
Graduado en Ingeniería Informática

Universidad a Distancia de Madrid

Departamento de Ingeniería Informática

TRABAJO DE FIN DE GRADO

# Diseño e implementación de una aplicación móvil para estudiantes con integración en plataforma de aprendizaje Moodle

Autor: Gustavo Calvo Canorea

Director: Dr. David Lizcano Casas

MADRID, JULIO DE 2020

Declaración de originalidad:

El autor de este trabajo, Gustavo Calvo Canorea, con D.N.I. 52.953.134-G, declara que el contenido de este trabajo de fin de grado es original, y en el caso de haber utilizado las ideas o contenidos de otros trabajos de otros autores están convenientemente citados, de acuerdo con las normas de citación establecidas.

El número de palabras de este trabajo, excluyendo los anexos es: 9.089

## Índice


Capítulo 1: Introducción 10

*1.1 Motivación* *10*

*1.2 Objetivos* *11*

*1.3 Estructura de la memoria* *12*

Capítulo 2: Trabajos relacionados 13

*2.1 Aplicaciones móviles y su historia* *13*

*2.2 Moodle como aplicación* *19*

Capítulo 3: Planificación temporal y costes 24

*3.1 Planificación temporal* *24*

*3.2 Costes asociados al proyecto* *25*

Capítulo 4: Propuesta 27

*4.1 Requisitos* *27*

*4.2 Análisis* *27*

*4.3 Diseño* *28*

*4.4 Implementación* *30*

4.4.1. Arquitectura 30

4.4.2 Comunicaciones y seguridad 34

4.4.3 Librerías externas 36

4.4.4 Funcionalidades de valor añadido 38

*4.5 Pruebas* *40*

Capítulo 5: Evaluación 42

*5.1 Acceso a la aplicación* *42*

*5.2 Inicio de sesión* *42*

*5.3 Consulta de asignaturas* *43*

*5.4 Consulta de tareas* *44*

*5.5 Cierre de sesión* *45*

*5.6 Segunda apertura y posteriores* *46*

*5.7 Menú lateral* *47*

*5.8 Multi-idioma* *48*

*5.9 Gestión de errores* *49*

Capítulo 6: Conclusiones 51

*6.1 Continuidad y trabajo futuro* *53*

Bibliografía 55

Anexos 58

*Anexo A1. Casos de uso* *58*

*Anexo A2. Requisitos funcionales* *61*

*Anexo A3. Casos de prueba* *64*

**Índice de figuras**

Ilustración 1. IBM Simon Personal Communicator 14

Ilustración 2. Steve Jobs presenta la App Store 15

Ilustración 3. Comparativa de Moodle Web en distintos dispositivos 20

Ilustración 4. Comparación de Moodle Web y Moodle App para iOS 21

Ilustración 5. Aplicación de Glasgow University 22

Ilustración 6. Aplicación de la Universidad Complutense de Madrid 23

Ilustración 7. Aplicación Universidad de Granada 23

Ilustración 8. Diagrama de planificación temporal 25

Ilustración 9. Casos de uso 28

Ilustración 10. Flujo de pantallas 29

Ilustración 11. Arquitectura MVC 30

Ilustración 12. Arquitectura VIPER 32

Ilustración 13. Jerarquía VIPER en la aplicación 33

Ilustración 14. Entornos de desarrollo de la aplicación 34

Ilustración 15. Accesos al llavero 36

Ilustración 16. Ejemplo de Podfile 37

Ilustración 17. Extensiones de clase 39

Ilustración 18. Acceso a la aplicación 42

Ilustración 19. Inicio de sesión 43

Ilustración 20. Actualización de información de asignaturas 44

Ilustración 21. Consulta de tareas 45

Ilustración 22. Cierre de sesión 46

Ilustración 23. Función "Recordar credenciales" 47

Ilustración 24. Detalle de menú lateral y pantalla de información de la aplicación 48

Ilustración 25. Multi-idioma 49

Ilustración 26. Gestión de errores 50

**Índice de tablas**

Tabla 1. Lenguajes de programación e interfaces de desarrollo por SO 16

Tabla 2. Tareas y fechas estimadas 24

Tabla 3. Tabla de costes 26

Tabla 4. Matriz de trazabilidad de casos de uso 40

**Resumen:**

La enseñanza cada vez tiene más presencia en entornos digitales, gracias en parte a la universalización de la banda ancha y los avances electrónicos que permiten a los estudiantes disponer de conexión a Internet desde prácticamente cualquier lugar.

Es éste el caso de las universidades a distancia, muchas de las cuales ofrecen una formación 100% en línea, a través de distintas plataformas y recursos de red, y por ello buscan mejorar los accesos al contenido y la comunicación con los estudiantes.

El objetivo de este proyecto es la realización de una aplicación móvil que permita mejorar esta comunicación mediante una interfaz visualmente agradable y una funcionalidad de fácil aprendizaje para los diversos perfiles de estudiantes que podría haber. Además, se pretende dotar a la aplicación de una arquitectura mantenible y escalable que permitan la incorporación de nuevas funcionalidades en el futuro.

**Palabras clave:**

Aplicación, iOS, Swift, Moodle, teléfonos inteligentes

**Abstract:**

Teaching is increasingly present in digital environments, thanks in part to the universalization of broadband and electronic advancements that enable students to have an Internet connection from virtually anywhere.

This is the case of distance learning universities, many of which offer 100% online training, through different platforms and network resources, and therefore seek to improve content access and communication with students.

The aim of this project is the creation of a mobile application that allows these universities to improve this communication through a visually pleasing interface and easy-to-learn functionality for the possible different student profiles. In addition, it is intended to provide the application with a maintainable and scalable architecture that allows adding new functionalities in the future.

**Keywords:**

Application, iOS, Swift, Moodle, smartphone

# CAPÍTULO 1: INTRODUCCIÓN

## 1.1 Motivación

Los avances tecnológicos y la cobertura de banda ancha de conexión a Internet han propiciado en los últimos años el crecimiento de la enseñanza digital o *e-learning*. La flexibilidad horaria y geográfica que ofrece, junto con el uso de metodologías interactivas permiten que numerosas entidades de formación, como universidades o institutos, hagan uso de Internet para impartir sus clases y/o mantener el contacto entre el profesorado y los alumnos. Para ello, hacen uso de plataformas online de gestión de contenidos para el aprendizaje (*Learning Content Management System* o LCMS por sus siglas en inglés), las cuales permiten crear y manejar el contenido de una parte de un programa de educación tales como un curso o una asignatura.

Algunas de estas plataformas son Chamilo (Chamilo, 2020), edX (edX Inc., 2020) o SWAD (openSWAD, 2020). Sin embargo, si hay una plataforma que se distingue del resto es Moodle, de distribución libre y licencia GNU GPL que, a fecha de marzo de 2020 dispone de 22 millones de cursos en el mundo y 188 millones de usuarios repartidos en 131 mil sitios, de los cuales más de 10.000 sitios se encuentran únicamente en España (Moodle, 2020).

Por otro lado, un estudio de Learning House and Aslanian Market Research afirma que, virtualmente, todos los estudiantes universitarios de estudios en línea disponen de teléfono inteligente o tableta digital y que el 67% de ellos utilizan dispositivos móviles para complementar sus cursos en línea (Magda & Aslanian, 2018). Es por ello que se antoja necesario asegurar que el contenido del curso y los sitios web están optimizados para dispositivos móviles si desean potenciar y retener a sus estudiantes.

En este aspecto, Moodle ofrece una aplicación móvil que permite a los estudiantes conectar con su centro educativo si éste tiene permitido en los ajustes de la plataforma la comunicación con otras aplicaciones. Sin embargo, a pesar de ser una aplicación muy completa e integrada, la aplicación oficial de Moodle es poco personalizable para las necesidades de cada centro educativo. Es por ello que se propone en el presente trabajo

el desarrollo de una aplicación móvil que permita acceder a los contenidos seleccionados de Moodle de una universidad, pudiendo incluir información adicional no residente en la plataforma Moodle y adaptando el diseño a la imagen de marca de la universidad.

## 1.2 Objetivos

La finalidad de este trabajo es desarrollar una aplicación que permita a un estudiante acceder a un entorno educativo que utilice Moodle para, a través de esta, poder consultar sus asignaturas matriculadas, notas provisionales y/o definitivas, tareas y calificaciones de estas y comentarios del profesor si los hubiera.

Esta aplicación se desarrollará en el presente trabajo de forma piloto para dispositivos iOS, en particular iPhone, haciendo uso del lenguaje de código abierto Swift 5.2 (Apple Inc, 2020).

Para ello, se hará uso del servicio gratuito que moodleCloud (Moodle Pty Ltd, 2020) ofrece, el cual dispone de, entre otros, los siguientes servicios:

- Número máximo de usuarios: 50
- Tamaño máximo de almacenamiento: 200 MB
- Última versión estable de Moodle
- Actividades y cursos ilimitados
- Nombre personalizado del sitio
- Habilitado acceso desde aplicación móvil

En este servicio se dará de alta los suficientes datos que permitan probar las funcionalidades de la aplicación móvil desarrollada.

La aplicación móvil, en su código fuente, dispondrá de un archivo de constantes donde se podría modificar la dirección del sitio Moodle que se desee utilizar para facilitar su publicación con un entorno en producción si fuera necesario.

Para ilustrar de una forma más cercana a la realidad el trabajo, la aplicación se desarrollará bajo demanda de una universidad ficticia llamada Madrid Open University, y el usuario

objetivo de la aplicación móvil será un estudiante de dicha universidad. La aplicación tendrá el nombre comercial “MadOU Aula Virtual”.

## 1.3 Estructura de la memoria

A continuación, se expone un resumen del contenido de cada uno de los capítulos que componen el presente documento:

- Capítulo 1, “Introducción”: Presentación del documento y motivaciones que llevan a la realización de este. También se muestra un breve resumen de la memoria.
- Capítulo 2, “Trabajos relacionados”: Exposición de trabajos actuales relacionados con la propuesta, estableciendo una comparativa con el proyecto propuesto.
- Capítulo 3, “Planificación temporal y costes”: Detalle de la planificación de las tareas del proyecto junto con los costes asociados a las mismas. Este capítulo incluye también los costes de herramientas utilizadas durante el desarrollo del proyecto, así como los costes necesarios para su puesta en producción.
- Capítulo 4, “Desarrollo”: Exposición en detalle de las fases del desarrollo, tales como requisitos, análisis, implementación y pruebas.
- Capítulo 5, “Evaluación”: En este capítulo se detalla la evaluación del producto para verificar el cumplimiento de los objetivos del software desarrollado.
- Capítulo 6, “Conclusiones”: Cierra el proyecto con una exposición de las conclusiones obtenidas en la elaboración del trabajo, así como propuestas de desarrollo futuro.

Además, el trabajo cuenta con una serie de anexos que amplían información considerada relevante durante la realización de la presente memoria.

# CAPÍTULO 2: TRABAJOS RELACIONADOS

## 2.1 Aplicaciones móviles y su historia

Las llamadas aplicaciones móviles, o simplemente *apps* por su abreviatura en inglés (*mobile application*) son programas informáticos o aplicaciones de software para ser ejecutadas en dispositivos móviles tales como teléfonos inteligentes, tabletas (o *tablets*), relojes inteligentes, etc.

La historia de las aplicaciones móviles está ligada a la aparición de los dispositivos móviles mismos. En un principio fueron ayudas a la funcionalidad misma de los teléfonos, tales como agendas de contactos o lectura de mensajería SMS. A medida que los teléfonos móviles y primeros dispositivos PDA aumentaban sus capacidades computacionales y su memoria, estos dispositivos fueron incorporando aplicaciones más complejas (calculadora, calendario, juegos…), generalmente desarrollados por el propio fabricante del terminal e incluidos con el sistema operativo de fábrica. A continuación, se describen algunos de los hitos en la historia de las aplicaciones móviles a través de distintos dispositivos que, si bien no fueron los primeros o los únicos en implementar ciertas funcionalidades, si marcaron las tendencias posteriores contribuyendo al asentamiento de prácticas y estándares.

Se considera que el primer teléfono inteligente fue el IBM Simon Personal Communicator (Wikipedia, 2020), lanzado por IBM el 1993, el cual incluía como aplicaciones una agenda de contactos, calendario, reloj mundial y calculadora.


