Diseño e implementación de una aplicación móvil para poder conectarse a Caronte y subir pdfs de exámenes previa validación como estudiante y autorización del profesor

Joan Sánchez Úbeda

Resumen— Actualmente, el sistema de recogida, corrección y revisión de exámenes presenta algunos problemas tanto a profesores como a alumnos. Entre estos se encuentran la recogida de estas actividades de forma segura o la presencialidad en las revisiones. Como solución a estos problemas, se ha planteado el desarrollo de un sistema a través de una aplicación externa a Moodle. Este proyecto tiene como objetivo crear una aplicación Android que permita a los alumnos subir los exámenes finalizados a una plataforma Moodle con reconocimiento biométrico para evitar entregas fraudulentas. Además, una vez estas actividades estén subidas, el profesor podrá corregirlas, puntuarlas y generar un archivo que contenga las notas de cada alumno y, finalmente, estos podrán visualizar las correcciones y tener feedback de las mismas. A lo largo de este artículo se explicarán las diferentes decisiones tomadas para el desarrollo tanto de la aplicación móvil como para el módulo de Moodle**.**

Palabras clave— Caronte, Moodle, Android, Reconocimiento biométrico, OCR.

---

1. Introducción

El sistema actual por el que se rige el proceso de realizar un examen y su corrección se basa en que el alumno lo realice en papel, lo entregue al profesor responsable y este se encargue de corregirlo y puntuarlo. Este procedimiento presenta varias problemáticas entre las que se encuentra el asegurarse que ningún estudiante tenga acceso a los exámenes de otros, el manejo de cientos de exámenes, o el obligar a los alumnos a hacer consultas presenciales en caso de querer revisar sus propios exámenes. Como propuesta de solución de estos problemas se ha presentado el proyecto realizado en este trabajo.

La finalidad de este proyecto es añadir una funcionalidad a Moodle [1] que permita a los alumnos subir un archivo con el contenido de un examen con la autorización del profesor para que éste, más tarde, pueda revisarlos y corregirlos, pudiendo editarlos desde el mismo Moodle. Una vez estén corregidos, Moodle detectará automáticamente la nota puesta por el profesor, la recogerá con un OCR [2] (Reconocimiento Óptico de Caracteres) y la asignará al alumno en cuestión. Además, también se implementará la posibilidad de descargar el conjunto de todas las notas del alumnado en informes en formato Excel por el profesor. Finalmente, el estudiante será capaz de revisar las correcciones

realizadas por parte del docente sin necesidad de desplazarse hasta la universidad.

Este trabajo forma parte de un proyecto más amplio que consiste en, por una parte, el diseño e implementación de una aplicación móvil para poder conectarse a Moodle y subir exámenes previa validación como estudiante y autorización del profesor y, por otra parte, el diseño e implementación de un nuevo módulo de Moodle para recoger y corregir estas actividades además de presentar las notas y realizar revisiones de estas. Este trabajo estará dividido en dos TFGs distintos, de los cuales me centraré en el que desarrolla la parte de ingeniería del software. Sin embargo, el diseño, objetivos, planificación así como las características generales del mismo han sido pensadas en conjunto entre los dos alumnos encargados de ambas partes con la ayuda de la tutora a cargo.

En la sección 2 de este documento expondré el estado del arte. La sección 3 está dedicada a los objetivos de este proyecto. A continuación, en el apartado 4, presentaré la metodología de trabajo que seguiré, además de como se pretende comunicar con la otra parte de este TFG, desarrollada por mi compañero Víctor Aguilar Martínez. En el apartado 5 expondré la planificación pensada para este trabajo. La sexta sección será el desarrollo de cada parte del proyecto. En la sección 7 se presentarán los resultados obtenidos. Finalmente, en el apartado 8, expondré unas conclusiones sobre el TFG y propondré una serie de mejoras o extensiones al mismo que se podrían incluir a futuro.

2. Estado del arte

Hasta donde nosotros hemos investigado, no existe una aplicación o sistema que conecte con Moodle y cubra las necesidades que tenemos.

Una de las primeras acciones que se deben realizar cuando se comienza un proyecto es la preparación del entorno donde éste se desarrollará. Para este trabajo es necesario tener disponibles varios recursos como un servidor Apache [3], un gestor de base de datos y un intérprete de PHP [4]. Esta es la base de Moodle y, por ello, se ha elegido el programa XAMPP [5], un paquete de software libre que contiene prácticamente todo lo que necesitamos. Sin embargo durante el montaje de un moodle en un servidor local encontramos un problema, la última versión estable de PHP que XAMPP nos proporcionaba no soporta la versión de moodle elegida. Debido a esto y después de una pequeña investigación se decidió instalar una versión más antigua de XAMPP que si soportase Moodle 3.10.

