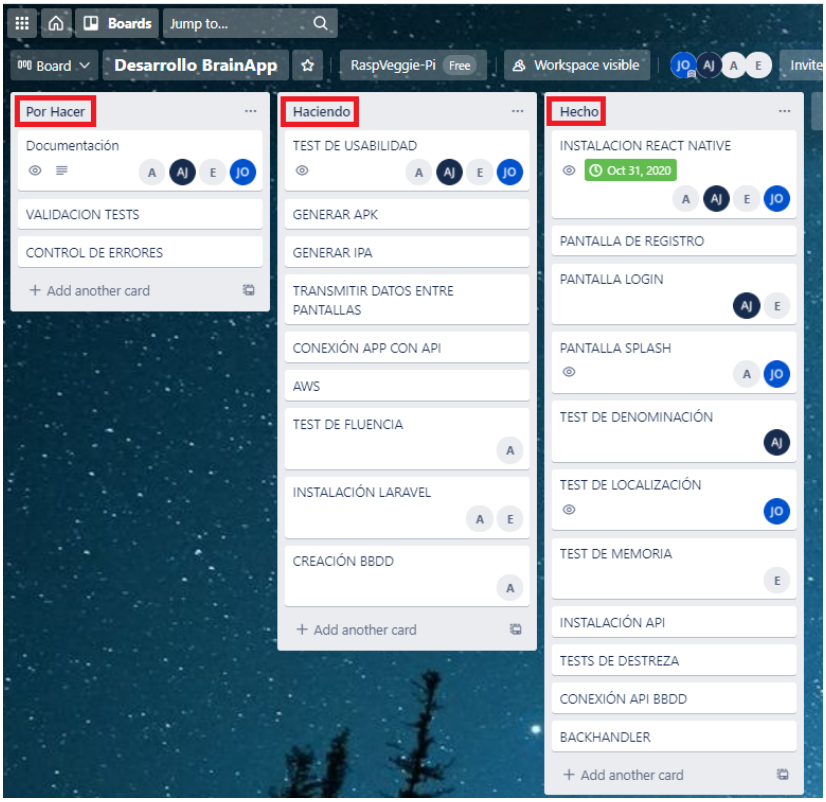


Figura 3.18: Planificación en Trello.

Las tarjetas solían representar tareas concretas, hitos y listas de acciones a realizar por parte de todos o algunos de los desarrolladores o testers.
Este entorno nos lo facilitó la plataforma web Trello, que guardaba, actualizaba y comunicaba (vía email) de forma distribuida e inmediata todas las tareas y sus estados cada vez que se creaban, asignaban o cambiaban de estado las tarjetas.

3.4.3. Organización General

1. **Departamento de Neurología**:
   El departamento de neurología del Hospital Clínico San Carlos, que propuso el desarrollo de la aplicación para evaluar las capacidades cognitivas de sus pacientes, está formado principalmente por el neurólogo Jordi Matias-Guiu Antem, como jefe y supervisor en la parte clínica de este proyecto, así como Vanesa Pytel Córdoba, como neuróloga y encargada del test de denominación del lenguaje y del test de memoria, Alfonso Delgado Álvarez como neuropsicólogo encargado del test de localización de objetos y del test fluencia de diseños y Cristina Delgado Alonso, como ayudante de neurología, encargada de supervisar los test que realizaron dos de los alumnos en el hospital.

2. **Equipo de desarrollo**:
   El equipo de desarrollo está formado por los alumnos que realizan este proyecto: Arantxa, Andrea, Jhoselin y Jaime en calidad de programadores, investigadores y redactores de forma predominantemente equitativa y en función de las necesidades del proyecto en cada momento. Así como el tutor Jose L. Ayala que a parte de poner en contacto a los médicos y los desarrolladores, también cumplía el rol de consultor para temas de arquitectura, diseño o tecnologías empleadas.

3. **Reuniones con el tutor**:
   Con el ánimo de tener seguimiento del desarrollo y de los avances, se acordó desde un inicio a tener reuniones periódicas, en las cuales el tiempo entre reunión era de por defecto de una semana y fue variando en función de la disponibilidad, periodos de exámenes, vacaciones o sprints de desarrollo.

   De forma asíncrona, para consultas y dudas utilizó la comunicación vía email.

4. **Viernes neurocomputacional**:
   Un viernes de cada mes, se produce una reunión en la que se tratan temas relevantes sobre diferentes proyectos neurocomputacionales, en la que están invitados médicos del Hospital Clínico San Carlos, investigadores y desarrolladores de ambos campos.

   En este encuentro los diferentes participantes realizan presentaciones del estado del arte y los progresos de sus proyectos. Tras cada presentación se aportan ideas y se hacen comentarios y se contrastan los resultados obtenidos.

   En lo que nos compete de este proyecto, se participó en el mes Septiembre. Se expuso a los asistentes una fase de análisis y de prototipos. Se analizaron los test que habitualmente

se realizaban para estudios de capacidades cognitivas, también se diseñaron prototipos para la implementación de dichos test en dispositivos móviles. Tras la presentación, los doctores nos hicieron comentarios sobre los test y nos dieron pautas para el futuro. Así, nos propusieron diseñar nuevos test diferentes a los habituales, pero que evaluaran las mismas capacidades cognitivas. Es entonces cuando se definió que la forma de trabajo y comunicación entre desarrolladores y doctores iba a ser más directa, a continuación profundizamos en esta metodología de trabajo.

5. **División en subgrupos**:
   Tras las varias reuniones con todos los integrantes de este proyecto, y con el fin de tener una comunicación más fluida, poder consultar dudas y compartir la viabilidad técnica de los test, el tutor propuso asignarlos, de tal forma que cada test iba a ser abordado por una pareja de doctor-alumno. Quedando organizado de la siguiente manera:

   - **Vanesa - Andrea**: Test relacionado con la denominación de objetos.
   - **Jordi - Jhoselin**: Test relacionado con la memoria
   - **Alfonso - Jaime**: Test relacionado con la detección de objetos y test relacionado con la fluencia de diseños.

   Participando en cada uno de los subgrupos Vanesa, Jordi y Alfonso en calidad de supervisores clínicos.

   Mientras tanto, **Arantxa** se encargaría de preparar la estructura de la aplicación y la arquitectura de comunicación con la API y la base de datos.

   Estos grupos trabajarían de forma ágil y elegirían entre ellos tanto la vía de comunicación ya fuera por mensajería instantánea, como por Meetings online y la frecuencia de los mismos.

6. **Drive Compartido**:
   El tutor, creó una carpeta compartida en Google Drive, a la que tenían acceso tanto los alumnos como los médicos, en la cual se agregaron documentos relativos a los diferentes test existentes, documentación relevante a programas de ordenador que evaluaban capacidades cognitivas, y otros estudios de interés sobre el tema.

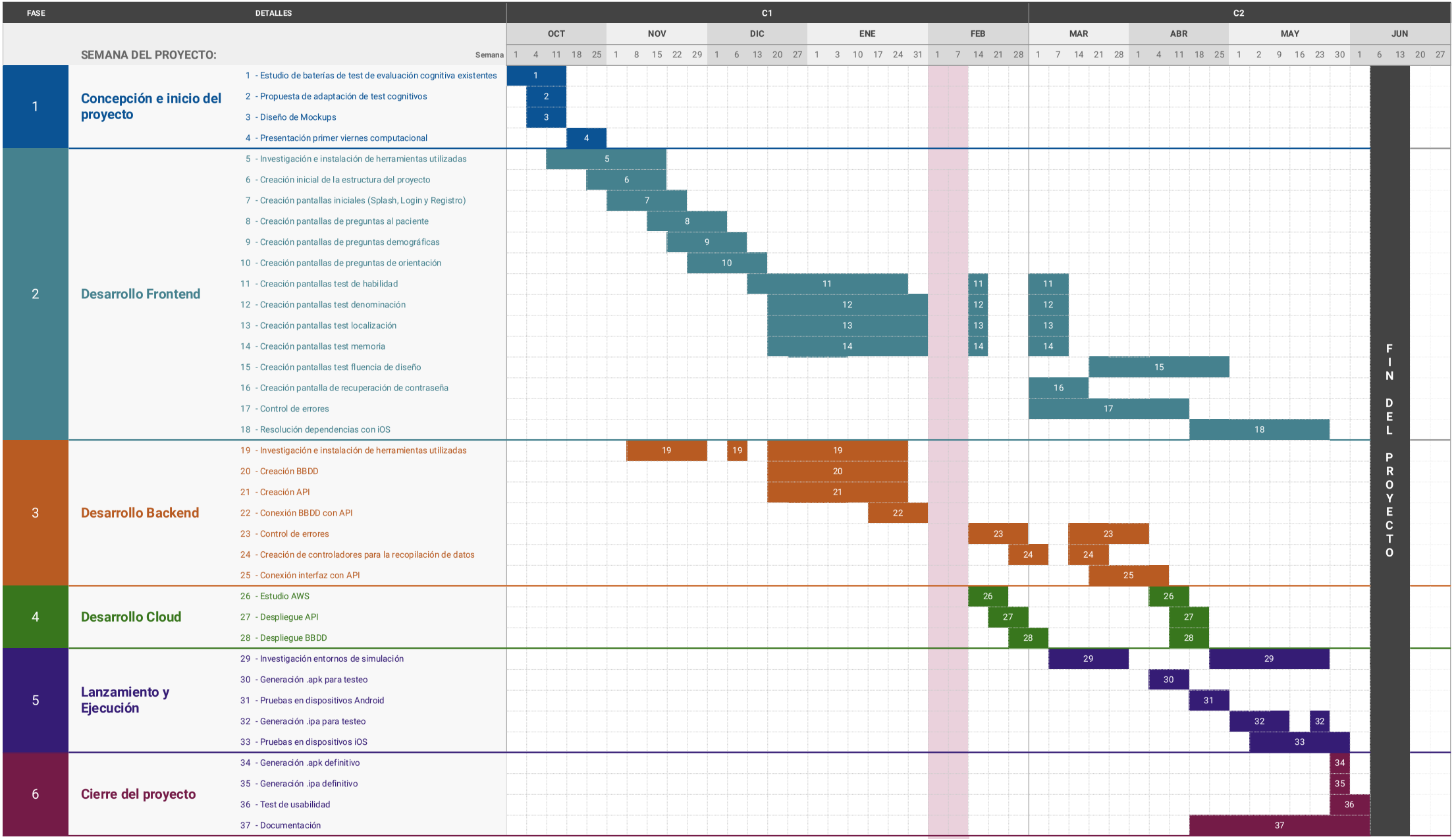
   Este espacio, sirvió también como lugar de trabajo en la definición de los test, así como comentarios y actualizaciones. Que daban constancia por escrito de requisitos, correcciones, avances, etc.