*Ilustración 1. IBM Simon Personal Communicator*

En 2002, Wap Forum (fundado por Sony Mobile Communications, Nokia, Motorola y Openwave) desarrolla el estándar WAP (Wikipedia, 2020) (*Wireless Application Protocol* o protocolo de aplicaciones inalámbricas) el cual define un entorno de aplicación y una serie de protocolos de aplicaciones y servicios accesibles a través de terminales móviles. Este nuevo protocolo permite que los desarrolladores diseñen aplicaciones de interconexión para terminales móviles, con predominancia de los teléfonos.

En 2003 se realizó el siguiente salto transcendente en la historia de los dispositivos inteligentes con el lanzamiento del primer teléfono BlackBerry (Wikipedia, 2020), de Research in Motion (RIM): un teléfono móvil con servicio de correo y navegación web, entre otros servicios inalámbricos. Su sistema operativo, BlackBerry OS, al igual que Symbian permitía el desarrollo de aplicaciones por parte de desarrolladores independientes, haciendo uso de la API disponible, si bien el uso de ciertas funcionalidades requería una firma digital.

El gran cambio es, sin duda, la aparición del iPhone de Apple (Wikipedia, 2020) en 2007, el cual daba un salto de gigante en la evolución de las aplicaciones móviles a través de un teclado digital (hasta este momento la mayor parte de los dispositivos utilizaban teclado físico), pantalla multi-táctil y un navegador web funcional. Junto con el iPhone, Apple

presentó la App Store, un servicio que permitía a los usuarios buscar y descargar aplicaciones desarrolladas por desarrolladores independientes con el iOS SDK y distribuidas por Apple. Estas aplicaciones podían ser de pago o gratuitas. En menos de 4 años desde su lanzamiento, App Store superó los 25 mil millones de descargas de aplicaciones (Apple Inc, 2012).


*Ilustración 2. Steve Jobs presenta la App Store*

En 2008 se presenta el primer teléfono con sistema operativo Android (Wikipedia, 2020), el HTC Dream. Este sistema operativo, Android, fue adquirido por Google unos años antes y adaptado a la telefonía móvil. Este sistema operativo se caracteriza por tener un código fuente licenciado principalmente bajo Licencia Apache, lo cual ha permitido su adaptación a teléfonos de muchos y muy diferentes fabricantes, llegando a una cuota de mercado en marzo de 2020 del 72,26% de los teléfonos inteligentes, frente a un 27,03% correspondiente a iOS de Apple, el segundo sistema operativo en cuota de mercado (StatCounter, 2020). La tienda de aplicaciones de los dispositivos Android es Play Store.

En la actualidad existen otros sistemas operativos como Windows Phone de Microsoft (actualmente descontinuado), EMUI de Huawei o PureOS entre otros, pero su cuota de mercado es mínima comparada con la que Android y iOS suman.

Según su naturaleza, las aplicaciones móviles pueden ser nativas, híbridas o web. Sus características son explicadas en los siguientes epígrafes.

### *2.1.1 Aplicaciones nativas*

Las llamadas aplicaciones nativas son las desarrolladas específicamente para cada sistema operativo, sea iOS, Android, Windows Mobile… Es por ello, que una aplicación desarrollada para iOS no es compatible con Android, y viceversa. Estas aplicaciones son desarrolladas con los respectivos lenguajes de programación de cada plataforma y haciendo uso de las interfaces de desarrollo (IDE) propias, los cuales se muestran en la siguiente tabla:

*Tabla 1. Lenguajes de programación e interfaces de desarrollo por SO*

| Sistema Operativo | Lenguaje de programación | Entorno de desarrollo |
|---|---|---|
| **Android** | Java, Kotlin (desde 2017) | Android Studio |
| **iOS** | Objective-C, Swift (desde 2014) | Xcode |
| **Windows Mobile** | C#, VB.NET, C/C++ | Visual Studio |

Las principales ventajas de las aplicaciones nativas son:

- Mejor experiencia de usuario. El funcionamiento es más sencillo al ser similar al experimentado por otras aplicaciones y el propio sistema operativo a lo largo del tiempo.
- Mejor rendimiento y velocidad. Al hacer uso de las API ofrecidas por el fabricante, el uso de recursos y consumo de batería está mucho más optimizado.
- Protección de datos. El hardware y el propio sistema operativo ofrece recursos de seguridad para la protección de datos a las aplicaciones nativas.
- Soporte por parte de la tienda de aplicaciones del fabricante (App Store de Apple, Play Store de Android…)

Sin embargo, presentan también desventajas:

- Falta de flexibilidad. El desarrollo debe hacerse para cada plataforma por separado.

- Desarrollo de mayor coste. Normalmente el desarrollo de una aplicación involucra distintas plataformas, por lo que, al realizar el desarrollo para las distintas plataformas, el coste de este aumenta.
- Publicación dependiente de aprobación. Para ser publicada, debe pasar una verificación de la tienda de aplicaciones que evalúa su estabilidad y funcionalidad, así como su contenido y podría dificultar actualizaciones o incluso la publicación.

Como ejemplo de aplicaciones nativas de uso cotidiano se encuentran Whatsapp, Telegram, Mi Vodafone…

### *2.1.2 Aplicaciones web*

A diferencia de las aplicaciones nativas, las aplicaciones web se adaptan a cualquier sistema operativo, estando únicamente limitadas por las características del navegador utilizado. Su uso se basa en paginas web que ofrecen funcionalidades similares a las aplicaciones nativas. Sus principales ventajas son:

- Más barato, al necesitar únicamente un desarrollo para todas las plataformas.
- Diseño *responsive*. La aplicación se puede adaptar a múltiples plataformas con diversos tamaños de pantalla.
- Accesibilidad. Se puede acceder a ellas desde cualquier dispositivo con conexión a Internet.
- Siempre actualizada. Al no depender de publicación a través de tiendas de aplicaciones, el usuario siempre accede a la última versión en producción. Los despliegues de las actualizaciones de la aplicación son más ágiles.
- Incremento de almacenamiento. Mientras que las aplicaciones móviles dependen de un espacio de almacenamiento local en el dispositivo, las aplicaciones web disponen de un espacio virtualmente infinito.

Entre las desventajas de su uso se encuentran las siguientes:

- Dependientes de Internet. Si el dispositivo no tiene conexión o ésta es de baja calidad, el usuario no podrá acceder a la aplicación.

- Seguridad. Los datos del usuario se mueven a través de la red continuamente, además de ser almacenado remotamente, por lo que el desarrollo requiere una especial atención en materia de seguridad.
- Menor velocidad que en una aplicación que actúa de forma local.
- Dependencia del navegador utilizado. Los navegadores pueden ser de distintos fabricantes, de distintas versiones y con distintas características, por lo que el desarrollo podría no funcionar correctamente en algunos dispositivos.

Ejemplos de aplicaciones web son Google Docs y Codepen.io

### *2.1.3 Aplicaciones híbridas*

Este tipo de aplicaciones combinan las características de las aplicaciones web y de las aplicaciones nativas según conveniencia.

El desarrollo principal se realiza bajo lenguajes web como Javascript, Angular o CSS al igual que las webs, pero son encapsuladas dentro de una aplicación móvil desarrollada que hace de pasarela entre la parte web y las características del dispositivo.

Sus ventajas son:

- Desarrollo unificado. La mayor parte del código utilizado es único para todas las plataformas, reduciéndose la cantidad de código nativo utilizado de forma considerable.
- Despliegue más rápido. La mayor parte de las actualizaciones son web, por lo que no es necesario actualizar la aplicación nativa ni pasar por la validación de la tienda de aplicaciones para poder publicar la mayoría de los cambios.
- Accesos a bajo nivel. El uso de una capa nativa permite a la aplicación web poder acceder a componentes hardware de los dispositivos inteligentes.
- Posibilidad de funcionamiento sin conexión a Internet. Aunque parte de su funcionalidad requiere conexión a Internet, se puede contar con funcionalidades del dispositivo para ofrecer servicio sin conexión.

Entre sus desventajas están:

- Rendimiento. La creación de una capa extra entre la parte web y la parte nativa se traduce en pérdida de rendimiento. Mark Zuckerberg, CEO de Facebook, llegó a decir “el mayor error que cometimos como compañía fue apostar por HTML5 en lugar de nativo” (King, 2012).
- Experiencia de usuario más pobre. Al compartir desarrollo para múltiples plataformas, adaptar su diseño más a una puede hacer que los usuarios de otra plataforma tengan peor experiencia en el uso de esta, por lo que es necesario encontrar un punto intermedio entre plataformas en el diseño y desarrollo de la aplicación.
- Requiere de desarrolladores nativos. Si bien la mayor parte del desarrollo es común, aún así se requiere de desarrollo nativo para resolver distintos problemas funcionales.

Algunos ejemplos de este tipo de aplicaciones son Instagram, Facebook o ING.

## 2.2 Moodle como aplicación

La plataforma Moodle ofrece tanto una aplicación web como una aplicación nativa que permite la conexión a los servicios web ofrecidos. En los siguientes epígrafes se realizará un breve análisis de ambas:

### *2.2.1 Moodle Web*

Esta es la aplicación de Moodle por defecto. Al crear un sitio en la plataforma, automáticamente se puede acceder con usuarios válidos a la aplicación web desde cualquier plataforma que disponga de un navegador compatible y conexión a Internet. La aplicación web muestra todas las opciones que por tipología y permisos disponga el usuario, sea este estudiante, profesor, administrador…

Además de las ventajas propias de aplicación web explicadas anteriormente, las principales ventajas de la aplicación Moodle son:

- Barato. No supone un coste adicional a las propias licencias de Moodle.

- Multiplataforma y multidispositivo. Cualquier dispositivo con conexión a Internet y un navegador compatible puede acceder.
- Segura. No requiere de habilitaciones de acceso para aplicaciones externas.
- Diseño adaptable (*responsive*). Como muestra la siguiente ilustración, la aplicación se adapta a distintos tamaños de pantalla, sea ésta de ordenador de escritorio, *tablet* o *smartphone*.

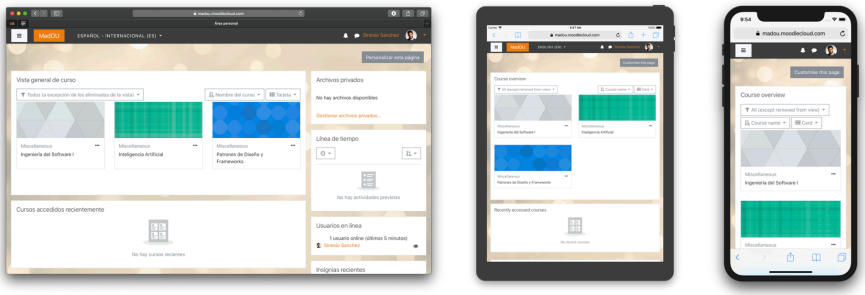

*Ilustración 3. Comparativa de Moodle Web en distintos dispositivos*

Entre sus desventajas encontramos las siguientes:

- Lenta, pues todas las cargas dependen de comunicación web.
- Visualmente poco atractiva. Al tener que hacer uso de componentes adaptables y compatibles con distintos navegadores, la visualización de estos no puede ser muy compleja, por lo que se pierde en diseño.
- Menor usabilidad. La información mostrada no se adapta al dispositivo por lo que, a menudo, el desplazarse entre secciones o visualizar cierta información puede ser complicado en dispositivos más pequeños como los teléfonos móviles.
- Menor posibilidad de personalización. Si bien es posible configurar los colores de la aplicación e incluir imágenes que puedan estar vinculados a la imagen de marca de la entidad educativa, estos están limitados en cuanto a las posibilidades que podría mostrar una aplicación nativa propia.