Al encontrarnos con la anterior problemática comenzamos a pensar en cómo interaccionan las diferentes herramientas y recursos que utilizaremos durante el trabajo. De este modo, llegamos a pensar para qué versión de android crearemos la aplicación móvil. Nuestra principal preocupación consistía en que si los resultados obtenidos, en el futuro, se aplicaban a Caronte, debían estar preparados para adecuarse a las tecnologías de ese momento y por lo tanto, no podemos escoger trabajar con tecnologías que en el momento de la producción fueran relativamente antiguas. Por otro lado, trabajar con susodichas tecnologías novedosas puede dificultar el probar el trabajo realizado.

Para poder satisfacer uno de los objetivos del trabajo, es necesario utilizar una herramienta externa que permita abrir, editar y guardar los archivos que los alumnos entreguen. En este caso, Google Drive proporciona una solución muy potente y fácil de aplicar que utiliza Google Cloud Services [6], sin embargo, esta posibilidad requiere de una inversión económica inicial, debido a esto, esta opción queda descartada. Teniendo esto en cuenta, se ha escogido Unoconv [7] como conversor de archivos gratuito.

3. Objetivos

A continuación expondré los objetivos principales del trabajo y explicaré el cómo se pretende cumplirlos. Estos objetivos están clasificados dependiendo de su importancia para el desarrollo del proyecto.

Existen tres niveles de prioridad: alta, media y baja. Aquellos objetivos definidos con un nivel de prioridad alta son los puntos claves del proyecto, los pilares en los que se sustenta el correcto funcionamiento de cada parte de este. Por otro lado, un nivel medio describe aquellos elementos que son un complemento o una funcionalidad de la que Moodle no dispone pero sin la cual el proyecto puede seguir adelante a costa de cambiar la idea original del producto. Finalmente, los objetivos de nivel de prioridad bajo son funcionalidades que facilitan el trabajo a los usuarios y sin las cuales cada parte del proyecto seguiría adelante sin problemas aunque fuese un producto de menor calidad.

1. Creación de un módulo de Moodle que pueda comunicarse con una aplicación Android. (Alta prioridad)

Para completar este primer objetivo se requiere una investigación sobre cómo crear un Moodle propio y, además, cómo crear también un nuevo módulo en este.

2. Creación del esqueleto de la aplicación móvil. (Alta prioridad)

El segundo objetivo consiste en la creación de la base de la aplicación que permitirá la autentificación de los alumnos y hacer fotos de los exámenes.

3. Implementación del proceso de autentificación y validación de la aplicación. (Alta prioridad)

Este objetivo consiste en el desarrollo de un método de autentificación que identifique al alumno para poder acceder a la entrega del examen. Además, crearé un método de validación en Moodle que le permita a los profesores aceptar la entrega que los alumnos están intentando realizar desde la aplicación.

4. Implementación de un proceso para enviar los archivos. (Alta prioridad)

El siguiente objetivo consiste en el desarrollo de los métodos que permitan tomar todas las imágenes necesarias para mostrar el examen, convertirlas a formato PDF y enviarlas a Moodle.

5. Crear una herramienta que permita al profesor corregir el examen desde Moodle. (Prioridad media)

Con este objetivo se pretende desarrollar una herramienta para el profesor que le permita corregir el examen desde Moodle sin necesidad de descargarse ningún archivo usando las herramientas Unoconv y Ghostscript [8]

6. Que Moodle pueda leer automàticamente del PDF del examen las notas de cada pregunta y genere la nota final del examen. (Baja prioridad)

Una vez superada la meta anterior, se pretende usar un OCR para detectar las notas en los exámenes y, automáticamente, generar los resultados de estos.

7. Que Moodle genere un archivo en formato Excel con todas las notas de los estudiantes. (Baja prioridad)

Finalmente se incluirá la posibilidad por parte de los profesores de generar un archivo formato Excel con las notas de todos los alumnos.

4. Metodología

Debido a la complejidad del trabajo propuesto y para poder coordinar lo más eficientemente posible los esfuerzos de ambos estudiantes que estarán participando en este trabajo, hemos utilizado Scrum [9]. Scrum es una metodología ágil que dicta un conjunto de buenas prácticas para trabajar en equipo y obtener resultados satisfactorios en los proyectos. Al usar esta metodología, realizaré entregas rutinarias del producto, lo cual me permitirá observar el crecimiento de este, su avance a lo largo del tiempo y, en caso de que en este proceso haya algún error o se requiera algún cambio importante, se podrá realizar minimizando riesgos. Las entregas mencionadas serán seis, y coincidirán con el final de cada uno de los sprints explicados más adelante en la planificación del TFG.