3.4.4. Cronograma

A continuación se puede observar el cronograma completo del proyecto, que abarca desde el 01/10/2020 hasta el 06/06/2021. En la siguiente imagen se pueden contemplar las diferentes fases de desarrollo del mismo, así como las tareas que forman cada una de las fases.

Cronograma BrainApp

TÍTULO DEL PROYECTO	BrainApp
FECHA DE INICIO	01/10/20
FECHA DE FIN	06/06/21



3.4.5. Descripción individual del trabajo realizado

En esta sección, vamos a describir individualmente el trabajo realizado por cada uno de los integrantes del grupo, mencionando las tareas reflejadas en el cronograma mostrado anteriormente. Está descripción se realizará de la siguiente manera:

1. **Descripción I: Andrea**
2. **Descripción II: Arantxa**
3. **Descripción III: Jaime**
4. **Descripción IV: Jhoselin**

De todas las tareas y subtareas identificadas en el cronograma anterior, cada integrante de nuestro grupo se centró y realizó una mayor participación en algunas en concreto. Fuimos repartiendo los puntos a trabajar y desarrollar en función de nuestras aptitudes, aunque realizamos una participación mínima en todas las tareas, para mantener una comprensión y entendimiento general sobre cómo estaba avanzando el proyecto.

Descripción I

Como mostramos en el cronograma, las primeras tareas realizadas estaban orientadas sobre todo al estudio de las baterías de test cognitivos utilizados actualmente, aplicaciones móviles que realizan una evaluación cognitiva y de comportamiento parecido al de nuestra futura aplicación, también páginas web y algún otro tipo de herramienta digital que se utilizase actualmente para la evaluación cognitiva. Cada uno de nosotros realizó este estudio inicial por separado.

Tras pasar los primeros días trabajando, mantuvimos una reunión junto con dos doctores y un neuropsicólogo del departamento de neurología del Hospital Clínico San Carlos, ellos mismos nos propusieron realizar una adaptación de los test que se realizaban en su departamento. Fueron varios los tipos de test propuestos por lo que decidimos centrarnos en uno en concreto y repartírnoslo. Yo en especial me centré en el VOSP Test, y empecé investigando estudios publicados sobre su metodología para aprender cómo funcionaba y se presentaba a los pacientes. Fuimos haciendo algunas reuniones para ir poniendo en común nuestras ideas para la adaptación de los test, así como las primeras características generales que debía tener nuestra aplicación.

Como es habitual en el desarrollo software, previamente al diseño de una aplicación se realizan una serie de maquetaciones y bocetos para hacerse una idea general de lo que quieres construir. Estos bocetos se les conoce como mockups y se realizan con la ayuda de alguna herramienta digital o también a papel. Se escogió realizarlo con la ayuda de la herramienta digital Adobe XD que hemos mencionado en el apartado 3.2 Herramientas Utilizadas.

A parte de diseñar algunos bocetos para pantallas generales, de forma individual, se realizaron mockups para las posibles adaptaciones de nuestros test. Yo en particular, me centré en la creación de los mockups de la adaptación de dos subtest (dado que el VOSP Test es una batería que conformada por varios subtest) del test en el que me había centrado.

Tras una presentación de nuestras primeras propuestas y una reunión con nuestro coordinador y los doctores, se nos detallaron las pruebas que debían incluirse en la aplicación; estas serían cuatro pruebas diferentes y también una serie de preguntas generales y de orientación. Para trabajar de forma más sencilla en el desarrollo de cada una de las pruebas, decidimos hacerlo por parejas; cada pareja se conformaba por un especialista (doctor o neuropsicólogo) y uno de los integrantes de nuestro equipo. Yo trabajé con Vanesa, neuróloga del hospital con el que hemos colaborado y realizamos tanto la adaptación como otras características del Test de denominación del Lenguaje que hemos mencionado en la introducción del apartado 3. Metodología.

Para empezar con el desarrollo frontend tuvimos que pasar por un periodo de estudio y aprendizaje, así como la instalación previa de todas las herramientas necesarias para empezar a programar con el framework React Native.

Después de terminar la correcta instalación del software necesario, nos pusimos en concreto con la creación de la estructura del proyecto. El inicio de este proyecto se creó en Visual Studio Code y lo subimos a GitHub para poder compartir todas las modificaciones entre todos los integrantes del grupo.

Una vez creado el proyecto inicial y decidida su estructura, empecé con la creación de las pantallas iniciales, Splash (pantalla de inicio a la aplicación), la pantalla de Login (Inicio de Sesión) y la pantalla de Register (Registro). El código desarrollado lo llevé a cabo junto con otra de los integrantes de nuestro equipo repartiéndonos equitativamente el trabajo.

Una vez terminadas las vistas de estas pantallas y tras otra reunión con los doctores del Hospital Clínico, concretamos las preguntas iniciales que se realizarían a los pacientes previamente y participé de la misma forma que he indicado para el desarrollo de las pantallas iniciales, es decir desarrollando el trabajo de forma equitativa junto con un integrante más.

Todo el cuestionario inicial lo comprendían diez pantallas de preguntas generales, dos pantallas de preguntas de orientación, y cuatro pantallas de preguntas demográficas, sumando un total de dieciséis pantallas que nos dividimos a la mitad. Realicé la creación de esas pantallas, así como las pantallas de información necesarias al inicio de cada tipo de bloque de preguntas.

Terminando esta parte, empecé con el desarrollo y creación de las pantallas para el Test de Denominación del Lenguaje. Al ser esta una prueba adaptada de un test cognitivo necesité reunirme varias veces con Vanesa, para concretar algunas especificaciones del diseño y de la funcionalidad. Este test estaba compuesto por veinte pantallas, y además de forma posterior se añadieron dos pantallas más para incluir demostraciones de la prueba y ayudar al paciente a entender como se realizaría. A todas estas pantallas se le sumó la creación de otras pantallas de información para indicar al paciente el orden y el tipo de test a realizar además de otras pantallas para mostrar las instrucciones.

Una vez fuimos terminando la parte frontend, se realizó un proceso para el control de errores en la aplicación. Realizamos gracias a los entornos de simulación Android Studio y XCode una revisión

del funcionamiento de nuestra APP, y recogimos un listado de posibles escenarios con los que nos podríamos encontrar y podrían dar error.

Dentro de todos los tipos de control de errores, yo me centré en los de la parte frontend, en especial en los posibles casos en los que el usuario estaba inactivo por un tiempo prolongado, también en control de tipos, que es cuando el usuario introduce una entrada de un tipo diferente al esperado, esto es por ejemplo cuando esperamos una entrada de tipo numérico y la entrada recibida sea de tipo string y viceversa. Otro tipo de control de errores que apliqué fueron el control de entradas vacías, para evitar en muchos de los casos donde necesitamos saber la respuesta del usuario a una pregunta y se le impidiese pasar a la siguiente pantalla o volver hacia atrás hasta que contestase, esto se hizo informando con un mensaje de alerta. Después de la aplicación de control de errores a la aplicación, se volvió a realizar la simulación completa de la secuencia de la aplicación para comprobar que estos controles de errores aplicados funcionaban correctamente.

Para la parte de desarrollo backend, también se realizó en grupo un estudio de las herramientas que necesitaríamos, así como la sintaxis que queríamos utilizar y el servidor que íbamos a escoger para almacenar nuestra BBDD. Una vez acordado tuve que aprender al principio el funcionamiento de estas mismas y realizar la instalación correcta de todas las herramientas (Laravel, MySQL Workbench, Postman, etc.) para empezar a trabajar con ellas y realizar futuras pruebas.

Cuando fueron creadas por otro de los integrantes del grupo la API y BBDD, realicé varias pruebas necesarias para conseguir la recepción adecuada de las respuestas del Test de Denominación del Lenguaje. Para el posterior despliegue en la nube de nuestra API y BBDD también tuvimos que estudiar como funcionaban los servicios que nos ofrecía Amazon Web Services y como podíamos utilizarlos.