### *2.2.2 Aplicación Moodle*

Moodle dispone de aplicaciones nativas en App Store (Moodle, 2020), para dispositivos iOS; y Google Play, para dispositivos Android. Ambas aplicaciones presentan un diseño adaptado a terminales móviles con mejor experiencia de usuario, como muestra la siguiente ilustración:

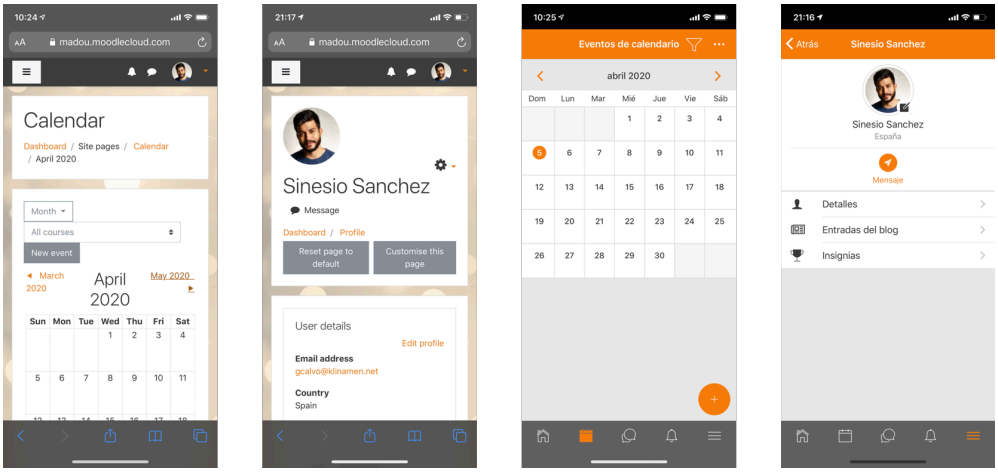


*Ilustración 4. Comparación de Moodle Web y Moodle App para iOS*

Entre las principales ventajas de la aplicación Moodle nativa se encuentran:

- Sin coste, pues no requiere ningún desarrollo
- Usabilidad. La aplicación se encuentra adaptada a los dispositivos móviles de forma nativa, por lo que la navegación entre distintos apartados es más intuitiva y cómoda.
- Mejor diseño visual, debido a su adaptación a elementos nativos.

Como desventajas presenta las siguientes:

- Mucha limitación en la personalización, tanto de imagen de marca como de funcionalidades que se quieren ofrecer en la versión móvil.
- Requiere actualizaciones. Si bien las actualizaciones de la información (o de los datos) son realizadas por Moodle, es probable que para disponer de características nuevas el usuario deba mantener actualizada la aplicación.

- Complejidad. Al no poder limitar las funcionalidades o rediseñar la ubicación de estas, la aplicación dispone de muchas posibilidades que requieren de aprendizaje por parte del usuario, y para algunos perfiles de usuario podría resultar muy complejo.
- No escalable. Al depender de Moodle no es posible añadir módulos que permitan funcionalidades distintas a las ofrecidas por la plataforma Moodle. Por ejemplo, integración con una secretaría virtual.
- Es necesario habilitar las conexiones a los servicios web REST.

### *2.2.2 Otras aplicaciones nativas*

Como se hacía referencia en el Capítulo 1, hay más de 131.000 sitios que hacen uso de Moodle en el mundo. Muchos de ellos son universidades y algunas de ellas han desarrollado aplicaciones nativas para adaptar el acceso de los estudiantes a Moodle y destacar su imagen de marca apostando por realizar aplicaciones que mejoren el acceso y las gestiones de sus estudiantes a través de dispositivos móviles. Algunos ejemplos de estas universidades son los siguientes:

- Universidad de Glasgow (University of Glasgow, s.f.)

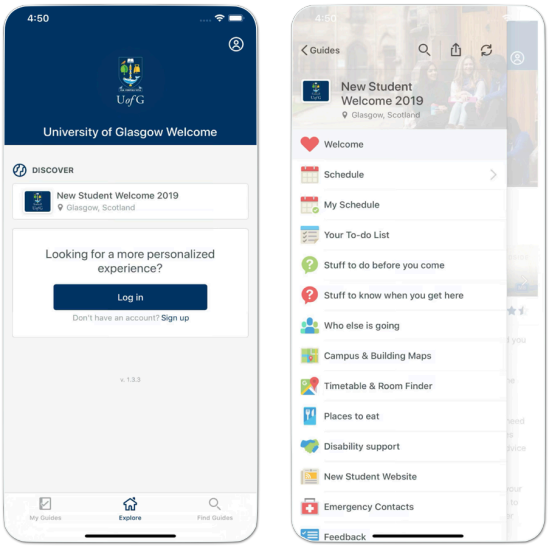


*Ilustración 5. Aplicación de Glasgow University*

- Universidad Complutense de Madrid (Universidad Complutense Madrid, s.f.)

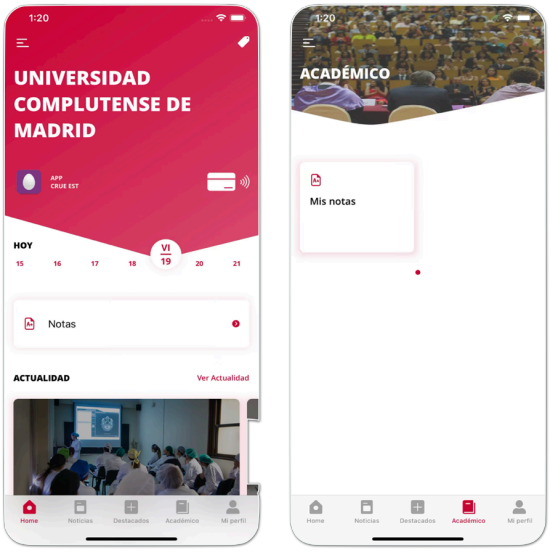


*Ilustración 6. Aplicación de la Universidad Complutense de Madrid*

- Universidad de Granada (Universidad de Granada, s.f.)

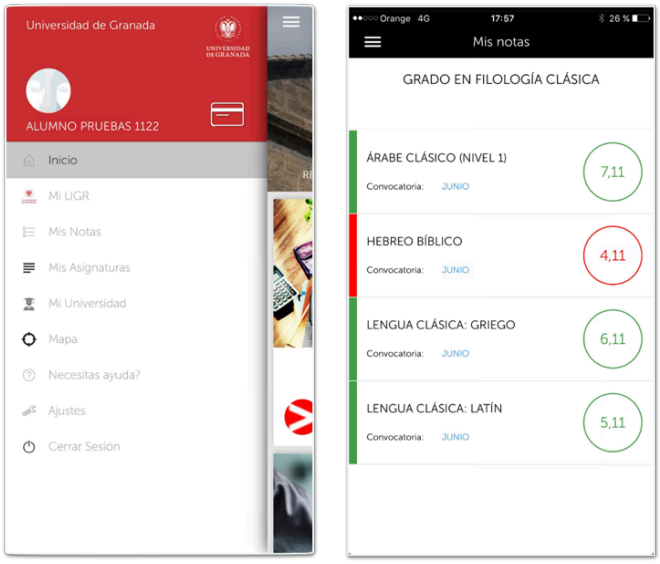


*Ilustración 7. Aplicación Universidad de Granada*

# CAPÍTULO 3: PLANIFICACIÓN TEMPORAL Y COSTES

## 3.1 Planificación temporal

La planificación del proyecto se encuentra adaptada a la naturaleza de un Trabajo de Fin de Grado, siendo llevada a cabo por un estudiante que compagina su actividad académica con una actividad profesional de tiempo completo. Así mismo, esta planificación se encuentra adaptada a las fechas de entregas de proyecto fijadas por la Universidad. Es por ello que se ha estimado que cada día de trabajo suponen 3 horas efectivas, siendo 15 las horas semanales aproximadas dedicadas al proyecto, no teniéndose en cuenta de forma teórica los fines de semana. Es por ello, que cabe destacar que, si esta planificación estuviera adaptada a un desarrollo empresarial, los costes y tiempo asociados serían distintos.

En la siguiente tabla se exponen las tareas y estimación de fechas asociadas a las mismas:

*Tabla 2. Tareas y fechas estimadas*

| *Tarea* | *Inicio* | *Finalización* | *Tiempo* |
|---|---|---|---|
| *Requisitos* | 20/02/2020 | 24/02/2020 | 3 días |
| *Diseño y prototipo* | 25/02/2020 | 13/03/2020 | 14 días |
| *Creación y gestión de entorno Moodle* | 14/03/2020 | 27/03/2020 | 10 días |
| *Creación de proyecto y arquitectura* | 28/03/2020 | 10/04/2020 | 10 días |
| *Desarrollo e integración* | 11/04/2020 | 27/05/2020 | 33 días |
| *Pruebas y estabilización* | 28/05/2020 | 04/06/2020 | 6 días |
| *TOTAL* | 20/02/2020 | 04/06/2020 | 76 días |

Se estima, por tanto, en 76 días o 228 horas el tiempo de desarrollo del proyecto.

El siguiente diagrama de Gantt muestra la planificación temporal anteriormente mencionada de forma gráfica:

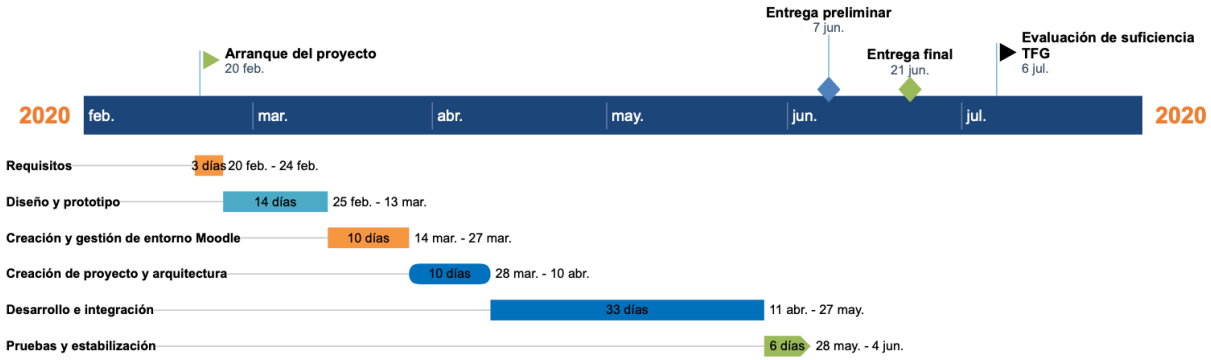


*Ilustración 8. Diagrama de planificación temporal*

## 3.2 Costes asociados al proyecto

En este apartado se listan los costes asociados al proyecto. Para ello, se da por supuestas una serie de premisas de cara a su puesta en producción:

- MadOU dispone de un sistema Moodle actualmente montado y funcionando.
- Los costes de administración de la plataforma Moodle necesarios para habilitar los accesos no se incluyen en el coste del desarrollo de la aplicación, si bien están contemplados en tiempo en las fases de análisis y de desarrollo e integración por la parte que corresponde al desarrollo de la aplicación.

En cuanto a los costes de desarrollo, a continuación, se detalla el software que se utilizará durante el desarrollo y el coste de su licencia si la tuviera. Obsérvese que dicha información no incluye los costes asociados a la elaboración del presente documento, únicamente atiende al desarrollo del proyecto:

- Sketch. Herramienta de diseño utilizada para realizar prototipos de la aplicación y la generación de los recursos gráficos asociados. Coste de la licencia individual: 114,83€

- Xcode. Interfaz de desarrollo utilizada para el desarrollo de la aplicación, incluye simuladores de dispositivos iOS. Coste: 0€
- Librerías de terceros. Todas las librerías, bajo distintas licencias, son de código abierto y gratuitas. Coste: 0€
- Licencia de desarrollador Apple. Necesaria para distribución de aplicaciones en dispositivos reales, tanto para pruebas como para producción. Coste: 99€
- MoodleCloud. Utilizado para crear un entorno Moodle de pruebas y poder realizar la integración de servicios. Coste: 0€

Además, para enriquecer de información los costes asociados al desarrollo, se ha tenido en cuenta el coste de las horas de trabajo del desarrollador, cuyo coste por hora se ha establecido en 9€.

La siguiente tabla muestra un resumen de la información anterior con todos los costes asociados:

*Tabla 3. Tabla de costes*

| *Concepto* | *Coste* |
|---|---|
| *Licencia Sketch* | 114,83€ |
| *Licencia Xcode* | 0,00€ |
| *Librerías utilizadas* | 0,00€ |
| *Licencia de desarrollador Apple* | 99,00€ |
| *Alojamiento MoodleCloud* | 0,00€ |
| *Repositorio Bitbucket* | 0,00€ |
| *Coste desarrollo (228 h. x 9€/h.)* | 2.052,00€ |
| ***TOTAL*** | **2.265,83€** |

Por lo tanto, suponiendo un coste asociado al desarrollador, el coste total del proyecto sería de 2.265,83€

# CAPÍTULO 4: PROPUESTA

## 4.1 Requisitos

La aplicación móvil dispondrá de las siguientes funcionalidades para los estudiantes:

- Acceso a la aplicación a través de identificación de forma segura.
- Visualización de asignaturas en curso.
- Visualización de tareas disponibles por asignatura.
- Visualización de comentarios del profesor en las tareas, si éstas dispusieran de comentario.
- Visualización de las notas de las tareas y de las asignaturas, si éstas hubiesen sido evaluadas.
- Se han identificado también las siguientes restricciones:
- Los usuarios no pueden realizar acciones que modifiquen de alguna manera los datos de la plataforma Moodle, siendo la aplicación "MadOU Aula Virtual" únicamente de carácter informativo para el usuario.
- La aplicación móvil requiere de un usuario activo en la plataforma Moodle para funcionar.
- La aplicación móvil necesita conectividad a Internet para funcionar.