La comunicación entre ambas partes del TFG es crucial para un desarrollo sano del mismo, por eso usamos Microsoft Teams, herramienta que nos proporciona la universidad, para realizar las reuniones tanto con mi compañero como con la tutora del trabajo. A lo largo del transcurso de todo el proyecto, mantendré reuniones semanales en las que expondré los avances realizados, problemas encontrados y dudas que surjan durante el desarrollo. Además, también participaré en reuniones donde establezca, junto a mi compañero, las bases del trabajo de ambos y donde daremos nuestra opinión de hacia dónde dirigir el futuro del proyecto.

5. Planificación

A la hora de planificar el desarrollo de este trabajo he tenido en cuenta las diferentes fechas importantes relacionadas con las entregas de los diferentes informes de seguimiento del TFG. Dicho esto, y teniendo en cuenta que utilizamos la metodología Scrum, se han propuesto los sprints que podemos observar en la Tabla 1.

Los primeros pasos realizados según esta planificación son las investigaciones previas al primer sprint y que consisten principalmente en la búsqueda y preparación de las herramientas y entornos de trabajo necesarios para el desarrollo del proyecto. Una parte de este trabajo de investigación consistió en encontrar y solucionar los problemas de compatibilidad entre la versión de Moodle que se emplea en este trabajo y la versión de XAMPP. También me encargué de buscar una alternativa gratuita al sistema de conversión de documentos instalado por defecto en Moodle que proporciona Google.

Núm. Sprint	Duración	Objetivos
Investigación de entornos	19 febrero - 14 marzo	Investigación de herramientas que se adecuen a nuestras necesidades (servidor web, base de datos e intérprete de PHP). Investigar y elegir el entorno de trabajo sobre el que se desarrollará la aplicación móvil.
1	15 marzo - 4 abril	1. Creación de un módulo de Moodle que pueda comunicarse con una aplicación. 2. Creación del esqueleto de la aplicación móvil.
2	5 abril - 25 abril	3. Implementación del proceso de autentificación y validación de la aplicación.
3	26 abril - 16 mayo	4. Implementación de un proceso para enviar los archivos. 5. Crear una herramienta que permita al profesor corregir el examen desde Moodle.
4	17 mayo - 30 mayo	6. Que Moodle pueda leer automàticamente el PDF del examen las notas de cada pregunta y genere la nota final del examen. 7. Que Moodle genere un archivo en formato Excel con todas las notas de los estudiantes.
5	31 mayo - 20 junio	Resolución de posibles fallos y puesta a punto de todo el trabajo.
6	21 junio - 28 junio	Preparación de las entregas finales y la defensa del TFG ante el tribunal

Tabla 1. Separación de los distintos objetivos a lo largo de cada sprint.

Durante este período de tiempo anterior al comienzo del desarrollo tanto de la aplicación móvil como del módulo de Moodle también se ha pedido a la universidad un servidor remoto en el que se recogerán las diferentes entregas y que, en última instancia, contendrá los resultados de este TFG. Este servidor deberá ser capaz de soportar el Moodle a desarrollar y, para ello, he seguido las recomendaciones de aquel que me lo proporcionará. Esto significa que el servidor contará con 4 cores, 4 GB de RAM y 20 GB de HDD y contará con la

última versión de Linux que soporte las diferentes herramientas y recursos comentados a lo largo de este informe.

Hace falta añadir también que completar un objetivo incluye realizar su correspondiente testeo. Esto incluye tanto a la creación de tests unitarios como la realización de *exploratory testing*[1].

Debido a que este proyecto está dividido en dos TFGs distintos, es prácticamente imposible que no haya momentos donde el desarrollo de una parte del proyecto dependa de otra, formando así un bloqueo. Para evitar en la medida de lo posible esta problemática se ha propuesto que el responsable del módulo que provoque este bloqueo proporcione una solución temporal que consista en una funcionalidad que dada una entrada concreta proporcione una salida en específico. De esta forma, se pretende que el otro desarrollador pueda seguir con su trabajo utilizando este *workaround* hasta que el bloqueo se haya resuelto.

A lo largo del desarrollo pueden aparecer problemas que requieran más esfuerzo y trabajo del pensado originalmente, como las funcionalidades que se basan en la comunicación entre distintos entornos. Un ejemplo de esto puede ser el inicio de sesión, donde es necesario el paso de datos desde la aplicación al servidor y viceversa. Otros objetivos conflictivos que poseen alto riesgo de perjudicar la planificación propuesta son la implementación del método que permita el envío de los archivos a Moodle o la validación de parte del profesor ante el intento de acceder a un examen por parte del alumno.

[1] **Exploratory testing:** Proceso simultáneo de diseño y ejecución de tests.