Cuando terminamos con una primera versión funcional de la aplicación debíamos realizar pruebas de testing para comprobar el correcto funcionamiento de esta. Para ello investigamos en como generar el fichero ejecutable de nuestra APP y una vez obtenido realizamos las pruebas. En mi caso cabe destacar que al no disponer de un dispositivo iOS no realicé pruebas para esta versión, pero si las realicé para Android. De esta forma pudimos encontrar algún fallo o punto de mejora, solucionarlo o mejorarlo antes de obtener la versión funcional.

Por último, realicé en conjunto con mis compañeros el diseño de la estrategia de evaluación de usuarios y las preguntas que íbamos a aplicar en el test de usabilidad.

Descripción II

Las primeras tareas realizadas fueron investigar las baterías de los test, aplicaciones, páginas web y herramientas utilizadas para la evaluación cognitiva mediante la realización test neurocognitivos.

A su vez, se nos pidió de algunos de los test utilizados en el Hospital Clínico San Carlos en la actualidad, una posible adaptación propuesta por nosotros. Para ello, hubo que estudiar cada uno de los test, como las áreas cognitivas que evaluaban, y hacer varias reuniones entre los integrantes del grupo para poner en común todas nuestras ideas.

Después de una investigación profunda de varias semanas y una idea común aceptada por todos, realizamos una presentación en la que proponíamos nuestras adaptaciones, incluyendo mockups realizados por nosotros mismos, para que pudiese quedar clara la idea propuesta.

Tras esa presentación, y reuniones con nuestro tutor y el grupo de neurólogos, se nos explicó en qué iba a consistir finalmente la aplicación que íbamos a desarrollar y las diferentes pruebas que dicha aplicación iba a contener. Para ello, se dividieron las principales tareas en subgrupos dentro del grupo de trabajo para hacer más dinámico el trabajo.

Para la parte de frontend, decidimos trabajar con React Native, puesto que es un framework con el que ninguno había trabajado antes, tuvimos que investigar la forma de utilización del mismo, así como su correcta instalación.

Una vez instaladas las herramientas para empezar con el desarrollo del frontend por cada uno de los integrantes del grupo, mientras mis compañeros se dedicaban al desarrollo de la parte frontend, yo me dedicaba a la parte del backend.

No se nos propuso ninguna herramienta en concreto para esta parte, por lo que tras reuniones entre el grupo de trabajo e investigar cada uno aquellas que nos pudiesen ser más fáciles y comprensibles de usar. Decidimos utilizar Laravel, ya que algunos integrantes del grupo habían trabajado antes con este framework y hacer una BBDD sql. Aun así tuve que estudiar su forma de instalación y utilización, ya que era completamente nuevo para mí.

Tras la instalación del framework y todas las herramientas que se necesitaban para su utilización, como MySQL Workbench para la visualización de la BBDD, Laravel para hacer pruebas de inserción de datos en la BBDD, etc. Creé la API que utilizaríamos para el proyecto.

Después de varias reuniones entre los integrantes del proyecto, exponiendo a mis compañeros mis ideas y presentando un posible esquema sobre la BBDD, las principales tablas que formarían la API y los datos que se guardarían en cada una de las tablas; comencé con la creación de la BBDD que iría integrada en la API creada anteriormente.

Esto supuso la creación de las primeras migraciones, modelos de datos y controladores para cada una de las primeras tablas de la BBDD. Estas fueron: la tabla de usuarios (users), la tabla de preguntas al paciente (questions), la tabla de preguntas demográficas (demographics) y la tabla de preguntas de orientación (orientation).

En este caso, el modelo de datos de cada una de las tablas contiene los diferentes atributos que se guardan en la tabla; así como sus respectivos controladores que se encargan de validar los datos introducidos por el paciente, es decir, que correspondan con los que admite la tabla, como los tipos de datos (string, bool, etc.) y de crear la fila correspondiente en la tabla.

Tras la creación de las tablas, con ayuda de Postman, comencé a hacer pruebas para la inserción de los datos en cada una de las tablas. A su vez, se expuso un control de errores, en el que destacan el caso de que los datos introducidos no correspondían con los requeridos, que no hubiese conexión con la BBDD, etc. Una vez me aseguré de que los datos se incluían correctamente, se conectó la BBDD con la API.

Una vez conectada la API con la BBDD y asegurándome de que seguía funcionando correctamente, se comenzaron a conectar cada una de las tablas del backend con su parte correspondiente en el frontend.
Este proceso fue algo más largo, porque iban surgiendo errores tanto en backend como en frontend y entre mis compañeros y yo tuvimos que ir solucionándolos.

Cuando la parte de registro, inicio de sesión y el guardado de las primeras respuestas de los usuarios estuvo correcta, procedí a crear las tablas de las pruebas principales, en el siguiente orden: tabla del Test de Localización, tabla del Test de Denominación del Lenguaje, tabla del Test de Memoria y finalmente la tabla del Test de Fluencia de Diseño. Para ello se crearon sus migraciones, modelos de datos y controladores correspondientes.
A diferencia de las cuatro primeras tablas que creé al comienzo, estas fueron algo más complejas de crear. Esto se debe a que a parte de guardar cada una de las respuestas del usuario que está haciendo la prueba, se calculan el número de fallos y el número de aciertos finales. Por lo que cada uno de los controladores además, contiene la lógica correspondiente en lenguaje php para obtener las puntuaciones correspondientes.

Tras crear las diferentes tablas y ser probadas de nuevo con Postman, las conecté con sus correspondientes partes en frontend. Una vez hecha esta conexión en cada una de las pruebas, se comenzó un proceso largo de pruebas, ya que al comienzo no se introducían bien en la base de datos, o directamente la conexión con la BBDD fallaba.

Finalmente, cuando todo el flujo de frontend y backend funcionaba correctamente, se creó la instancia en AWS de la API y la BBDD. Haciendo previamente un estudio de como hacer el despliegue de ambas en AWS. Cuando la instancia se creó y funcionaba correctamente desde nuestros ordenadores, investigué sobre la forma de generar un archivo ejecutable para poder probar la APP desde un dispositivo móvil.

Tras su correspondiente investigación,tuvimos que generar dos ejecutables diferentes, uno para Android, llamado .apk y otro para iOS llamado .ipa.
El .apk podía ser generado desde cualquier ordenador, pero el .ipa únicamente se podía hacer desde un ordenador con sistema operativo macOS.
Por lo que me centré en generar el correspondiente .apk, ya que el sistema operativo de mi ordenador es Windows. Este proceso nos llevó bastante tiempo, porque nos fuimos dando cuenta de errores en el flujo de las pantallas que tuve que solucionar con mis compañeros, como los contadores de tiempo que generaban errores en el paso de pantallas, correcciones de estilos en las pantallas, etc. En este proceso también estuvo presente nuestro tutor, ya que cada .apk generado a parte de ser probado por nosotros mismos, fue probado por él y nos iba dando feedback sugiriendo algunos cambios.

Se hicieron todas las correcciones pertinentes, y posteriormente me centré en generar el .apk definitivo con el cuál hemos presentado este trabajo, y que tienen tanto el tutor cómo el grupo de neurólogos, para hacer sus correspondientes pruebas.

Al mismo tiempo del período de generación de archivos ejecutables, junto con mis compañeros propusimos una serie de preguntas que servirían de ayuda para mejoras futuras de la APP, como test de usabilidad. El cual fue aprobado por nuestro tutor y difundido a los doctores.

Descripción III

Como podemos ver en el cronograma, la primera fase es el estudio de baterías de test de evaluación cognitiva existentes. En ella los analizamos, para entender cuál era su función y cómo se realizaban, qué normas había que seguir para su realización y cómo interpretar sus resultados. Tras esto, realicé test de forma presencial, en el hospital. El programa que utilicé ("Vienna Test System"), implementa una batería de test, que me proporcionó información sobre cómo se disponían test en una plataforma digital, en vez de los tradicionales, en papel. Después de esta fase de análisis de las pruebas cognitivas, participé en las propuestas de adaptaciones de test concretos a una aplicación móvil. Esta fase estudió cuales eran más viables y qué ventajas y desventajas se nos presentan al utilizar un smartphone, como medio para las pruebas. Lo más importante de esta fase es la generación de nuevos test. Es decir, test que evaluaran las mismas capacidades cognitivas, pero fueran diferentes a los originales.

De forma complementaria, estudié test similares, existentes en otras entidades. Busqué características como: qué evaluaban sus test, si usaban sonidos, si necesitaban de supervisión y si eran de pago. El siguiente paso fue la realización de mockups y prototipos de las adaptaciones. En concreto, yo realicé la adaptación del test de "La Torre de Londres". Este test disponía reglas similares, pero en lugar de utilizar torres y discos, utilizaba un tablero vertical y fichas de colores. También, el número de "columnas" era superior y el número de fichas de las que disponía, no era limitado. Esta fase de estudio fue expuesta en el primer viernes computacional, en la que participé activamente y pude contrastar lo propuesto, con los doctores.

Seguidamente, vino la fase de instalación de las herramientas. Antes de empezar el proyecto no había tenido contacto con tecnologías de aplicaciones para dispositivos móviles. El Framework "React Native", fue el primer paso en este mundo, y con el lenguaje de programación en el que está basado, JavaScript. Por ello, al inicio, busqué documentación oficial, foros, estudié tutoriales y entendí cómo era la arquitectura del Framework. Tras esta inmersión en la tecnología, participé de forma pasiva durante la creación de las primeras pantallas que se presentan en nuestra aplicación. Aportando ideas, investigando cómo poder representar ciertas características y probando algunos componentes que pudieran resultar de ayuda.