## 4.2 Análisis

En base a los requisitos expuestos en el epígrafe anterior, se identifica un único actor que interactuará con el software a desarrollar: el estudiante. Este actor llevará a cabo distintas actividades diferenciadas, las cuales han sido identificadas como casos de uso. La siguiente figura muestra los distintos casos de uso, los cuales definen la comunicación entre:

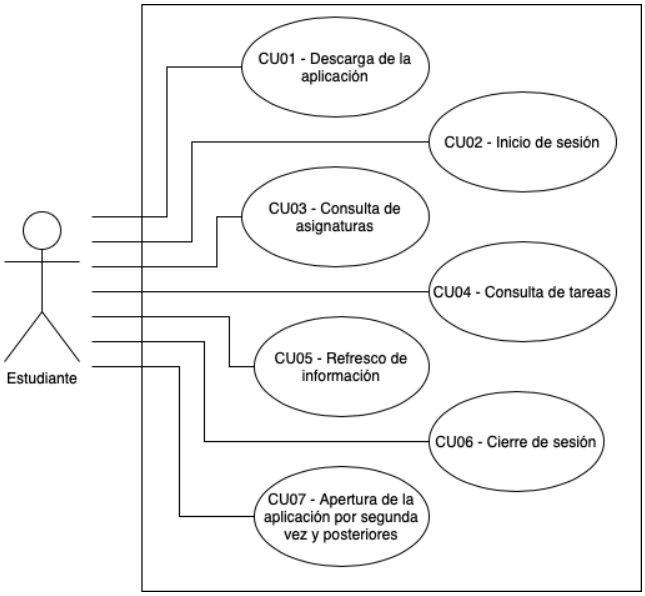


*Ilustración 9. Casos de uso*

Estos casos de uso han sido analizados y defininidos con precondiciones y flujos, y su detalle puede ser encontrado en el Anexo A1.

Con los casos de uso definidos, se pueden establecer diferentes funcionalidades que la aplicación debe ser capaz de cumplir. Estas funcionalidades son conjuntos de entradas, comportamientos y salidas que servirán para verificar que el software cumple con el comportamiento esperado. Se han elaborado unos requisitos funcionales, que pueden ser encontrados en el Anexo A2, los cuales detallan las dependencias y el comportamiento esperado.

## 4.3 Diseño

Para cumplir con los requisitos funcionales anteriormente expuestos, se ha elaborado un prototipo de flujo de pantallas en el cual basar el desarrollo. La siguiente figura muestra dicho flujo de pantallas de forma esquematizada:

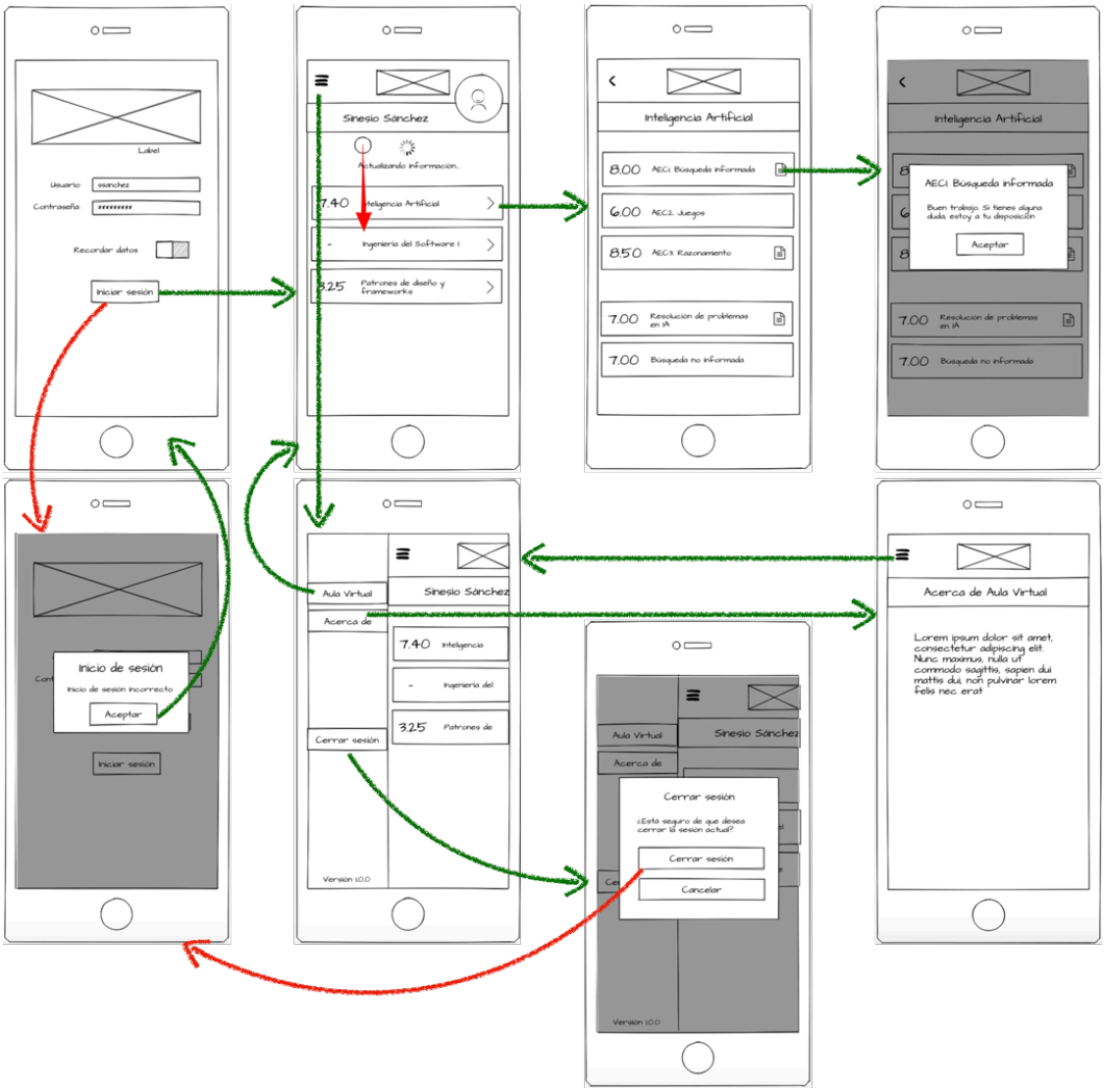


*Ilustración 10. Flujo de pantallas*

En el flujo de pantallas expuesto en la figura anterior, se pueden ver representados los casos de uso asociados al desarrollo de la aplicación. Esto es excluyendo el caso de uso de descarga de la aplicación por parte del usuario desde la App Store. Obsérvese no solo los flujos entre pantallas, sino también mensajes de error, confirmación o de información, además de gesto táctil que permita la actualización de la información mostrada.

También se ha diseñado un menú lateral que permitiría incluir futuras funcionalidades a la aplicación, facilitando la navegación a las mismas.

## 4.4 Implementación

### *4.4.1. Arquitectura*

Existen varias posibles arquitecturas para el desarrollo de aplicaciones nativas en iOS, las cuales siguen diferentes patrones de diseño y se adaptan mejor o peor a las necesidades de diferentes proyectos. La arquitectura más sencilla que soporta iOS es la MVC (Model-View-Controller) (Pathak, 2018), que encaja en proyectos muy sencillos. Su nombre implica tres conceptos: Modelo, Vista y Controlador. La siguiente figura ilustra la interacción de las tres capas:

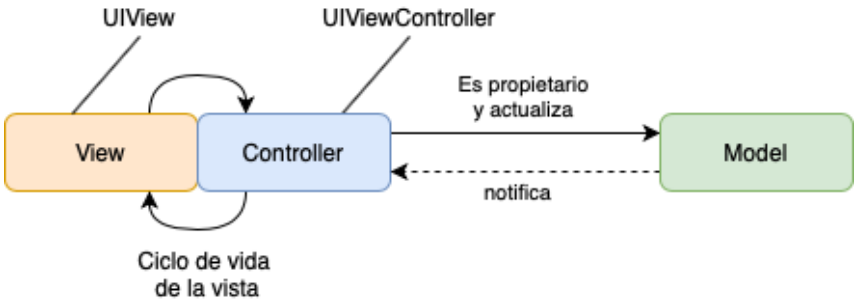


*Ilustración 11. Arquitectura MVC*

Cada capa tiene una funcionalidad definida:

- Modelo. Es una colección de diferentes clases que representan la lógica de negocio, como un modelo de datos. También define las reglas por las cuales los datos pueden ser modificados y manipulados.
- Vista. Representa las interfaces a través de las cuales el usuario interactúa con las funcionalidades de la aplicación como, por ejemplo, un botón o una caja de texto.
- Controlador. Es el mediador entre el modelo y la vista. Decide cómo el modelo debe ser representado en la vista y procesa las interacciones del usuario a través del modelo de datos.

Sin embargo, el modelo MVC presenta limitaciones cuando la aplicación crece en complejidad, puesto que toda la lógica se centra en el Controlador. Es por ello que, si no se cuida bien el código, se corre el riesgo de que las siglas MVC representen el llamado

*Massive View Controller* (Controlador de Vista Masivo), al añadir cada vez más y más funcionalidad a los controladores de vista, haciendo muy difícil su comprensión y mantenimiento.

Es por ello que surgen otros modelos de arquitectura que añaden capas con tareas específicas que, a pesar de ser más complejas de implementar, permiten un crecimiento más ordenado de la aplicación. Algunas de estas arquitecturas son las siguientes:

- MVP (Model-View-Presenter). Toma la capa de visualización como un único elemento lógico, incluyendo tanto la vista como el controlador en el mismo, y añade una capa de presentación que haga de intermediario entre la vista y el modelo.
- MVVM (Model-View-ViewModel). Crea una capa con un modelo que interpreta la vista y, a través del cual, la vista interactúa con el modelo. En este caso también se toma la vista como el conjunto de vista y controlador de vista.

La arquitectura propuesta para el desarrollo de la aplicación va un paso más allá, y se basa en las llamada filosofía de arquitectura limpia (o *clean architecture*) creada por uno de los autores del Manifiesto Ágil (*Agile Manifesto*) (Beck, y otros, 2001), Robert C. Martin en su libro Clean Architecture (Martin, 2018), junto con las recomendaciones que el mismo autor realiza en su libro Clean Code (Martin, 2009). La arquitectura limpia pretende ofrecer una metodología de codificación que facilite el desarrollo de un código de calidad, más fácil de mantener, escalar y con menos dependencias. Para ello divide la estructura lógica de la aplicación en distintas capas de responsabilidad.

En particular, la arquitectura software de la aplicación se basa en el modelo VIPER (Mutual Mobile, 2014), un modelo de arquitectura limpia propuesto por Mutual Mobile que adapta las arquitecturas limpias al desarrollo de aplicaciones iOS. VIPER es el acrónimo de las cinco capas lógicas: View, Interactor, Presenter, Entity, Router. Cada una de estas capas tiene una funcionalidad muy específica dentro de la aplicación:

- View (Vista): Corresponde al conjunto formado por el archivo de vista del *Interface Builder* en Xcode (extensión xib) y su correspondiente *ViewController* o controlador de vista. La responsabilidad de esta capa es enviar las interacciones

del usuario a la capa Presenter y mostrar lo que sea que el Presenter le ordene. Se encarga de la parte visual de la aplicación.

- Interactor (Interactuador): Contiene la lógica de negocio.
- Presenter (Presentador): Su responsabilidad es obtener los datos del Interactor cuando una acción de usuario a través de la vista lo requiere y, una vez que la tiene, se la envía a la vista para que la muestre. También interactúa con el Router para la navegación entre pantallas.
- Entity (Entidad): Contiene modelos de datos para ser usados por el Interactor.
- Router: Se encarga de la lógica de navegación entre pantallas. En el código de la aplicación se recoge bajo el nombre de *wireframe*.