Debido a esto, la planificación presentada en este documento es ligeramente distinta de la pensada en un principio. Durante el segundo sprint, la dificultad y problemas encontrados en el proceso de autentificación de la aplicación utilizando las credenciales de Moodle fue mucho mayor del pensado, esto supuso un retraso en el avance del desarrollo y, por tanto, decidí retrasar la implementación del envío de archivos desde la aplicación a Moodle al siguiente sprint.

Además, y para no sobrecargar el tercer sprint de trabajo, decidí también retrasar algunos objetivos (Tabla 1) para mantener la idea original de completar dos por sprint y evitar así bajar la calidad del producto final que acompañaría una carga excesiva de trabajo. Aunque estos problemas hayan afectado el desarrollo del proyecto, no suponen un problema mayor debido a que en la planificación original ya contaba con que podrían surgir problemas y dejé tiempo extra en los últimos sprints para poder permitirme esta flexibilidad.

Tal y como he descrito anteriormente, la subida de ficheros al servidor ha sido un objetivo conflictivo y el tiempo necesario para su cumplimiento ha sido difícil de cuantificar. Este ha sido un objetivo que estaba previsto terminar durante el tercer sprint pero se ha retrasado hasta el cuarto, debido a esto, el resto de objetivos también han tenido que ser retrasados. Sin embargo, debido a la planificación realizada, esta problemática ha sido resuelta sin problemas.

6. Desarrollo

Actividad de Moodle

Creación de un módulo de Moodle que pueda comunicarse con una aplicación.

El módulo de Moodle pensado en el primer objetivo deberá estar en un servidor, sin embargo, será necesario instalarlo también en un servidor local para poder hacer pruebas antes de subirlo. Esto reducirá la complejidad de realizar cambios en el futuro código y ayudará a detectar problemas antes de que estos interactúen con otras partes del proyecto.

Esta instalación en local requerirá de la descarga de la última versión estable de Moodle a fecha del inicio de este trabajo (Moodle 3.10.1). Como servidor web Apache y sistema de gestión de base de datos utilizaremos XAMPP en su última versión estable que soporte nuestro Moodle, la versión 7.3.27. En cuanto al sistema de gestión de base de datos mencionado anteriormente, se utilizará phpMyAdmin [10].

La creación de un nuevo módulo consiste en la creación de varias carpetas y archivos ".php" dentro del directorio Moodle/mod donde se encuentran almacenados todos.

En estos archivos se crea y guarda la información necesaria para crear nuevas instancias de 'Exam', alterar la base de datos que Moodle usa o configurar todos los parámetros necesarios para su uso por parte de los usuarios finales.

Una vez se han cumplido todos estos pasos y el módulo ha sido instaurado en el servidor, Moodle detectará automáticamente que existe un nuevo *plugin* y procederá a instalarlo. Al finalizar la instalación, ya será posible por parte de los profesores crear nuevas instancias de este recurso y la base de datos quedará actualizada con las nuevas tablas que almacenan los datos de estos.

Generación de un archivo con las notas de los alumnos.

Una vez se hayan subido los exámenes al servidor, estos podrán ser consultados por los profesores, quienes podrán corregirlos, guardar las notas y generar un archivo con esta información.

Aplicación móvil

Para un mejor entendimiento de la comunicación entre la aplicación Android y Moodle consultar el Apéndice 3, qué la explica centrándose únicamente en las peticiones que debe realizar la aplicación para obtener los datos necesarios para su correcto funcionamiento.

Creación del esqueleto de la aplicación móvil.

La base del desarrollo de esta parte del proyecto se basa en un primer esqueleto con diferentes pantallas y botones necesarios para navegar por la aplicación, sin tener ninguna funcionalidad importante implementada aún.

Lo primero que deberé tener en cuenta será en qué versión de android desarrollaré esta parte, puesto que la aplicación ha de poder aplicarse en un futuro a situaciones reales y, por lo tanto, debe poder adaptarse a futuras tecnologías.

Una vez disponible este esqueleto implementé la primera funcionalidad, un acceso a la cámara de fotos disponible en el dispositivo [11] que se utilizaría para fotografiar el examen finalizado que más tarde enviaríamos a Moodle (ver Apéndices 1 y 2).

En el momento de desarrollar esta funcionalidad tuve que tener en cuenta dos cosas importantes. Primero, todo el propósito de este proyecto reside en el

hecho de poder realizar fotografías desde un dispositivo móvil y por tanto, esta aplicación debe tener acceso al hardware de la cámara. Por otra parte, debo asegurarme que la cámara dispone de tal hardware y que está en condiciones de ser utilizado. Si alguno de estos requisitos no se cumple, no se podrán realizar las fotografías que la aplicación necesita para cumplir su propósito.