Como se acordó, fui el encargado de adaptar el Test de Localización. Partiendo de la descripción propuesta por el neuropsicólogo, estudié cómo poder implementar este test de la mejor forma. En un principio, existieron varios requisitos que abordar. El primero era cómo llevar este test a la temática de la APP (supermercado), puesto que la descripción inicial, eran dos cuadrados. Uno superior con números dispuestos aleatoriamente y otro inferior con un punto que representa la posición de un número concreto, de los del cuadrado de arriba. Tras la validación del neuropsicólogo, encontré una solución que fue la definitiva: los cuadrados se convertirían en "carritos de la

compra" y los números, en piezas de fruta. Llegado a este punto, la parte visual estaba cubierta. Faltaba, entonces, implementar la parte interna en la que se iban a recoger las acciones del usuario, como contabilizar un fallo o un acierto. Para implementar esto, busqué la mejor forma que me ofrecía JavaScript para recoger la información y opté por dividir la pantalla en cinco celdas. Cada celda era un área invisible táctil que recogía la información de la pulsación del usuario. Todo ello, para situar el área de acierto, exactamente, sobre la fruta que era la correcta y descartar las áreas colindantes. Esto hacía el código reutilizable y que sólo hubiera que modificar los tamaños de las áreas en función de dónde se encontrará la fruta a elegir. Una vez el test cumplía los requisitos, se acordó que tendría dos pantallas demo con ejemplos, diez pantallas de evaluación y la puntuación se recogería como aciertos o fallos.

Otro de los test que realicé fue el de Fluencia de Diseño. Este test, también descrito por el neuropsicólogo, constaba de la realización de diseños únicos uniendo cinco puntos con líneas. Se sustituirían los puntos por fresas, pero las líneas se mantendrían. Una vez definido visualmente, la forma de aplicarlo fue la siguiente: las líneas de unión, serían implementadas como botones, que al pulsarlos pintarían una línea, entre las dos fresas más cercanas entre sí. Si se pulsaba de nuevo, la línea desaparecía. Respecto a la puntuación, se definió que el usuario pudiera crear un máximo de quince diseños (únicos o no), y que como máximo, se dispusiera de un minuto. Para esto último tuve que añadir un temporizador. En este test se utilizó una única pantalla que se iba modificando en función de las acciones del paciente. Cuando este finalizaba un diseño, se pulsaba un botón en la parte inferior que decía: "Siguiente". En este momento se almacenaba la información del diseño realizado y se limpiaba la pantalla. La demo disponía de tres ejemplos, en los que se le informa al paciente cómo se deben realizar los diseños. También se le advertía si repetía alguno. Ello supone el análisis de los diseños y la comparación entre sí. Una vez terminada la prueba se envía una lista con todos los diseños, de donde se extrae la información sobre: diseños únicos, repetidos y porcentaje de acierto.

Respecto a los errores relativos al diseño, las dos pruebas que implementé, eran test en los cuales la interacción a interpretar, por parte del usuario con el smartphone, era la posición de la pulsación y también, esta posición respecto de lo que se representaba en la pantalla. Era muy importante que los diferentes tamaños de pantalla no afectasen a la representación visual de los elementos, ni a la posición relativa tocada por el paciente. Esto me obligó a probarlo en diferentes pantallas. Las cuales mostraban las imágenes de forma desestructurada y hubo que refactorizar los estilos de los componentes visuales, para que usaran porcentajes relativos a la pantalla y no a píxeles. Otra resolución de bugs que tuvieron que ver más con toda la aplicación en sí, fueron: el control de fugas de memoria y el mal uso de temporizadores. Así como, errores relativos al traspaso de información entre pantallas. Por último, en la fase proactiva del control de errores que pudiera realizar el paciente, investigué e implementé la forma en la que los pacientes no pudieran ir hacia atrás, en mitad de una prueba; evitar que la aplicación se pusiera en modo apaisado, y finalmente, las alertas por exceso de tiempo sin actividad.

Ya que era el único alumno que disponía de un equipo Mac, fui el encargado de realizar la versión de la aplicación en el sistema operativo iOS. Una de las ventajas de usar React Native es que

el código es válido para ambos entornos. Tal y cómo organizamos el proyecto, la aplicación se desarrolló inicialmente en Android y luego, se "transformó" la aplicación a iOS. El proceso inicial de instalación del entorno fue parecido, aunque tuvo sus dificultades, puesto que ciertos pasos eran únicos para este sistema operativo y la documentación no era tan abundante. Una vez salvado esto, tuve que reinstalar todas las dependencias usadas en Android, más sus homólogas en iOS. Por cada componente que utilizábamos, de una biblioteca, tenía que buscar un equivalente para este sistema operativo. La mayoría de las bibliotecas tenían homólogo, pero no todas. Entre las que no tenían homólogo encontré: una relativa al uso de imágenes vectoriales, otra al manejo de ir a la pantalla anterior y otra relativa a la fuente de textos, entre otras. Es por esto que puede verse alguna mínima variación visual entre los dos sistemas operativos. Después de este proceso de adaptación, conseguí que la aplicación funcionase idénticamente en iOS.

En lo referente al Backend y el desarrollo en Cloud, participé en la investigación de qué tecnologías utilizar y en la creación de la estructura y tablas de la base de datos, para almacenar los resultados de los tests y sus puntuaciones. También, y como proceso intermedio al almacenamiento, ayudé en la creación de los algoritmos de los controladores para la recopilación de datos, puesto que para test, como por ejemplo el de Fluencia de Diseños, había que comprobar los diseños únicos de los repetidos. Probé el envío de datos desde la APP a la API. También en el desarrollo Cloud, la investigación de AWS, y la máquina virtual EC2, en la cual se iba a instalar y desarrollar la API de Laravel.

Otra de las fases importantes, antes de testear la APP en los dispositivos físicos, fue la emulación de los dispositivos. Gracias al Framework, podíamos emular casi cualquier tipo de dispositivo Android e iOS, y al hacer cambios, verlos de forma dinámica; por lo que a la hora de encontrar errores o bugs fue más ágil. Una vez se resolvieron todos los errores y la simulación emulaba exactamente lo que se pedía, investigué en la generación para Android, del archivo de aplicación .apk: cómo había de generarse, cómo se generaban los certificados y cómo se firmaba y se generaba el mismo. Tanto mis compañeras como yo, instalamos la APP en dispositivos móviles y testeamos cada una de las pantallas, resolviendo algunos errores que surgieron al pasar a este medio. En lo relativo a la generación de archivo de aplicación en iOS, la dinámica era muy diferente a cómo se hacía en Android. Era necesario abrir el proyecto desde Xcode y tener una cuenta de desarrollador, ya que, para poder firmar una APP en iOS, debes tener una cuenta de "Apple Developer Program" de suscripción y pago anual. Con ella puedes generar certificados y firmar la aplicación, y de esta forma generar el archivo, distribuir o subirla al "AppStore". Esto fue una barrera que me impidió avanzar y tuve que consultar a compañeros que se dedican al mundo del desarrollo iOS. Tras su consejo y varias pruebas, utilicé una cuenta personal para la generación de un "Certificado local" para poder firmar la aplicación. Dicho certificado expiraba a la semana y no se podía distribuir. No obstante, me permitía generar el archivo y lo más importante, instalar la APP en local a un dispositivo iPhone. Gracias a esto, pude hacer las pruebas físicas de la aplicación, aunque con limitaciones. Logré, finalmente, resolver los pocos errores que surgieron y la aplicación tenía funcionalidad correcta pasando el test en dispositivo. El último hito, tras aprobación, fue la generación del el último archivo de aplicación .ipa.

Respecto al test de usabilidad, ayudé en la creación de las preguntas e investigué la mejor opción para realizarlo de cara a resultados. Encontré "Google Forms" como la herramienta más indicada para ello.

Descripción IV

Como hemos mostrado anteriormente, en el cronograma de planificación, durante las primeras tareas nos enfocamos en realizar búsquedas de información sobre los distintos test de evaluación cognitiva existentes, para poder hacer adaptaciones de estos a dispositivos móviles. Tras recolectar la información necesaria, en un primer momento mantuvimos reuniones con nuestro tutor para poder obtener requisitos como por ejemplo el tipo de dispositivo al que estaría orientada nuestra aplicación, en esa reunión se decidió que nuestra aplicación estaría orientada a todo tipo de público, por ello decidimos que la mejor opción era desarrollar una aplicación para smartphones ya que la mayoría de personas dispone de uno en casa, y es de fácil uso.

Después de recolectar la información y saber el tipo de dispositivo al que estaría orientada nuestra aplicación, empezamos con la creación de los mockups, para luego poder mostrarlos en una reunión a dos doctores y un neuropsicólogo del Hospital clínico San Carlos, en dicha reunión ellos nos propusieron las adaptaciones de los test que ellos utilizaban en las consultas del Hospital clínico San Carlos. Los test propuestos por el departamento de neurología fueron cuatro, junto con preguntas generales, de orientación, de datos demográficos y un Test de Habilidad. Para repartirnos el trabajo, yo en concreto me centré en el test de habilidad y el Test de Memoria.