La siguiente imagen muestra gráficamente la arquitectura de la aplicación:

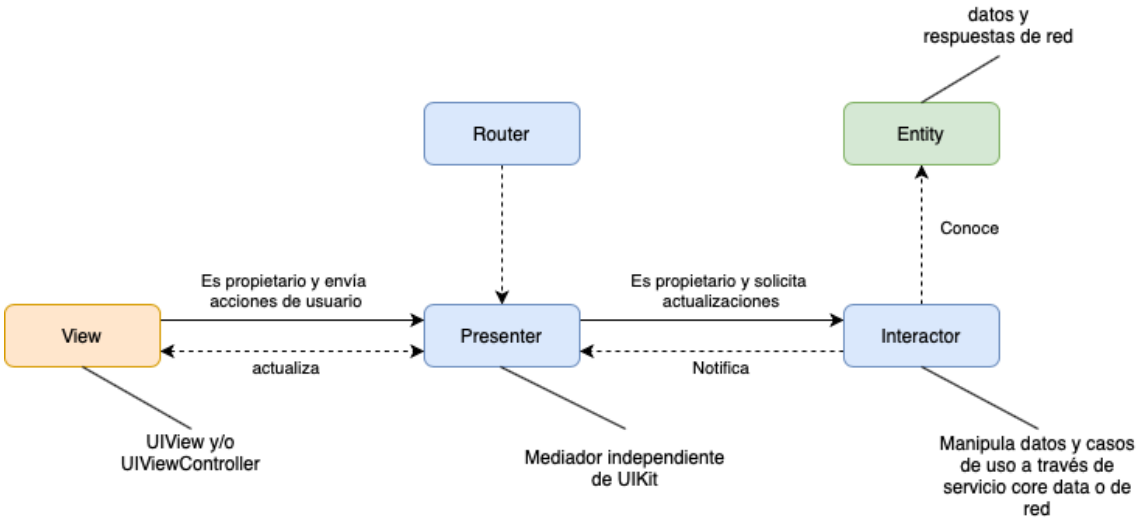


*Ilustración 12. Arquitectura VIPER*

En la aplicación, el modelo VIPER se ha implementado por funcionalidad, si bien algunas nomenclaturas han sido modificadas ligeramente. La siguiente figura muestra la funcionalidad "Login", que corresponde al Inicio de Sesión. En ella se puede ver un fichero llamado LoginDTO, el cual es un objeto de datos creado específicamente para la inyección de modelos de datos y de información en la funcionalidad. También se pueden ver los ficheros de Vista (LoginView.xib, LoginView_iPad.xib y LoginView.swift), el

Presenter (LoginPresenter.swift), el Interactor (LoginInteractor.swift), el Router (LoginRouter.swift), el Provider (MoodleProvider.swift), además de diferentes entidades utilizadas (LoginMode.swift, UserTokenMode.swift…).

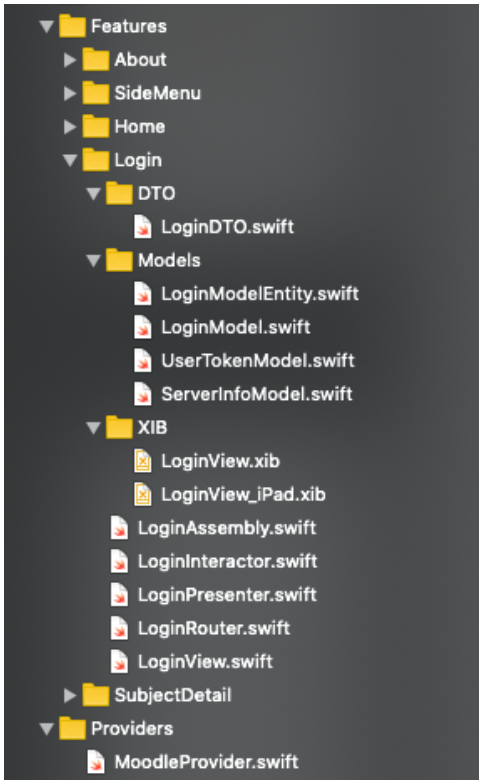


*Ilustración 13. Jerarquía VIPER en la aplicación*

El proyecto, además, ha sido configurado para soportar hasta cuatro entornos diferenciados para facilitar, en el futuro, el desarrollo y pruebas sin afectar a datos en producción. Los entornos son los siguientes:

- AulaVirtualLOC (*Local*): Entorno que podría ser usado para pruebas locales sin acceso a servicios web, pudiendo hacer uso de ficheros JSON creados al efecto para poder trabajar con datos estáticos.
- AulaVirtualDEV (*Development*): Entorno de desarrollo que haría uso de servicios web que también podrían estar en cambios por el desarrollo necesario en el backend. Aquí se realizaría la integración y las pruebas con los servicios web. Es en este entorno donde se ha llevado a cabo el desarrollo de este trabajo.

- AulaViertualPRE (*Pre-production*): Entorno cerrado a cambios pero que aún no utiliza datos de producción. Permite realizar pruebas de la integración previas a la puesta en producción de la aplicación o de los cambios llevados a cabo en el servidor Moodle.
- AulaVirtualPRO (*Production*): Entorno utilizado para la distribución final de la aplicación. Este entorno es el que utilizará el usuario, así como la aplicación publicada en la App Store.


*Ilustración 14. Entornos de desarrollo de la aplicación*

### *4.4.2 Comunicaciones y seguridad*

Las comunicaciones desde la aplicación con la plataforma Moodle se realiza mediante llamadas a servicios web tipo REST a través de un protocolo HTTPS el cual crea un canal seguro para transferencia de datos en formato JSON. Una vez realizado el proceso de inicio de sesión por parte del usuario, introduciendo nombre de usuario y contraseña, el servidor Moodle devuelve un código especial llamado *token* de sesión (o *login token*), el cual tiene un tiempo de validez y ayuda a proteger de algunas vulnerabilidades tales como el robo de la sesión de un usuario. A partir del inicio de sesión, cualquier petición de servicio al servidor Moodle incorporará dicho *token* para validar la sesión activa. Si este *token* no es válido o ha caducado, la sesión deberá iniciarse de nuevo.

Por otro lado, la aplicación ofrece al usuario la posibilidad de recordar las credenciales (nombre de usuario y contraseña), para evitar tener que introducirlos cada vez que el *token* haya terminado o la aplicación haya sido cerrada. Para el almacenamiento de dichos

elementos, se hará uso de una funcionalidad nativa de los dispositivos iOS llamada *Keychain Services* (Apple Inc., s.f.), la cual ofrece a las aplicaciones un mecanismo de almacenamiento para pequeñas cantidades de datos de usuario, tales como contraseñas o datos de tarjetas de crédito, en una base de datos encriptada llamada "llavero" ("*keychain*") de forma que, si algún usuario malintencionado obtuviese el teléfono del usuario legítimo, no podría acceder a la información de la contraseña almacenada sin antes poder desencriptar el llavero. Además, evita tener que implementar una lógica de encriptado en la aplicación, la cual no almacenará datos sensibles, y reduce cualquier posible error en la seguridad de dicho desarrollo.

Los elementos almacenados en el llavero son encriptados utilizando dos claves AES-256-GCM diferentes: una clave de tabla (metadatos) y una clave por fila (clave secreta). Los metadatos del llavero se cifran con la clave de metadatos para acelerar las búsquedas, mientras que el valor secreto se encripta con la clave secreta.

Además, los datos almacenados por una aplicación determinada sólo pueden ser leídos por aplicaciones firmadas por el mismo desarrollador, para evitar que otra aplicación con permisos de lectura en la base de datos SQLite encriptada pueda acceder a contraseñas almacenadas por una aplicación distinta.

Adicionalmente, en futuros desarrollos se podría aumentar esta seguridad incluyendo reconocimiento biométrico para acceder a la contraseña almacenada, como muestra el siguiente diagrama:

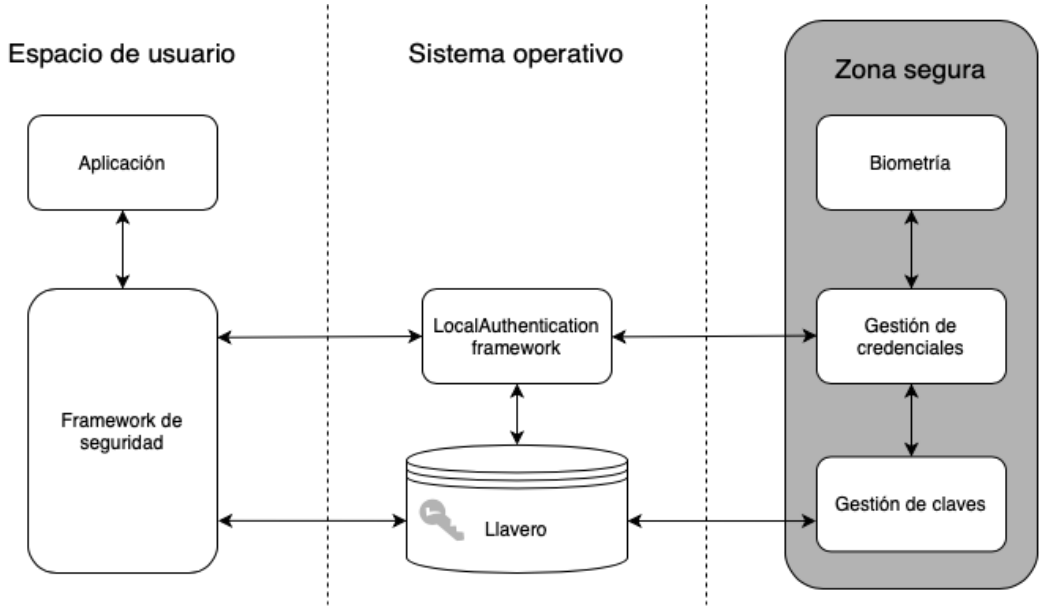


*Ilustración 15. Accesos al llavero*

### *4.4.3 Librerías externas*

Para facilitar el desarrollo y minimizar los riesgos de seguridad, se han hecho uso de algunas librerías externas. Estas librerías son integradas en el proyecto por un gestor de dependencias construido con Ruby llamado CocoaPods[1]. CocoaPods dispone de más de 71.000 librerías usadas en más de 3 millones de aplicaciones. A través de un fichero de configuración llamado "Podfile" es posible gestionar qué librerías externas se quiere utilizar, pudiendo actualizar a versiones más recientes de las mismas a través de sencillos comandos, o limitar la versión que se desea tener en la aplicación. La herramienta automáticamente descarga e integra las librerías seleccionadas en la aplicación, para que el desarrollador pueda hacer uso de estas.

[1] https://cocoapods.org

```
# Uncomment the next line to define a global platform for your project
platform :ios, '10.0'

def shared_pods
    use_frameworks!
    inhibit_all_warnings!

    pod 'SwiftLint'
    pod 'Alamofire', '~> 4.7'
    pod 'SwiftKeychainWrapper'
    pod 'SideMenu', '~> 6.0'
end
target 'AulaVirtualLOC' do
    workspace 'AulaVirtual'
    project 'AulaVirtual'
    shared_pods
end
target 'AulaVirtualDEV' do
    workspace 'AulaVirtual'
    project 'AulaVirtual'
    shared_pods
end
target 'AulaVirtualPRE' do
    workspace 'AulaVirtual'
    project 'AulaVirtual'
    shared_pods
end
target 'AulaVirtualPRO' do
    workspace 'AulaVirtual'
    project 'AulaVirtual'
    shared_pods
end
```

*Ilustración 16. Ejemplo de Podfile*

Las librerías usadas en la aplicación son las siguientes:

- SwiftLint[1]. Esta herramienta con licencia MIT, de Realm Inc, impone al desarrollador el uso de convenciones y estilos para el desarrollo de código Swift propuestos por la Guía de Estilo para Swift de Github (Github, 2017). SwiftLint ha sido integrado en tiempo de compilado en el entorno de desarrollo (AulaVirtualDEV), ofreciendo advertencias o, incluso, deteniendo la compilación si no se cumplen determinadas reglas de estilo o convenciones. De esta forma se fuerza al desarrollador a mantener un código homogéneo y limpio, facilitando su lectura y comprensión.
- Alamofire[2]: Librería con licencia MIT desarrollada en Swift por Matt Thompson y en propiedad de Alamofire Software Foundation para servir de intermediaria entre el desarrollador y las librerías de red de Apple, permitiendo una sintaxis más clara y sencilla.
- SwiftKeychainWrapper[3]: Librería con licencia MIT desarrollada por Jason Rendel para gestión del llavero de dispositivos iOS. Esta librería permite el

[1] https://github.com/realm/SwiftLint
[2] https://github.com/Alamofire/Alamofire
[3] https://github.com/jrendel/SwiftKeychainWrapper

almacenamiento y lectura de datos de usuario en el llavero del dispositivo de una forma más sencilla y clara.