Implementación del proceso de autentificación

Durante el segundo sprint se han implementado dos funcionalidades en paralelo. La primera consiste en un sistema de autentificación por huella dactilar que sirva como una capa extra de seguridad antes de poder tomar las imágenes con la cámara. Esto se ha realizado utilizando la API de BiometricPrompt [12]. Para más información acerca de esta funcionalidad recomiendo consultar el TFG "*Diseño e implementación de una aplicación móvil para poder conectarse a Caronte y subir pdfs de exámenes previa validación como estudiante y autorización del profesor*" realizado por Víctor Aguilar Martínez y que parte de la misma base que este.

La segunda funcionalidad implementada durante este período es la que implementa la conexión de la aplicación móvil con la base de datos de Moodle, un login para los usuarios y la visualización de los cursos y exámenes[2] a los cuales el usuario puede acceder. Esta conexión a la base de datos comenzó a implementarse utilizando la API AsyncTask [13]. Aunque esta clase estaba obsoleta, la documentación encontrada y los procedimientos necesarios para que funcionara se adaptaban bien a la aplicación, sin embargo, durante el desarrollo me encontré con varios problemas relacionados con la construcción de la *request* al servidor.

Al investigar la causa y posibles soluciones a este problema encontré la API HttpURLConnection [14], que no solo solucionaba el problema que tenía, sino que también era una API más actualizada y con más funcionalidades que AsyncTask, por lo que finalmente me decanté por el uso de esta.

Un detalle que hay que tener en cuenta al trabajar con aplicaciones que se conectan a bases de datos es que mientras se establece la conexión y se realizan las consultas, la aplicación debe seguir realizando sus funcionalidades paralelamente. Debido a esto, he decidido separar ambos flujos en *threads* [15], de forma que cuando el usuario realiza una acción que requiere el acceso a la base de datos, un thread diferente del principal se encarga de procesar la información en un segundo plano y de entregársela al flujo principal una vez ha finalizado la consulta.

Siguiendo la misma lógica que con el inicio de sesión, los cursos y exámenes disponibles para el usuario se muestran en las siguientes pantallas. Sin embargo, al mostrar estos recursos en la aplicación aparece una nueva problemática, pues como en un principio se desconoce la cantidad de cursos a los que el usuario está inscrito o cuantos exámenes existen en uno, no se puede definir un número de campos exacto en la parte gráfica de la aplicación. Esto se resuelve definiendo esta parte como un *LinearLayout* [16] sin ningún elemento extra, de forma que lo mostrado sea una pantalla en blanco sin más. Por otro lado, en la parte lógica de la clase, podemos implementar un método que cree tantas instancias de texto como

[2] **Exámenes:** En este contexto, hace referencia a las instancias del módulo 'Exam'.

necesitemos, asigne los datos de los cursos a estas y los añada al *LinearLayout* que hemos creado antes.

Implementación del proceso de subida de archivos al servidor

Una de las partes clave de la aplicación móvil se basa en la capacidad de subir ficheros al servidor a través de la aplicación móvil. Para lograr esto, debemos crear primero un lugar dentro del servidor donde se almacenarán todos estos archivos. Este directorio estará dentro del módulo 'Exam' debido a que de esta manera podemos mantener todos los archivos necesarios para el funcionamiento de este bajo el mismo directorio.

Una vez hecho esto, será necesario guardar en la base de datos una nueva tabla. En ella guardaremos, para cada archivo, un identificador único, la dirección dentro del servidor donde se encuentra almacenado, el identificador del usuario que lo ha subido y el identificador del examen al que pertenece. Tanto esta información como la del archivo guardado dentro del servidor deberán estar encriptadas para asegurar que delante de un ataque al sistema, nadie tenga acceso a los exámenes de los alumnos.

Por otra parte, necesitamos que la aplicación sea capaz de: por un lado recoger el fichero desde el almacenamiento interno del móvil y, por otro lado, subirlo al servidor.

Para conseguir lo primero, he creado una nueva pantalla con los selectores adecuados y, a través del manejo de las URI[3], he identificado el lugar donde el archivo seleccionado se encuentra para almacenarlo temporalmente dentro de la aplicación. Este paso ha sido uno de los responsables de los retrasos causados por este objetivo debido al coste de tiempo necesario para aprender el funcionamiento del almacenamiento interno de un móvil android y el manejo de las URIs.

Para finalizar, utilizando la librería "MultipartUploadRequest" [17] se realiza una consulta FTP al servidor, almacenando el fichero en el directorio especificado y añadiendo una entrada a la tabla de la base de datos con la información pertinente. Este método también ha sido de alta complejidad debido a las diferencias entre las consultas HTTP con las que estoy familiarizado y las FTP, las cuales han añadido una fase de aprendizaje elevada.