Tras la reunión con algunos integrantes del departamento de neuropsicología y sabiendo los test que teníamos que desarrollar, empezamos con la búsqueda de frameworks para el desarrollo de la aplicación. Como se ha explicado anteriormente, nos decantamos por React Native.
Antes de empezar a desarrollar en este framework tuvimos que instalar este nuevo software junto con otras herramientas que nos servirían de ayuda para el desarrollo de nuestra APP, después de la instalación y de una pequeña inmersión en este nuevo framework empezamos con la creación de un nuevo proyecto junto con la creación de una estructura que luego subiríamos a Git-Hub para poder mantener un control de versiones sobre nuestro código.

Después de subir el código a Git-Hub una integrante del equipo y yo empezamos con la creación de las pantallas iniciales de nuestra APP, Splash Screen (Pantalla inicial), Login Screen (Pantalla de inicio de Sesión) y Register Screen (Pantalla de Registro). Tras comprobar que estas pantallas funcionaban correctamente a nivel visual, decidimos continuar con las preguntas generales, el test de orientación y el test de datos demográficos, que nos propusieron los doctores del Hospital Clínico San Carlos. Estas tareas nos la repartimos equitativamente una integrante del equipo y yo.

Tras terminar con el desarrollo de las primeras pantallas, continué con el desarrollo del Test de Habilidad, este test sirve para que el usuario pueda probar su destreza con el uso del dispositivo móvil, específicamente la destreza mínima que se requiere para poder utilizar nuestra APP correctamente.

Seguidamente continué con el desarrollo del Test de Memoria, para la creación de este test recibí un documento Word con todas las especificaciones necesarias para el desarrollo de este test. El Test de Memoria consiste en mostrar nueve palabras distintas, cada una de estas palabras pertenecen a tres departamentos distintos y cada departamento tiene un color diferente. Tras mostrarse las nueve palabras, el usuario deberá escribir todas las palabras que recuerde, seguido de las palabras pertenecientes al departamento de limpieza, al departamento de aseo, y al de alimentación. El test de memoria también cuenta con un test de demostración para que el usuario pueda realizar pruebas antes de comenzar con el Test de Memoria.

Tras finalización del Test de Memoria junto con su control de errores, y después que una compañera del equipo terminase con la creación del login y el registro de la API, continué con la conexión de nuestra APP con la API, para ello primero era necesario almacenar el token de inicio de sesión del usuario. El token es una cadena de caracteres único para cada usuario, nos sirve para identificar a los usuarios que realizan los tests. Para almacenar el token en el dispositivo móvil, tuve que descargar una dependencia de Expo llamada SecureStore, también tuve que descargar react-native-unimodules que me permitía modificar la configuración interna de Android, esta modificación se pide desde la documentación de SecureStore. Tras realizar todos los pasos de instalación de esta dependencia tuve que crear una nueva clase en nuestra APP para poder almacenar el token de una forma segura y encriptada.

Respecto al control de errores de la APP, se valoró que el dispositivo no tuviese auto rotación en el momento en el que el usuario este realizando los test, para ello tuve que modificar un fichero llamado Android Manifest. También se me asigno la tarea de recuperación de Contraseña, para ello tuve que añadir un nuevo controlador a nuestra API junto con un nuevo servicio de mensajería, para que el usuario pueda recibir un mensaje en su bandeja de correo electrónico cada vez que este solicite un cambio de contraseña. También tuve que añadir nuevas pantallas en la APP para que el usuario pueda solicitar el cambio de contraseña.

Después de comprobar que la API funcionaba correctamente en nuestros ordenadores, y después de una reunión con nuestro tutor, este nos sugirió subir nuestra API junto con la base de datos a Amazon Web Services. Tras esa reunión, me encargue de crear una nueva BBDD (RDS) y Máquina Virtual (EC2) en AWS.

Al igual que el resto de mis compañeros también ayude con la solución los errores y las dudas que iban surgiendo a medida que íbamos avanzado con el desarrollo de nuestro Proyecto.

Capítulo 4

Test y Evaluación de usabilidad

4.1. Test

En un producto software encontramos un atributo fundamental como es el de la usabilidad, que estudia la forma de diseñar y evalúa la calidad de un proyecto de la manera más cómoda, fácil e intuitiva para los usuarios que vayan a interactuar con dicho producto.

La evaluación de la usabilidad de un producto software permite entre otras cosas, descubrir errores internos o de diseño para que puedan ser corregidos con la mayor antelación posible.
En nuestro caso, la evaluación de usabilidad del proyecto se hizo en dos fases:

- **Mediante expertos:** en esta primera fase, se hizo una evaluación de la aplicación por personas competentes en el ámbito del producto. Este proceso engloba una serie de pruebas independientes, en primer lugar, por parte de los desarrolladores, seguidamente por parte del coordinador del proyecto y por último, por parte del neuropsicólogo del hospital. Nuestro objetivo, es detectar posibles errores y solucionarlos en primeras versiones funcionales, antes de obtener una versión final.

- **Mediante pruebas o test de usabilidad:** Una vez pasada la fase anterior y obtener la última versión funcional, decidimos realizar un test de usabilidad para poder obtener una evaluación objetiva y precisa desde la perspectiva de usuarios finales de una manera fácil.
  El objetivo con el que se realizó este test de usabilidad fue principalmente para medir:

  - Facilidad de uso.
  - Eficiencia.
  - Posibles errores.
  - Satisfacción de los usuarios.

  Este test está dirigido, en un principio, a pacientes que no presenten ningún tipo de enfermedad neurodegenerativa. Esto se debe a que es una versión que está aún en fase de prueba.

  Gracias a los resultados obtenidos del test de usabilidad y los errores reportados por los usuarios, se podrán hacer las mejoras pertinentes, con el fin de obtener un producto final

lo suficientemente robusto y útil para que finalmente pueda ser utilizado por personas con alguna enfermedad neurodegenerativa; siendo este el objetivo principal de nuestro proyecto.

4.2. Evaluación de la usabilidad

A continuación, explicaremos como se planteó la estrategia para la validación del diseño de nuestra aplicación, también como llevamos a cabo la evaluación de usuarios y el posterior análisis de los resultados.

4.2.1. Diseño de la estrategia de validación

Para el diseño de esta estrategia dividimos en dos fases independientes, mencionadas anteriormente, el proceso de validación. En la primera parte, nuestro equipo realizó una evaluación exhaustiva de las primeras versiones funcionales de la APP. Una vez solucionamos los errores encontrados, se pasó a la siguiente versión funcional al coordinador del proyecto y al neuropsicólogo para seguir testeando la APP y encontrar posibles puntos de mejora. Tras obtener la aceptación de la correcta funcionalidad de la herramienta por las distintas partes, se pasó a diseñar el test de usabilidad.

Para esta segunda parte debimos tener en cuenta que, para obtener unos mínimos resultados y poder realizar el análisis y buena comprensión de estos mismos, necesitábamos tener al menos un par de usuarios que realizasen la evaluación completa de la APP y realizasen también el test de usabilidad. Por lo tanto, el número mínimo establecido para nuestra evaluación se estipuló en dos usuarios.

Como está recogido en numerosos estudios de investigación de carácter médico, la correcta evaluación y posterior validación de aplicaciones de este tipo se deben realizar en supervisión de un neuropsicólogo, es por ello por lo que una vez obtenida nuestra versión funcional de la aplicación contactamos con Alfonso, neuropsicólogo del departamento de neurología del Hospital Clínico San Carlos, que había estado participando con nosotros desde el principio del proyecto. Gracias a su colaboración se realizó la prueba de evaluación con tres usuarios distintos.

La idea de nuestra estrategia de evaluación era realizar un formulario de nueve preguntas para valorar el uso de la aplicación. Este formulario debía ser rellenado por el usuario una vez finalizase el ciclo completo de preguntas y pruebas que administra nuestra aplicación. Esto significa que los usuarios que fuesen a probar la APP debían realizar un uso completo de la misma, es decir primero registrarse, entrar con su usuario y luego iniciar la secuencia completa de preguntas, pruebas de destreza y los cuatros test de evaluación cognitiva adaptados. Todo esto bajo supervisión del neuropsicólogo.

Para recoger las respuestas a todas las preguntas, consideramos la idea de realizarlo a través de Google Forms, y escogimos esta forma finalmente porque nos pareció bastante útil las gráficas de los resultados que aporta este servicio y que su uso sencillo facilitaría al usuario hacerlo más cómodamente.

Por último, una vez recogidos los resultados de todas las evaluaciones hechas, debíamos analizar estos resultados. Por ello, en el siguiente apartado veremos en concreto con que intención se realizó cada pregunta y como sacamos las principales conclusiones del formulario.