- SideMenu[1]. Componente desarrollado por Jon Kent que integra un menú lateral en la aplicación, facilitando la navegación entre distintos controladores de vista independientes. Este componente, además, ofrece un alto nivel de configuración que facilitará la mayoría de los cambios que requiera la aplicación en el futuro.

### *4.4.4 Funcionalidades de valor añadido*

Multi-idioma:

De forma adicional se ha previsto, durante el diseño de la arquitectura, la posibilidad de adaptar la aplicación a distintos idiomas. La aplicación actualmente dispone de dos idiomas (inglés y español) para los textos que no dependan de los servicios ofrecidos por Moodle, tales como botones, alertas o textos. La elección de idioma es realizada automáticamente por la aplicación, en función del idioma del dispositivo. Si el idioma del dispositivo no estuviese contemplado por la aplicación, se aplicará el español por defecto.

El funcionamiento de la internacionalización (i18n, como es conocida en inglés) en el proyecto se realiza a través de unos ficheros con cadenas de texto, llamados Localizable.strings. Se crea uno por cada idioma en función de las necesidades. El formato estándar consta de uno o más pares clave-valor, cadenas de texto encerradas entre comillas dobles y separadas por un signo igual, como muestra el siguiente ejemplo:

```
/* -------- LOGIN -------- */
"LOGIN" = "Iniciar sesión";
"USERNAME" = "Nombre de usuario";
"PASSWORD" = "Contraseña";
```

[1] https://github.com/jonkykong/SideMenu

De esta forma se podrían añadir otros idiomas (catalán, gallego…) si las necesidades de la aplicación así lo requirieran en el futuro.

Extensiones y utilidades:

Se ha incluido un catálogo completo de utilidades y extensiones que el autor del presente trabajo ha creído que podrían aportar beneficios en el futuro basado en su experiencia en el desarrollo de aplicaciones. Estas utilidades y extensiones han sido desarrolladas por el autor en distintos proyectos según ha habido necesidad de estas y las ha ido manteniendo e incorporando a una colección personal. Son muchas las extensiones de clases y utilidades añadidas, pero algunas de ellas son las siguientes:

- Detección de dispositivo "pirateado" que pueda poner en riesgo la seguridad de la aplicación.
- Descarga de imágenes en red.
- Redimensionamiento de imágenes.
- Formateador de divisa.
- Trabajos con fechas.
- Trabajos con cadenas.

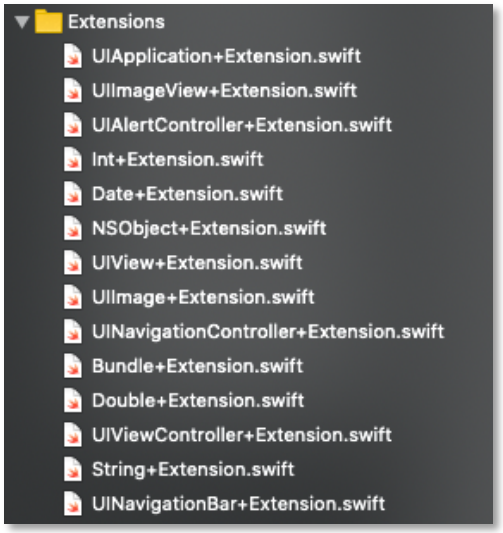


*Ilustración 17. Extensiones de clase*

## 4.5 Pruebas

Para validar el correcto funcionamiento de la aplicación en base a sus requisitos y casos de uso, se han elaborado una serie de casos de prueba (véase Anexo A3). Como muestra la siguiente matriz de compatibilidad, estos casos de prueba cubrirían todos los casos de uso identificados:

*Tabla 4. Matriz de trazabilidad de casos de uso*

| | CU01 | CU02 | CU03 | CU04 | CU05 | CU06 | CU07 |
|---|---|---|---|---|---|---|---|
| CP01 | X | | | | | | |
| CP02 | | X | | | | | |
| CP03 | | X | | | | | |
| CP04 | | | X | | | | |
| CP05 | | | | X | | | |
| CP06 | | | | | X | | |
| CP07 | | | | | | X | |
| CP08 | | | | | | | X |
| CP09 | | | | | | | X |

Cabe mencionar que el caso de prueba CP01 no podrá ser validado de forma real, al depender la publicación de la aplicación de una validación previa por parte de Apple. Para poder acceder a la aplicación y realizar las pruebas se hará uso de un terminal iPhone donde se instalará y ejecutará la aplicación a través de Xcode.

Para poder llevar a cabo la evaluación de los anteriormente mencionados casos de prueba, se parte de la gestión del entorno Moodle habilitado al efecto, creando un usuario lógico que simularía a un estudiante, el cual dispone de las siguientes características:

- Nombre del estudiante: Sinesio Sánchez
- Nombre de usuario: ssanchez
- Contraseña: Prueba.2020
- Asignaturas matriculadas:
    - Inteligencia Artificial. Dispone de tareas, algunas evaluadas y algunas con comentarios del profesor.

- Ingeniería del Software I. Dispone de tareas, algunas evaluadas y algunas con comentarios del profesor.
- Patrones de diseño y frameworks. Dispone de tareas, algunas evaluadas y algunas con comentarios del profesor.

# CAPÍTULO 5: EVALUACIÓN

Para la evaluación del proyecto software llevado a cabo se han tomado como referencia los casos de pruebas mencionados en el epígrafe “4.5 Pruebas” y que se detallan en el Anexo A3. De esta forma se espera comprobar que la aplicación cumple con los objetivos marcados. Los siguientes epígrafes del capítulo incluyen una breve descripción de la prueba y evidencias gráficas del funcionamiento de la aplicación que pueden ser contrastadas con los casos de prueba mencionados.

## 5.1 Acceso a la aplicación

Una vez instalada la aplicación, la misma es accesible a través del icono “Aula Virtual”. La primera vez que se accede, se muestra la pantalla de inicio de sesión.


*Ilustración 18. Acceso a la aplicación*

## 5.2 Inicio de sesión

La siguiente ilustración muestra tanto un inicio de sesión con las credenciales correctas como un acceso con credenciales incorrectas:


*Ilustración 19. Inicio de sesión*

## 5.3 Consulta de asignaturas

Una vez iniciada la sesión correctamente, la aplicación solicita al servidor Moodle el listado de asignaturas para el usuario activo. Además de la información de las asignaturas, se recuperan detalles del propio usuario como el nombre, los apellidos y la foto si éste la tuviese configurada y visible.

Para actualizar la información mostrada en la pantalla, el usuario puede hacer uso del patrón "*Pull to Refresh*" (o "tirar para actualizar"). Este patrón permite utilizar un gesto que se ha convertido en un estándar (Babich, 2016) en muchas aplicaciones y el cual consiste en arrastrar hacia abajo el contenido de la pantalla para actualizar el contenido de esta.

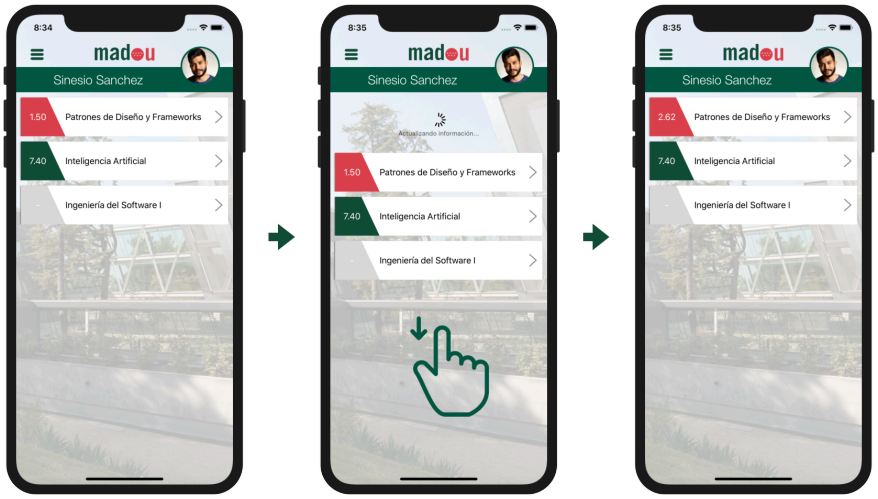


*Ilustración 20. Actualización de información de asignaturas*

Nótese como, si el valor numérico de la calificación de la asignatura es inferior a 5, el distintivo muestra un color rojo; en el caso de ser 5 o superior, el distintivo es verde. Si aún no existe evaluación o ésta no es numérica, el distintivo es gris.

## 5.4 Consulta de tareas

Al seleccionar una asignatura, la aplicación navega a otra pantalla donde se pueden ver las tareas de la asignatura, tanto evaluadas como sin evaluar.

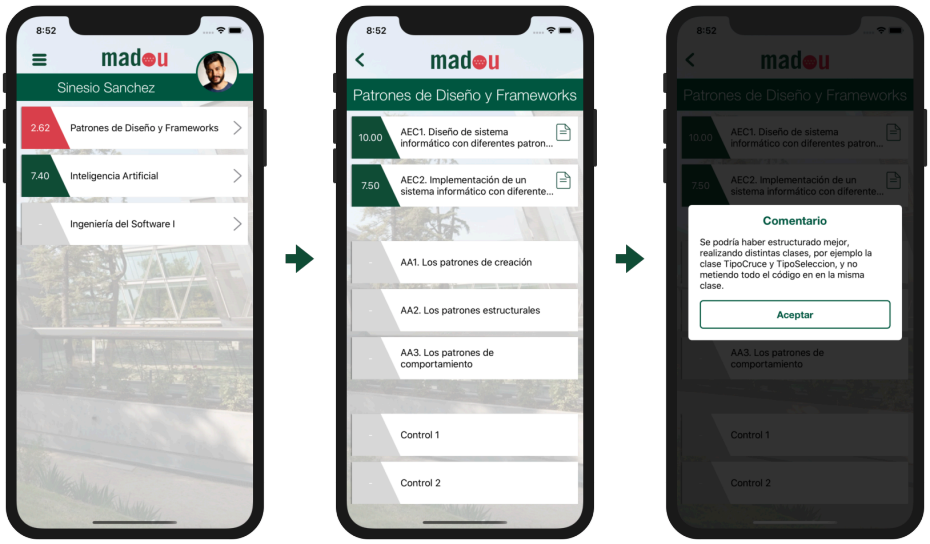


*Ilustración 21. Consulta de tareas*

Además, las tareas con comentario del profesor presentan un icono distintivo que permite visualizar dicho comentario. Las tareas presentan los mismos distintivos de color en función de la calificación que se han mencionado en el epígrafe “5.3 Consulta de asignaturas”.

## 5.5 Cierre de sesión

Para poder cerrar la sesión actual, el usuario debe hacerlo a través del menú situado en la esquina superior izquierda de las pantallas principales, con un icono del estilo “hamburguesa” (Wikipedia, 2020). Tras pulsar el botón de “Cerrar sesión”, una alerta pide confirmación al usuario para cerrar la sesión actual.