Tests realizados

Para probar el correcto funcionamiento de este proyecto se han realizado dos tipos de tests. Por un lado, se ha realizado exploratory testing con el propósito de atacar el software en diferentes puntos vulnerables del sistema. Estos puntos son aquellas funcionalidades que piden cierto tipo de interacción con el usuario o las que se comunican con Moodle. Teniendo esto en cuenta, se ha intentado forzar valores incorrectos o que podrían suponer un problema para el correcto funcionamiento de la aplicación o de Moodle. Como resultado de estos tests, hemos encontrado problemas con el rendimiento de la aplicación en el momento en el que el usuario intenta pulsar un botón antes que se termine de procesar otra acción.

Por otra parte, se le ha dado a probar el software a usuarios ajenos al proyecto y con conocimientos nulos sobre el desarrollo de aplicaciones para que

[3] **URI:** Identificador de los recursos internos del móvil.

opinen sobre el producto. Como resultados de estas pruebas, he observado que la selección tanto de cursos como de exámenes es poco intuitiva, problema derivado del apartado gráfico pues presenta elementos seleccionables que, a primera vista, no lo son. Además, también he notado que realizar todo el proceso desde que el usuario se loguea hasta que sube un archivo es confuso y sin una guía de qué hacer, el usuario tarda mucho más tiempo del planeado.

7. Resultados

Para los momentos finales del desarrollo se ha terminado alcanzando los objetivos propuestos, aunque algunos de ellos hayan sido modificados desde el comienzo del proyecto.

Dicho esto, el sistema desarrollado es capaz de autentificar a un usuario, permite a este navegar por sus cursos y por los exámenes que estos tengan disponibles y, además, permite fotografiar un examen y subirlo a Moodle. Una vez hecho esto, el profesor puede recoger estos archivos, puntuarlos y descargar un fichero con todas las puntuaciones.

Teniendo en cuenta las diferentes problemáticas descritas en este artículo, los resultados obtenidos, aunque aún se encuentran dentro del margen de error previsto, no son los mejores.

Se han encadenado una serie de retrasos que han consumido prácticamente todo el tiempo reservado como seguridad ante estos contratiempos, dejando al proyecto vulnerable al menor de los problemas.
Esto, sin embargo, no resulta una catástrofe para el trabajo si tenemos en cuenta que los fallos encontrados han sucedido en las partes conflictivas en las que se esperaban.

8. Conclusiones

Este TFG se presentó como solución a una problemática común en varias asignaturas que podemos encontrar en cualquier universidad.

El proyecto a presentar se trata de un aplicativo android que permite, después de una rigurosa verificación tanto biométrica como por parte del profesor, la subida de actividades evaluativas a Moodle. Esta aplicación estará vinculada a un nuevo módulo de Moodle que permitirá visualizar y evaluar estas actividades directamente desde la plataforma sin la necesidad de descargar cientos de archivos ni de corregir esa misma cantidad de exámenes en papel para finalmente poder recoger los resultados en cómodos ficheros excel que permitan al profesorado un manejo de las puntuaciones más fácil.

Como posibles extensiones a realizar de este trabajo encontraríamos la mejora del rendimiento de la aplicación móvil, la implementación de un OCR para la corrección de las actividades y permitir a los profesores una mayor libertad a la hora de crear sus modelos de exámenes.

Agradecimientos

Me gustaría agradecer a Gemma Sánchez Albaladejo por toda la guía y ayuda proporcionada durante el proyecto. A Víctor Aguilar Martinez por la posibilidad de trabajar juntos en el desarrollo de este trabajo. Y a mi familia y amigos por el apoyo brindado durante el mismo.

Bibliografía

[1] MOODLE. 'Moodle docs'. [consultado por última vez: 25 abril 2021]. Disponible en Internet: https://docs.moodle.org/310/en/Main_page

[2] ABBYY. '¿Qué es OCR?'. [consultado por última vez: 25 abril 2021]. Disponible en Internet: https://pdf.abbyy.com/es/learning-center/what-is-ocr/

[3] APACHE. 'HTTP Server Project'. [consultado por última vez: 17 abril 2021]. Disponible en Internet: https://httpd.apache.org/docs/2.4/es/

[4] PHP. '¿Qué es PHP?'. [consultado por última vez: 20 abril 2021]. Disponible en Internet: https://www.php.net/manual/es/intro-whatis.php

[5] APACHE FRIENDS. 'XAMPP Apache + MariaDB + PHP + Perl'. [consultado por última vez: 21 febrero 2021]. Disponible en Internet: https://www.apachefriends.org/es/index.html

[6] GOOGLE. 'Primeros pasos con Google Cloud'. [consultado por última vez: 20 abril 2021]. Disponible en Internet: https://cloud.google.com/docs?hl=es

[7] LINUX. 'unoconv'. [consultado por última vez: 01 marzo 2021]. Disponible en Internet: https://linux.die.net/man/1/unoconv