4.2.2. Formulario de evaluación para usuarios y resultados

Realizamos el test de usabilidad propuesto por el equipo de desarrollo a seis personas. En la siguiente tabla se muestran sus principales datos socio demográficos y el sistema operativo en el que cada uno realizó la prueba:

USUARIO	EDAD	GÉNERO	NIVEL DE ESTUDIOS	SISTEMA OPERATIVO
Usuario I	24	Femenino	Grado universitario	iOS
Usuario II	29	Femenino	Grado universitario	iOS
Usuario III	31	Masculino	Doctorado	Android
Usuario IV	37	Masculino	Formación profesional	iOS
Usuario V	45	Masculino	Secundaria	Android
Usuario VI	52	Femenino	Secundaria	Android

Cuadro 4.1: Tabla de usuarios

Mediante un formulario de Google Forms, se han obtenido una serie de gráficas de cada una de las preguntas pertenecientes al formulario.
Este formulario se divide en cuatro bloques, que son los siguientes:

1. El primer bloque está formado por las preguntas 1, 2, 3 y 4 que mostraremos a continuación. Estas preguntas se diseñaron específicamente para averiguar la facilidad de uso de nuestra aplicación y a su vez, evaluar si el usuario se encontró con algún tipo de dificultad que le impidiese utilizar de forma óptima la aplicación.

    - **Pregunta 1:** Como se puede observar en esta primera gráfica correspondiente a la primera pregunta del formulario, las seis personas a las que se les ha realizado dicho test y han utilizado la aplicación, consideran que es bastante fácil de usar.

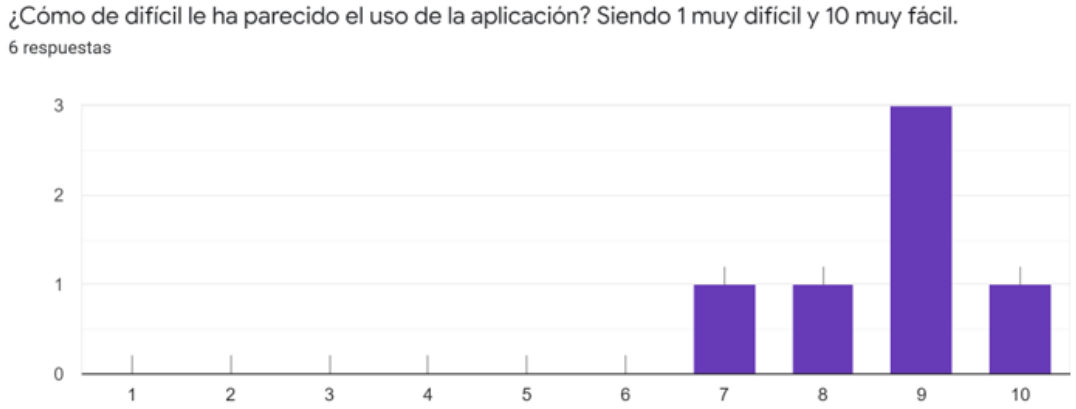


Figura 4.1: Pregunta 1 del test de usabilidad.

- **Pregunta 2:** De la siguiente gráfica referente a la pregunta número 2, podemos concluir que las demostraciones presentes al comienzo de cada uno de los test sirven de ayuda a los usuarios de la APP, para luego poder realizarlos correctamente.

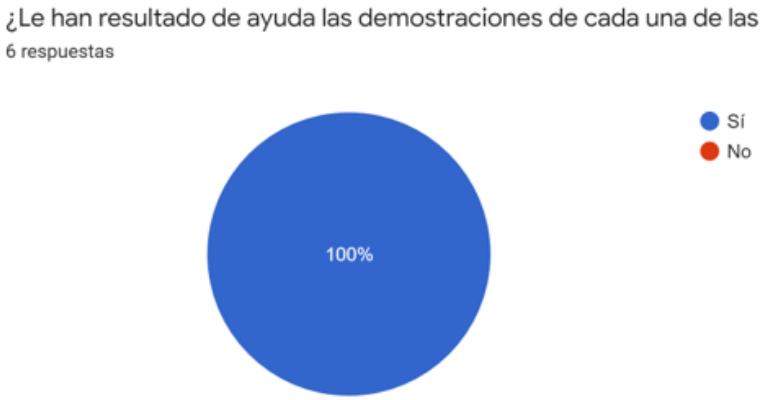


Figura 4.2: Pregunta 2 del test de usabilidad.

- **Pregunta 3:** La gráfica correspondiente a la tercera pregunta nos informa de que las explicaciones de los test de la APP son lo suficientemente claras para los usuarios.

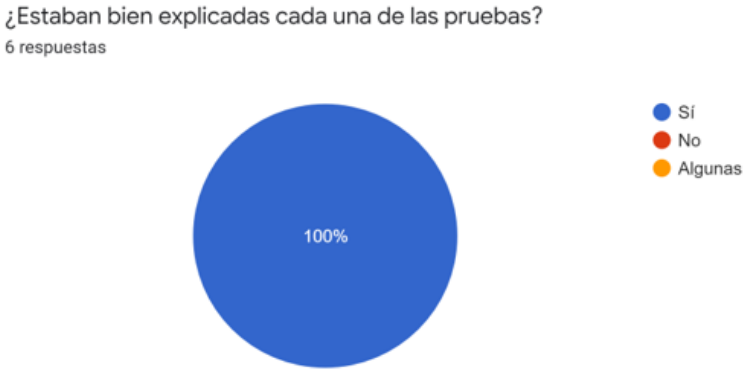


Figura 4.3: Pregunta 3 del test de usabilidad.

- **Pregunta 4:** La respuesta a la pregunta número 4 nos indica que para la mayoría de usuarios que han probado la APP la prueba más difícil de realizar ha sido el Test de Memoria.

Figura 4.4: Pregunta 4 del test de usabilidad.

2. Este siguiente bloque formado por las cuestiones 5 y 6, evalúan cómo era el entorno en el que se encontraba el paciente cuando realizó la prueba. Estas preguntas las consideramos de suma relevancia debido a que, si el usuario contaba con algún tipo de distracción en el momento de hacer la prueba, esto podría influir en que percibiese alguna parte de la ejecución con dificultad y que no se debiese en concreto por la funcionalidad y diseño de la APP.

    - **Preguntas 5 y 6**: Las respuestas correspondientes a las preguntas 5 y 6 del test de usabilidad sirven de ayuda a la hora de analizar las respuestas de los test realizados por cada uno de los usuarios en la BBDD, ya que en este caso, al no haber tenido ningún tipo de distracción no deberían existir gran número de fallos en cada uno de los test.

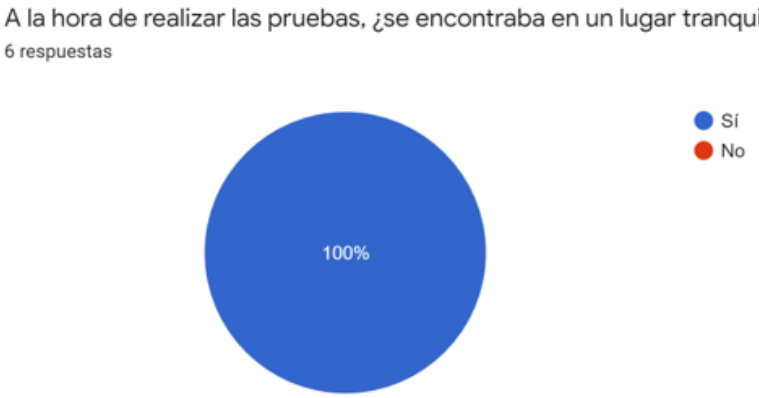


Figura 4.5: Pregunta 5 del test de usabilidad.

Figura 4.6: Pregunta 6 del test de usabilidad.

3. El tercer bloque formado solo por la pregunta 7, está indicado para evaluar si el tiempo que damos a cada una de las pruebas es suficiente para el usuario.

    - **Pregunta 7:** Cada uno de los usuarios han respondido que el tiempo dado para cada una de las pruebas es el adecuado.

Figura 4.7: Pregunta 7 del test de usabilidad.

4. El último bloque de preguntas se centra en evaluar la satisfacción general del paciente con respecto a nuestra aplicación, dándole la opción de añadir algún tipo de comentario y obtener aspectos concretos que podamos valorar y mejorar en un futuro.

- **Pregunta 8:** La satisfacción de los usuarios con la APP es bastante favorable.

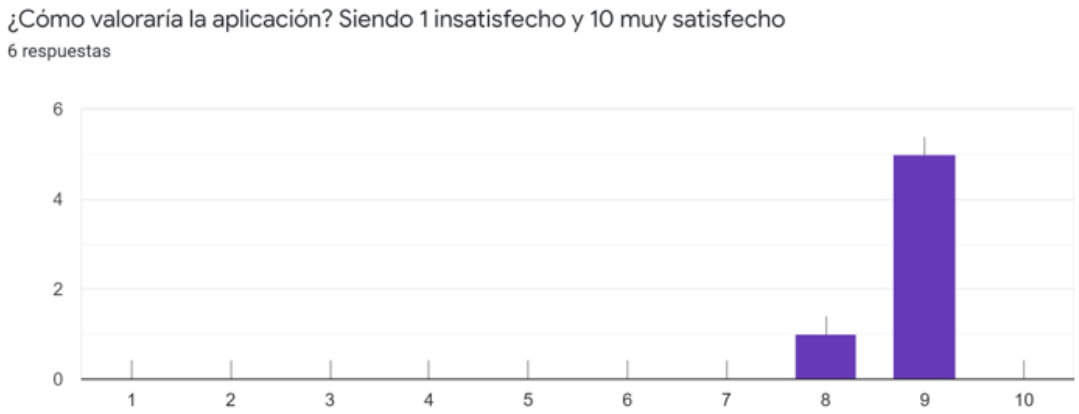


Figura 4.8: Pregunta 8 del test de usabilidad.