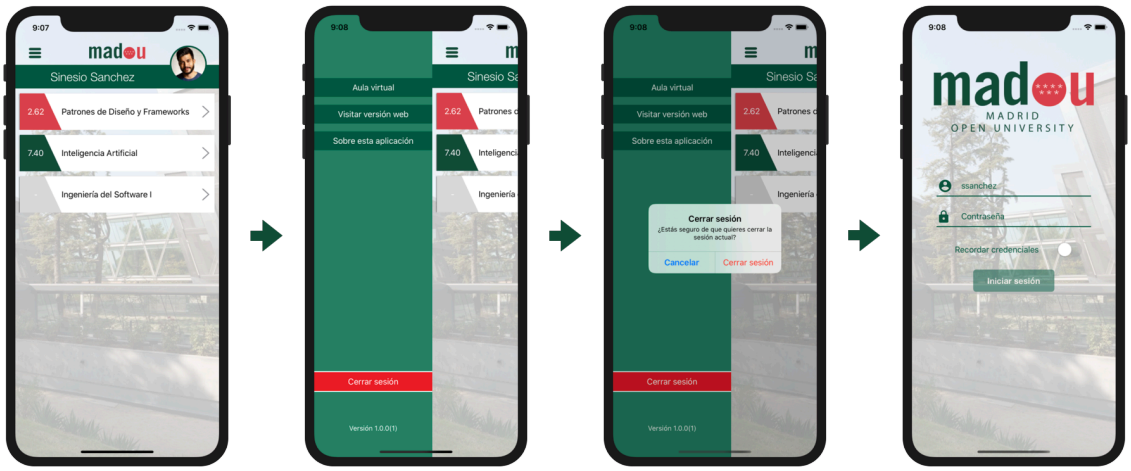


*Ilustración 22. Cierre de sesión*

## 5.6 Segunda apertura y posteriores

Durante el inicio de sesión, el usuario tiene la posibilidad de que la aplicación recuerde sus credenciales, almacenándolas de forma segura en el dispositivo. De esta forma si cierra la aplicación con la sesión iniciada, cuando vuelva a acceder a la misma, ésta no solicitará las credenciales y navegará directamente a la pantalla de asignaturas. Si, por el contrario, el usuario decide no guardar las credenciales, se le volverán a solicitar al abrir de nuevo la aplicación.


*Ilustración 23. Función "Recordar credenciales"*

## 5.7 Menú lateral

El menú lateral permite una navegación sencilla por la aplicación, facilitando la escalabilidad de esta pudiendo añadir nuevos elementos a medida que la aplicación lo requiera.

Actualmente el menú permite las siguientes opciones:

- Aula virtual: Permite acceder a las asignaturas matriculadas y sus correspondientes tareas.
- Visitar versión web: Abre el aula virtual Moodle de la aplicación en un navegador, permitiendo al usuario acceder a funcionales que aún no hayan sido implementadas en la aplicación.

- Sobre esta aplicación: Ofrece información sobre la aplicación, el autor y componentes de terceros utilizados.
- Cerrar sesión: permite al usuario cerrar la sesión actual. Si el usuario hubiese marcado la opción de recordar credenciales, se mantendrá el nombre de usuario en la pantalla de inicio de sesión, eliminando la contraseña para evitar un inicio de sesión automático tras haber cerrado sesión.
- Información de versión y número de compilación.

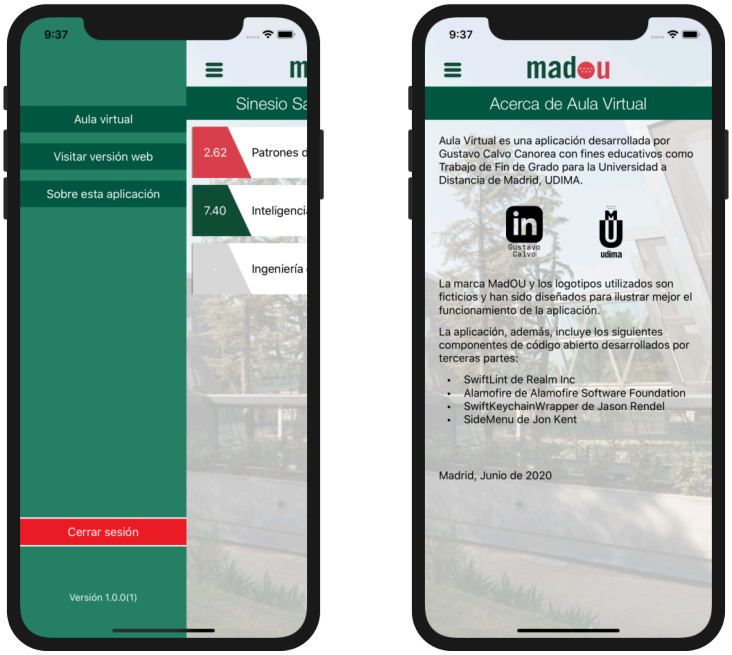


*Ilustración 24. Detalle de menú lateral y pantalla de información de la aplicación*

## 5.8 Multi-idioma

La aplicación dispone de soporte a distintos idiomas, pudiendo ser incluidos nuevos idiomas en el futuro. Actualmente la aplicación se encuentra en español e inglés, tomando el idioma configurado en el dispositivo móvil como referencia. En el caso de que el idioma del dispositivo móvil esté en un idioma no contemplado por la aplicación, ésta se basará en el orden de prioridad para idiomas secundarios configurados por el usuario en

el teléfono, siendo el inglés el idioma por defecto. La siguiente ilustración muestra la comparativa de algunas pantallas en inglés y en español:

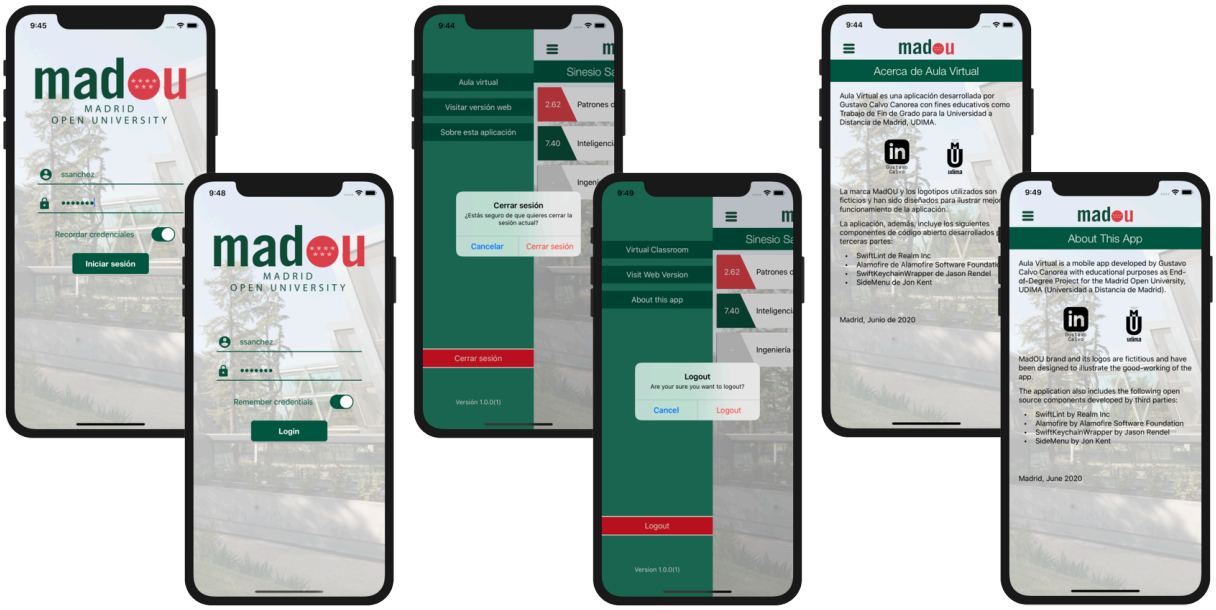


*Ilustración 25. Multi-idioma*

## 5.9 Gestión de errores

La arquitectura de las comunicaciones de la aplicación permite informar al usuario de los posibles errores que puede ofrecer la aplicación, pudiendo configurar el mensaje de errores específicos o mostrando la descripción genérica de los distintos errores si estos no están contemplados de forma específica. Por ejemplo, si la aplicación no dispusiese de conexión a Internet en el momento de hacer una petición de red, se mostrará un error genérico donde informa que no se ha podido completar la operación y, entre paréntesis, el tipo de error de red recibido y el código del mismo. En el caso concreto de credenciales incorrectas durante el inicio de sesión, sí que se encuentra identificado el error, por lo que el mensaje es más específico. Ambos ejemplos se pueden ver en la siguiente ilustración:

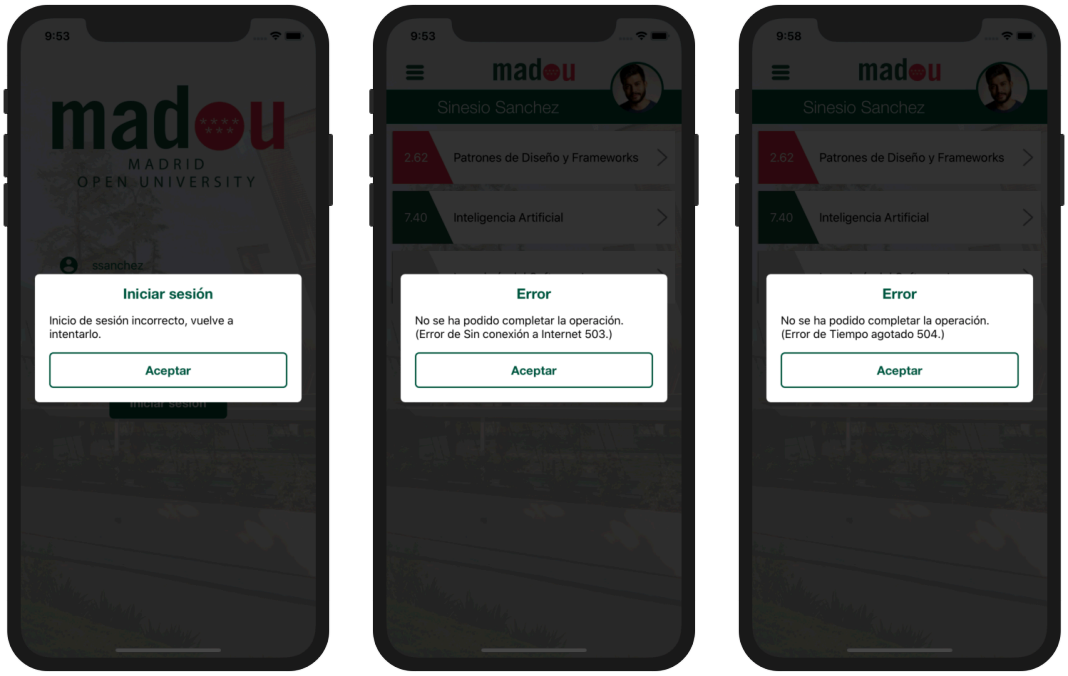


*Ilustración 26. Gestión de errores*

# CAPÍTULO 6: CONCLUSIONES

Al inicio del presente proyecto, se había planteado como objetivo el desarrollo de una aplicación móvil que permitiese a un estudiante acceder a un entorno educativo que utilice Moodle. Esta aplicación debía consultar sus asignaturas matriculadas, notas provisionales y/o definitivas, tareas y calificaciones de estas y comentarios del profesor si los hubiera. Estos objetivos se tradujeron en una serie de requisitos funcionales, sus correspondientes casos de uso y sus casos de prueba.

La aplicación desarrollada cumple con los objetivos planteados inicialmente, como se ha evidenciado en el Capítulo 5: Evaluación. El propio desarrollo de la aplicación ha supuesto aprender a utilizar herramientas específicas no vistas en asignaturas de la carrera, tales como:

- Xcode. La interfaz de desarrollo utilizada para el desarrollo de dispositivos iOS es muy completa y, en general, intuitiva. Sin embargo, para determinadas funcionalidades, tales como el uso de diferentes entornos o la incorporación de repositorios externos (CocoaPods), ha sido necesaria una investigación más profunda del entorno.
- Sketch. Con esta aplicación se ha diseñado la interfaz gráfica de la aplicación, así como la imagen de marca, permitiendo exportar los recursos gráficos a formatos compatibles con Xcode adaptables a los distintos tipos de pantalla. Se ha intentado diseñar y utilizar únicamente imágenes vectoriales para facilitar el escalado de las mismas y reducir el espacio ocupado en el dispositivo. No ha sido posible hacerlo con todas, pues se hace uso de fotos como fondo de pantalla que no están vectorizadas. El diseño tanto de la experiencia de usuario (UX) como de la interfaz de usuario (UI) ha sido realizado desde cero basando su diseño en aplicaciones similares y en las pautas propuestas por distintos fabricantes.
- Swift. El lenguaje multipropósito utilizado para el desarrollo de aplicaciones iOS que, a diferencia de Objective-C (el lenguaje utilizado tradicionalmente), es más limpio visualmente, más fácil de mantener, más estable y de código abierto.

- Moodle. Si bien, como estudiante, estoy familiarizado con el entorno Moodle, la administración de este y la gestión desde el punto de vista del profesor ha supuesto un reto de aprendizaje. Otro reto que ha supuesto el uso de Moodle es la comunicación a través de los servicios web, cuya documentación de API REST no ha sido fácil de entender e implementar.