[8] GHOSTSCRIPT. 'Ghostscript Overview'. [consultado por última vez: 04 marzo 2021]. Disponible en Internet: https://www.ghostscript.com/

[9] SCRUM. 'The home of Scrum'. [consultado por última vez: 25 abril 2021]. Disponible en Internet: https://www.scrum.org/

[10] PHPMYADMIN. 'Bringing MySQL to the web'. [consultado por última vez: 25 abril 2021]. Disponible en Internet: https://www.phpmyadmin.net/

[11] ANDROID. 'Camera API'. [consultado por última vez: 4 abril 2021]. Disponible en Internet: https://developer.android.com/training/camera

[12] ANDROID. 'BiometricPrompt API'. [consultado por última vez: 15 abril 2021]. Disponible en Internet: https://developer.android.com/reference/android/hardware/biometrics/BiometricPrompt

[13] ANDROID. 'AsyncTask API'. [consultado por última vez: 20 abril 2021]. Disponible en Internet: https://developer.android.com/reference/android/os/AsyncTask

[14] ANDROID. 'HttpURLConnection API'. [consultado por última vez: 25 abril 2021]. Disponible en Internet: https://developer.android.com/reference/java/net/HttpURLConnection

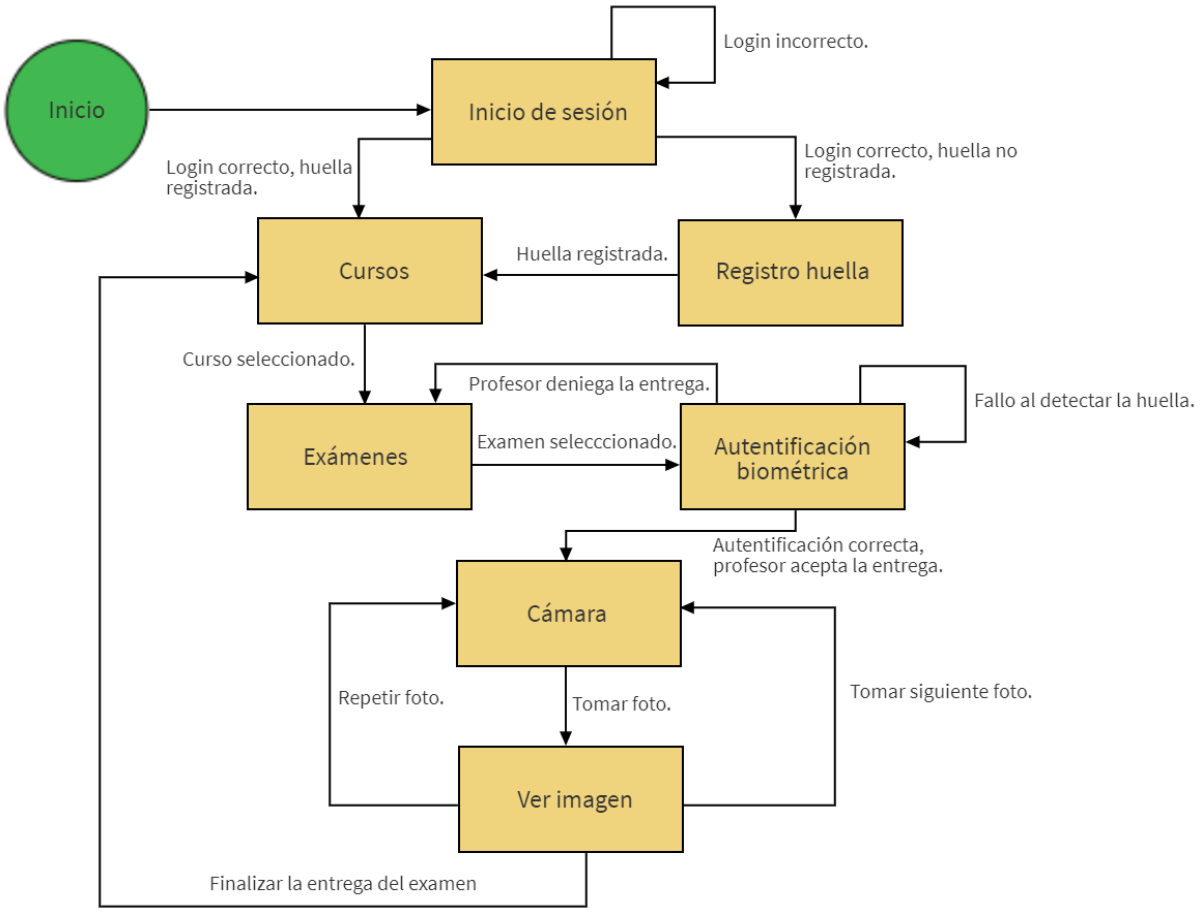
[15] ANDROID. 'Descripción general de los procesos y subprocesos'. [consultado por última vez: 25 abril 2021]. Disponible en Internet: https://developer.android.com/guide/components/processes-and-threads