- **Pregunta 9:** Finalmente en el test de usabilidad añadimos una pregunta en la que los usuarios podían aportar la opinión que considerasen sobre la APP. De esta pregunta obtuvimos una única valoración, en la cual, tres de los usuarios coincidían y consideraban que el tiempo de espera en cada una de las pantallas correspondientes al Test de Denominación del Lenguaje era demasiado. A lo que propusieron o añadir un botón que permitiese avanzar o bien disminuir dicho tiempo de espera. Aunque esta valoración se recoge con la intención de mejorar aspectos a futuro, en concreto, este tiempo está estipulado a 20 segundos por criterio médico; puesto que la APP está principalmente dirigida a personas con enfermedades neurodegenerativas, este tiempo podría resultar más adecuado para ellas.

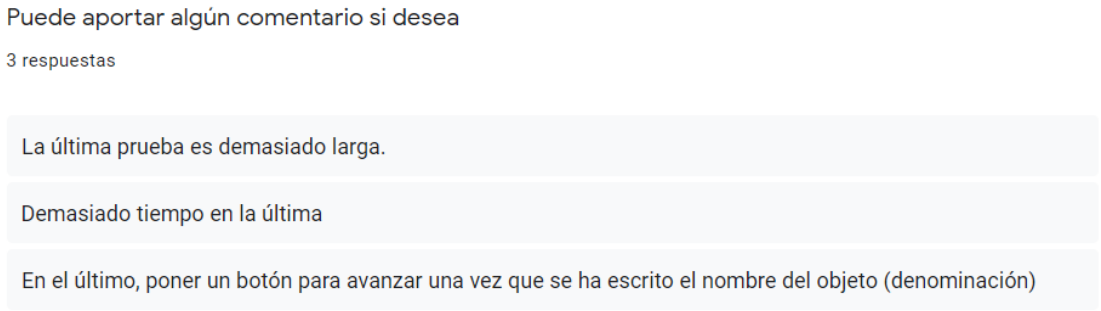


Figura 4.9: Pregunta 9 del test de usabilidad.

Finalmente, tras realizar el proceso completo de validación observamos una correcta aceptación de la herramienta por parte de todos los usuarios, incluido el neuropsicólogo que ha estado presente durante la administración de los test.

Capítulo 5

Conclusiones

La principal conclusión obtenida tras el estudio llevado a cabo a lo largo de este Trabajo de Fin de Grado es que hemos cumplido con el primer objetivo marcado al inicio del proyecto. Siendo este el desarrollo e implementación de una aplicación de autoadministración de test cognitivos para dispositivos móviles. Aunque esta no es la versión final, se ha conseguido realizar el despliegue de una versión funcional en varios dispositivos móviles Android e iOS.

Por otro lado, observamos que la lógica final aplicada permite la recopilación y procesamiento de las respuestas de los usuarios a cada uno de los test, proporcionando así una estructura de tipo almacén de datos, que deja todos los resultados del paciente de forma estructurada y filtrada. Además, estos resultados se han procesado con el propósito concreto de facilitar al especialista el número de aciertos y fallos obtenidos en cada una de las pruebas.

Finalmente, gracias a la colaboración de Alfonso Delgado, neuropsicólogo del Hospital Clínico San Carlos, se han podido realizar las evaluaciones de validación necesarias para realizar un análisis sobre la usabilidad de nuestra aplicación.
Del estudio posterior de estos resultados podemos concluir que nuestra aplicación cumple con los principales objetivos propios de cualquier producto software, que es, ser fácil de usar e intuitiva, además de que la satisfacción general de los usuarios ha sido claramente positiva.

Capítulo 6

Planificación a futuro

Tras todos los resultados obtenidos con el desarrollo de la APP y las conclusiones mencionadas en el apartado anterior, cabe indicar que todavía queda parte de desarrollo a continuar para ir complementando y perfeccionado esta aplicación. De esta forma podríamos obtener una herramienta de autoadministración de test cognitivos que se pueda implantar dentro de departamentos de neurología y pudiera ser utilizado por recomendación de especialistas para el uso en casa por parte de los usuarios.

Como hemos mencionado en la sección 4 de esta memoria, desarrollamos una técnica de evaluación a través del diseño de un test de usabilidad. Este test nos sirvió para comprender los aspectos de la APP que debemos mejorar para facilitar la interacción entre la interfaz de usuario y el futuro paciente. Estos aspectos para mejorar que hemos obtenido como resultado de una primera tanda de evaluación de usuarios, deberán ser los primeros en modificarse y adaptarse para seguir desarrollando la APP de cara a una nueva versión.

En segundo lugar, y una vez realizados las modificaciones oportunas, deberíamos ampliar el tamaño muestral de usuarios para evaluar la APP y realizar un análisis estadístico de los resultados obtenidos. Este estudio debería realizarse en población sana (como hemos hecho primero, evaluando a los acompañantes de pacientes que acuden al departamento de neurología) para ayudar a la normalización de los datos.

Como hemos explicado en esta memoria, en cada respuesta de cada test se evalúa al usuario con acierto o fallo, y nos quedamos con el total de éstos al final de que se realicen todos los test. Normalizar, implica que debemos analizar cual es la puntuación media obtenida por parte de todos los usuarios evaluados para luego poder interpretar a partir de esos resultados cuáles serían los obtenidos por una persona con alguna enfermedad neurodegenerativa presente.

Este análisis no solo implicaría analizar los resultados de los test si no también analizar las respuestas a las preguntas generales y de orientación que se realizaron previamente al inicio de los test.

Una vez realizada esta fase de estudio se podría aplicar esta misma evaluación a otro tipo de población como son los usuarios con enfermedad neurodegenerativa ya diagnosticada. Con estos resultados se podrían comparar los obtenidos previamente en la fase anterior.

Para dar otro paso más adelante se podría adaptar algún otro tipo de test cognitivo que mida otro tipo de funcionalidad, incluyendo técnicas de reconocimiento de voz y estímulos auditivos, para intentar realizar una evaluación cognitiva más completa al paciente. De esta manera se podría ir adaptando en medida de las necesidades que vayan surgiendo, cambios o adaptaciones de otros posibles test que se realizan de manera presencial y supervisada a través de la aplicación.

Por último, podríamos implementar y reconfigurar el diseño de alguno de nuestros test como es el de localización de punto o el test de fluencia de diseño, para adaptarlo a dispositivos tablets. Actualmente nuestra aplicación está configurada para dispositivos smartphone, pues estos dispositivos son los más habituales y de los que en mayor probabilidad dispone una persona. Pero para aquellas personas que dispongan además de este tipo de dispositivo o tableta (o bien dispongan sólo de este tipo), implementar una versión adaptando ciertos parámetros de configuración que permitan el uso en tablets facilitará la posibilidad de realizar los test a través de nuestra aplicación y ampliará aún más el número de personas con posibilidad de realizarlas desde su propio dispositivo.

Apéndice A

Introduction

The increase in human life expectancy, combined with demographic growth, has led to the emergence of a greater number of cases of neurodegenerative diseases. Among other things, this increase has led to a delay in obtaining a diagnosis due to a lack of medical care. Therefore, the intention of this work is to help, as far as possible, to mitigate this problem.

A.1. Neurodegenerative diseases

Neurodegenerative diseases form an extensive part in the field of medicine, specifically in the field of neurology. Moreover, they are commonly characterised as diseases of unknown origin, with progressive development of their symptoms, which, in turn, show the gradual disintegration of one or more parts of the human being's nervous system. In addition to these characteristics, there is also a decrease in functionality and personal independence.
People suffering from one of these diseases require full and continuous care, which justifies the need for patients to be treated in specific units. [1]

These diseases can be classified depending on the clinical manifestations that the patient presents; amongs them we can differentiate those that mainly show a dementia syndrome, such as Alzheimer's; Those that are exposed with movement and posture disorders, in the case of Parkinson's; those that present progressive ataxia, i.e. lack of muscle control or coordination of voluntary movements, such as eating, in the case of Cerebellar Ataxia; or those that show muscle weakness and atrophy, for example Amyotrophic Lateral Sclerosis; and many other diseases with different symptoms.

All these neurodegenerative diseases do not have a treatment that acts on the cause that originates it, but rather palliative treatments are applied, that is to say, they do not find a cure. [2]

A.2. Why this TFG is proposed

The early detection and diagnosis of neurodegenerative diseases, according to different international studies, is currently considered a significant public health problem still unresolved, as they

reflect an average delay of 8 to 32 months, between the appearance of the first symptoms and the diagnosis of the disease itself.

An early diagnosis would, among many other things, increase understanding of the problem and provide access to treatment and social support. However, to date, no country has proposed systematic screening of patients suffering from this type of disease in primary care, despite the fact that the percentage of undiagnosed cases is very high. [3]

The proposal of this final degree project, in the current context, aims to help with the problems described, that is, to allow cognitive tests to be carried out not only and exclusively in a medical centre, whether it is a health centre or a hospital. This would also provide a solution to the current delay in obtaining the diagnosis of the disease, which would also accelerate the pressure of healthcare.

A.3. Objectives of this TFG

The main objective of this Final Degree Project is the implementation and development of a hybrid mobile application, that is to say, available for devices with Android or iOS operating systems, which includes a series of cognitive assessment tests.

To this end, this work was done in collaboration with the Neurology Department of the Hospital Clínico San Carlos, who showed us the different tests they worked with and provided us with their adaptation for mobile devices. Thus, the answers of each of the tests performed on the patients are stored; and, in turn, scores are obtained from them, which will later be studied by these specialists. These scores will provide the neurologist with an estimate when making the patient's diagnosis. Likewise, another objective of this application is to allow the self-administration of the tests to the patients in a comfortable and simple way.

On the other hand, it is worth noting the challenge involved in carrying out a group software project from the outset, as this involved detailed planning of all its phases, from the study to its implementation and testing. This made it necessary to organise the entire team, plan in detail the time and resources available and put into practice the knowledge acquired throughout the years of the course.

A.4. Structure of the document

This document is structured in a series of chapters, the contents of which are described below:

- **Introduction:** describes the problem explained in the previous section, as well as the main objective for which this work is carried out.
- **State of art:** contains a detailed description of the research and publications carried out in the field of our work, i.e. within the field of neurocognitive diseases and the tools currently used for carrying out cognitive tests.

- **Methodology:** describes the final result of the application. At the same time, the technologies and tools used for the development of our application are explained in detail, as well as the steps and organisation of the team when carrying out the work.
- **Usability test and evaluation:** presents a usability form completed by users after using the APP. As well as a compilation of the different responses obtained.
- **Conclusions and future lines:** this chapter presents the conclusions of the work carried out, as well as the possible improvements that could be made to the application after it has been presented.

Apéndice B

Conclusions

The main conclusion obtained after the study carried out throughout this final project degree is that we have achieved the first goal set at the beginning of the project. This mentioned goal consists of the development and implementation of an application, which itself provides cognitive tests for mobile devices. Although this is not the final version, it has been possible to carry out the display of useful version on several Android and iOS mobile devices.

On the other hand, we observe that the applied final logic allows the compilation and processing of the responses of the users to every test. In this way, a type of data base is provided, which supplies every results of the patients in a structured and filtered way. In addition, these results have been processed with the specific purpose of providing the specialist with the number of successes and failures obtained in each of the tests.

To sum up, thanks to the collaboration of Alfonso Delgado, neuropsychologist of the Clínico San Carlos Hospital, it has been possible to carry out the necessary validation evaluations to analyze the use of our application. From the subsequent study of these results, we can conclude that our application achieves the main purpose of any software product, which is to be easy to use and intuitive. In addition, we can also say that it has been registered a general satisfaction of users in a clearly positive way.

Bibliografía

[1] Abril Carreres, M.A., N. Ticó Falguera y R. Garreta Figuera: *Enfermedades neurodegenerativas*. Rehabilitación, 38(6):318–324, 2004, ISSN 0048-7120.

[2] Pipaón, I Sáenz de y R Larumbe: *Programa de enfermedades neurodegenerativas*. En *Anales del Sistema Sanitario de Navarra*, volumen 24, páginas 49–76, 2001.

[3] Villarejo, A. y V. Puertas-Martín: *Utilidad de los test breves en el cribado de demencia*. Neurología, 26(7):425–433, 2011, ISSN 0213-4853.

[4] Montenegro Peña, Mercedes, Pedro Montejo Carrasco, Marcos Llanero Luque y Ana Isabel Reinoso García: *Evaluación y diagnóstico del deterioro cognitivo leve*. Revista de Logopedia, Foniatría y Audiología, 32(2):47–56, 2012, ISSN 0214-4603.

[5] Rey, A: *Test de la Figura Compleja de Rey*. TEA Edificio, Madrid, 1980.

[6] Quental, Natália Bezerra Mota, Sonia Maria Dozzi Brucki y Orlando Francisco Amodeo Bueno: *Visuospatial Function in Early Alzheimer's Disease—The Use of the Visual Object and Space Perception (VOSP) Battery*. PLOS ONE, 8(7):1–7, Julio 2013.

[7] Calvo, L., M. Casals-Coll, G. Sánchez-Benavides, M. Quintana, R.M. Manero, T. Rognoni, R. Palomo, F. Aranciva, F. Tamayo y J. Peña-Casanova: *Estudios normativos españoles en población adulta joven (proyecto NEURONORMA jóvenes): normas para las pruebas Visual Object and Space Perception Battery y Judgment of Line Orientation*. Neurología, 28(3):153–159, 2013, ISSN 0213-4853. https://www.sciencedirect.com/science/article/pii/S0213485312000758.

[8] Zakzanis, Konstantine K., Richard Mraz y Simon J. Graham: *An fMRI study of the Trail Making Test*. Neuropsychologia, 43(13):1878–1886, 2005, ISSN 0028-3932.

[9] Zimmerman, Molly E., Mindy J. Katz, Cuiling Wang, Leah C. Burns, Robert M. Berman, Carol A. Derby, Gilbert LÍtalien, David Budd y Richard B. Lipton: *Comparison of "Word" vs. "Picture" version of the Free and Cued Selective Reminding Test (FCSRT) in older adults*. Alzheimer's Dementia: Diagnosis, Assessment Disease Monitoring, 1(1):94–100, 2015, ISSN 2352-8729.

[10] Watanabe, Kiyoko, Tatsuya Ogino, Kousuke Nakano, Junri Hattori, Yoko Kado, Satoshi Sanada y Yoko Ohtsuka: *The Rey–Osterrieth Complex Figure as a measure of executive function in childhood*. Brain and Development, 27(8):564–569, 2005, ISSN 0387-7604.

[11] *NeuroNation*. https://www.neuronation.com/.

[12] *CogniFit*. https://www.cognifit.com/es.

[13] *Imentia: La app de estimulación cognitiva*. https://www.imentia.com/.

[14] *Unico*. https://unicostudio.co/.

[15] *CogniFit*. https://www.cognifit.com/es.

[16] *Neotiv*. https://neotiv.com/en.

[17] *App*. https://www.mocatest.org/app/.

[18] Valladares Rodriguez, Sonia Maria *y cols.*: *Detección precoz del deterioro cognitivo mediante técnicas de gamificación, aprendizaje máquina y herramientas TIC*. Tesis de Doctorado, Enxeñaría telemática, 2019.

[19] Martínez, Rocío Erandi Barrientos, Nicandro Cruz Ramírez, Héctor Gabriel Acosta Mesa, Ivonne Rabatte Suárez, María del Carmen Gogeascoechea Trejo, Patricia Pavón León y Sobeida L Blázquez Morales: *Árboles de decisión como herramienta en el diagnóstico médico*. Revista médica de la Universidad Veracruzana, 9(2):19–24, 2009.

[20] Danielsson, William: *React Native application development: A comparison between native Android and React Native*, 2016.

[21] Hagos, Ted: *Android Studio*, páginas 5–17. Apress, Berkeley, CA, 2018, ISBN 978-1-4842-3156-2.

[22] Chen, Xianjun, Zhoupeng Ji, Yu Fan y Yongsong Zhan: *Restful API Architecture Based on Laravel Framework*. Journal of Physics: Conference Series, 910:012016, oct 2017.

[23] *Laravel - The PHP Framework For Web Artisans*. https://laravel.com/.

[24] *Postman | The Collaboration Platform for API Development*. https://www.postman.com/.

[25] Cloud, Amazon Elastic Compute: *Amazon web services*. Retrieved November, 9(2011):2011, 2011.

[26] *IDE de , editor de código, Azure DevOps y App Center*. https://visualstudio.microsoft.com/es/.

[27] Jhoselin, Oscco Carbajal, Andrea Janampa Zanambria, Arantxa Brock Rabines y Jaime Palazon Othon: *BrainApp*. https://github.com/jhosstich/BrainApp.

[28] Arantxa, Brock Rabines, Andrea Janampa Zanabria, Jhoselin Oscco Carbajal y Jaime Palazón Othon: *BrainApp Api*. https://github.com/Arantxa1209/ApiTFG.

[29] Maida, Esteban Gabriel y Julián Pacienzia: *Metodologías de desarrollo de software*. 2015.

[30] Huddleston, Rob: *Introduction to Adobe Experience Design*, páginas 7–21. Apress, Berkeley, CA, 2017, ISBN 978-1-4842-2964-4.

Índice de figuras

Índice de cuadros

Agradecimientos

Primeramente, agradecer a cada una de nuestras familias, en especial a nuestros padres y hermanos, por confiar y creer en nosotros, no sólo en nuestros años de carrera universitaria, sino en cada uno de nuestros pasos en la vida.
Gracias a nuestros amigos por apoyarnos y darnos fuerzas para continuar cuando se presentaban dificultades en nuestro camino.

Agradecer también a todas aquellas personas con las que hemos trabajado durante los años de nuestra carrera, tanto compañeros como los diferentes docentes que hemos tenido a lo largo de estos años.
En especial, agradecer al tutor de este proyecto José Luis Ayala Rodrigo, que nos ha orientado y ayudado en todo momento desde los inicios de este trabajo, mostrándose disponible siempre que ha podido y lo hemos requerido. Atendiendo a cada una de nuestras necesidades y problemas, aportándonos posibles soluciones para que este trabajo se realice con éxito.

Por último agradecer a Vanesa Pytel Córdoba y Jordi Matías-Guiu Antem, neurólogos y a Alfonso Delgado Álverez, neuropsicólogo del Hospital Clínico San Carlos, que también nos han ayudado en la realización de este trabajo, aclarándonos cualquier duda y mostrando interés por la fase de desarrollo del mismo.