[16] ANDROID. ‘LinearLayout’. [consultado por última vez: 25 abril 2021]. Disponible en Internet: https://developer.android.com/reference/android/widget/LinearLayout

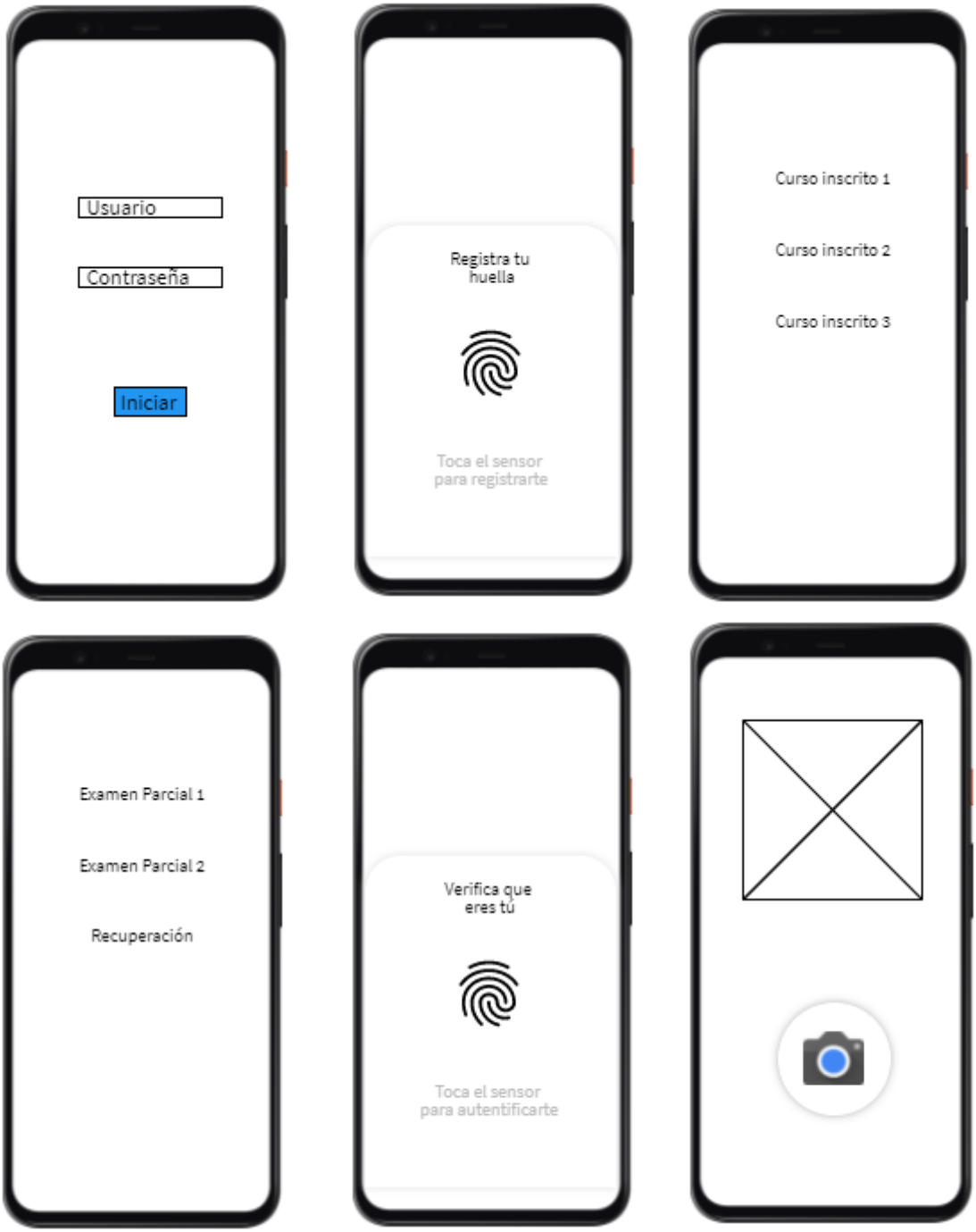
[17] JAVADOC. MultipartUploadRequest. [consultado por última vez: 29 mayo 2021]. Disponible en Internet: https://www.javadoc.io/doc/net.gotev/uploadservice/3.3/net/gotev/uploadservice/MultipartUploadRequest.html

Apéndice

Apéndice 1. Diagrama de flujo de la aplicación móvil.



Apéndice 2. Mock-up de la aplicación móvil



Apéndice 3. Comunicación entre la aplicación móvil y Moodle