Además de lo expuesto anteriormente, me impuse unos retos adicionales para la implementación de la aplicación:

- Arquitectura. VIPER es una arquitectura avanzada que permite un crecimiento modular y muy desacoplado de los componentes de la aplicación. Su implementación ha supuesto un reto importante debido a sus cinco capas lógicas. Al principio, algunas de las capas pueden parecer no tener mucho sentido, pero la experiencia posterior cuando ha habido que hacer algún cambio ha demostrado una mayor sencillez en el proceso de la modificación, al existir un bajo acoplamiento entre capas.
- Seguridad. Desde el principio se planteó la necesidad de realizar las comunicaciones web de forma segura a través de los canales seguros que ofrece Moodle. Sin embargo, las credenciales en el dispositivo inicialmente se iban a almacenar como propiedades de la aplicación. Esta solución no era del todo segura, por lo que hubo que implementar el almacenamiento de dichas credenciales a través del "llavero" encriptado que ofrece iOS a los desarrolladores.
- Escalabilidad. Desde la concepción de la idea de la aplicación, se tuvo en cuenta la necesidad de que ésta fuera escalable, pudiendo añadir nuevas funcionalidades fácilmente. Es por ello por lo que se optó por VIPER como arquitectura. Pero, además, se incluyó la posibilidad de traducir la aplicación a distintos idiomas sin necesidad de cambiar el código de la aplicación, únicamente incluyendo un fichero con un formato específico con el nuevo idioma. La inclusión de un menú lateral también permite añadir nuevas funcionalidades a la aplicación de una forma relativamente sencilla.

La finalización de este proyecto me ha permitido poner a prueba los conocimientos adquiridos durante la carrera y la capacidad de adaptación y aprendizaje desarrollados durante la misma. El desarrollo de aplicaciones móviles nativas me ha resultado bastante interesante y me ha dado ideas para posibles nuevas aplicaciones móviles. Es probable que entre mis siguientes pasos de aprendizaje se encuentre el aprender a desarrollar aplicaciones para otras plataformas como Android.

## 6.1 Continuidad y trabajo futuro

La aplicación ha sido desarrollada de forma que, con un sencillo cambio en el fichero de constantes, se pueda indicar una dirección web a otra plataforma Moodle. Mi propuesta es que se permita la apertura del acceso a la API de la Universidad a Distancia de Madrid, para poder conectar la aplicación a la misma y que los estudiantes puedan probar el acceso y la información mostrada, así como poder acceder de una forma cómoda a sus asignaturas y tareas. Una sustitución de recursos gráficos en la aplicación bastaría para pasar de la marca MadOU a UDIMA.

Esta apertura de la API también facilitaría el desarrollo de una aplicación Android que permita a más usuarios acceder a la plataforma.

Las nuevas funcionalidades propuestas para el futuro podrían ser las siguientes:

- Uso de información biométrica para inicio de sesión, a través de cámara o huella dactilar.
- Añadir formas de contacto con los profesores, pudiendo conocer su extensión telefónica o correo electrónico entre otros para poder contactar con el profesor en cuestión.
- Posibilidad de visualizar foros o recursos, e interactuar con ellos.
- Ver el estado de las entregas.
- Poder cargar o descargar documentos desde la aplicación.
- Acceso a Secretaría Virtual o web de elección de exámenes. Este desarrollo requeriría implementar nuevas formas de comunicación con las distintas APIs de

las plataformas mencionadas. De nuevo, la arquitectura VIPER utilizada permite una implementación más sencilla al poder utilizar parte del desarrollo existente.

- Incluir lenguas cooficiales, facilitando la accesibilidad.
- Adaptación a *tablet*. Actualmente la aplicación se puede utilizar en una *tablet* de tipo iPad, y el contenido se adapta, pero no aprovecha el potencial de una superficie de pantalla más grande. En el planteamiento inicial de la arquitectura se tuvo en cuenta la inclusión de ficheros de vista específicos para iPad, de forma que se facilitase su uso en el futuro.

## BIBLIOGRAFÍA


Apple Inc. (5 de Marzo de 2012). *Apple's App Store Downloads Top 25 Billion*. Recuperado el 5 de Abril de 2020, de https://www.apple.com/newsroom/2012/03/05Apples-App-Store-Downloads-Top-25-Billion/

Apple Inc. (2020). *Swift.org*. Recuperado el 5 de Abril de 2020, de https://swift.org/about/

Apple Inc. (s.f.). *Using the Keychain to Manage User Secrets*. Recuperado el Abril de 2020, de https://developer.apple.com/documentation/security/keychain_services/keychain_items/using_the_keychain_to_manage_user_secrets

Babich, N. (25 de Octubre de 2016). *Pull to Refresh UI Pattern*. Recuperado el 2 de Junio de 2020, de https://uxplanet.org/pull-to-refresh-ui-pattern-42a85f671cdf

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., . . . Mellor, S. S. (2001). *Manifesto for Agile Software Development*. Obtenido de https://agilemanifesto.org

Chamilo. (2020). *Chamilo.org*. Recuperado el 5 de Abril de 2020, de Chamilo.org – Asociación Chamilo: https://chamilo.org/

edX Inc. (2020). *edX | Free Online Courses by Harvard, MIT, & more*. Recuperado el 5 de 4 de 2020, de https://www.edx.org/

Github. (21 de Agosto de 2017). *Guía de estilo y convenciones de programacón para proyectos en Swift.* Recuperado el 3 de Junio de 2020, de Github: https://github.com/sejas/swift-style-guide/blob/spanish/README-ES.md

King, R. (11 de Septiembre de 2012). *Facebook's Mark Zuckerberg knocks HTML5 in favor of native apps*. (ZDNet) Recuperado el 5 de Abril de 2020, de https://www.zdnet.com/article/facebooks-mark-zuckerberg-knocks-html5-in-favor-of-native-apps/

Magda, A., & Aslanian, C. (2018). *Online college students 2018: Comprehensive data on demands and preferences.* Recuperado el 5 de Abril de 2020, de https://49hk843qjpwu3gfmw73ngy1k-wpengine.netdna-ssl.com/wp-content/uploads/2018/06/OCS-2018-Report-FINAL.pdf

Martin, R. C. (2009). *Clean Code: A Handbook of Agile Software Craftsmanship.* Crawfordsville, In, Estados Unidos: Prentice Hall.

Martin, R. C. (2018). *Clean Architecture: a craftsmans guide to software structure and design.* Londres, Reino Unido: Prentice Hall.

Moodle. (2020). *Moodle App - Mobile Learning on iOS, Android & PC* . Recuperado el 5 de Abril de 2020, de https://moodle.com/app/

Moodle. (2020). *Moodle Statistics*. Recuperado el 5 de Abril de 2020, de https://stats.moodle.org

Moodle Pty Ltd. (2020). *MoodleCloud - Moodle hosting from the people that make Moodle.* Recuperado el 5 de Abril de 2020, de https://moodlecloud.com

Mutual Mobile. (24 de Septiembre de 2014). *Meet VIPER: Mutual Mobile's application of Clean Architecture for iOS apps*. Recuperado el Abril de 2020, de https://mutualmobile.com/resources/meet-viper-fast-agile-non-lethal-ios-architecture-framework

openSWAD. (2020). *OpenSWAD: plataforma educativa*. Recuperado el 5 de Abril de 2020, de https://openswad.org/es

Pathak, P. (18 de Noviembre de 2018). *An Overview of Architectural Design Patterns for iOS Developers*. Recuperado el 3 de Junio de 2020, de dzone.com: https://dzone.com/articles/an-overview-of-architectural-design-patterns-for-i

StatCounter. (Abril de 2020). *Mobile Operating System Market Share Worldwide | StatCounter Global Stats*. Recuperado el 5 de Abril de 2020, de https://gs.statcounter.com/os-market-share/mobile/worldwide

Universidad Complutense Madrid. (s.f.). *Campus Virtual Universidad Complutense Madrid*. Recuperado el 5 de Abril de 2020, de https://cv4.ucm.es/moodle

Universidad de Granada. (s.f.). *Plataforma de Recursos de Apoyo a la Docencia*. Recuperado el 5 de Abril de 2020, de https://prado.ugr.es

University of Glasgow. (s.f.). *University of Glasgow Moodle*. Recuperado el 5 de Abril de 2020, de https://moodle.gla.ac.uk/

Wikipedia. (Abril de 2020). *Android*. Recuperado el 5 de Abril de 2020, de https://es.wikipedia.org/wiki/Android

Wikipedia. (Abril de 2020). *BlackBerry*. Recuperado el 2020 de Abril de 5, de https://es.wikipedia.org/wiki/BlackBerry

Wikipedia. (28 de Mayo de 2020). *Hamburger button*. Recuperado el 2 de Junio de 2020, de Wikipedia: https://en.wikipedia.org/wiki/Hamburger_button

Wikipedia. (4 de Abril de 2020). *IBM Simon*. Recuperado el 5 de Abril de 2020, de https://en.wikipedia.org/wiki/IBM_Simon

Wikipedia. (5 de Abril de 2020). *iPhone*. Recuperado el 5 de Abril de 2020, de https://es.wikipedia.org/wiki/IPhone

Wikipedia. (1 de Abril de 2020). *Protocolo de aplicaciones inalámbricas*. Recuperado el 5 de Abril de 2020, de https://es.wikipedia.org/wiki/Protocolo_de_aplicaciones_inalámbricas

# ANEXOS

Anexo A1. Casos de uso

| CU01 - Descarga de la aplicación | | |
|---|---|---|
| Personal involucrado y objetivo | El **estudiante** descarga la aplicación desde la tienda de aplicaciones de la plataforma iOS. | |
| Precondiciones | • La aplicación debe haber sido publicada previamente y ser visible para los usuarios de smartphone de las plataformas iOS.<br>• El estudiante debe disponer de un terminal iOS con los requisitos mínimos necesarios para hacer uso de la aplicación, así como una cuenta con el proveedor para realizar su descarga. | |
| Garantías de éxito | • La aplicación ha sido descargada y es posible ejecutarla en el terminal del estudiante. | |
| Flujo básico | **Paso** | **Acción** |
| | 1 | El estudiante accede a la tienda de aplicaciones de su plataforma (App Store) con sus credenciales. |
| | 2 | El estudiante descarga la aplicación. |
| | 3 | El estudiante ejecuta la aplicación, mostrándose una pantalla en la que identificarse como estudiante de Madrid Open University |

| CU02 - Inicio de sesión | | |
|---|---|---|
| Personal involucrado y objetivo | El **estudiante** introduce sus credenciales en la pantalla de inicio de sesión, pudiendo elegir si quiere mantener iniciada la sesión para futuras ejecuciones de la aplicación. | |
| Precondiciones | • El estudiante dispone de una forma de identificarse en el portal (usuario y contraseña) | |
| Garantías de éxito | • Si las credenciales son correctas se accederá a una pantalla en la que se muestran las asignaturas matriculadas si las tuviera.<br>• Si las credenciales no son correctas se informa de ello al usuario, manteniéndose la aplicación en la pantalla de inicio de sesión. | |
| Flujo básico | **Paso** | **Acción** |
| | 1 | El estudiante introduce su usuario y contraseña. |
| | 2 | El estudiante decide si quiere mantener su sesión iniciada (opcional, por defecto no se mantiene iniciada). |
| | 3.1 | Si las credenciales son correctas, la aplicación muestra otra pantalla donde se identifican las asignaturas matriculadas si estuvieran disponibles. |
| | 3.2 | Si las credenciales no son correctas, se informa al usuario de ello y la aplicación se mantiene en la pantalla de inicio de sesión. |

| CU03 - Consulta de asignaturas | | |
|---|---|---|
| Personal involucrado y objetivo | El **Estudiante** visualiza las asignaturas y, si estuviera disponible, su calificación parcial/final. | |
| Precondiciones | • El estudiante ha iniciado correctamente sesión (CU04).<br>• El estudiante dispone de asignaturas visibles | |
| Garantías de éxito | • Se muestra el listado de asignaturas disponibles para el usuario (nombre y nota parcial/final) | |
| Flujo básico | **Paso** | **Acción** |
| | 1 | El estudiante inicia sesión o el sistema ha mantenido las credenciales |
| | 2 | Se muestra una pantalla con el listado de asignaturas |

| CU04 - Consulta de tareas | | |
|---|---|---|
| Personal involucrado y objetivo | El **Estudiante** visualiza las tareas asociadas a una determinada asignatura, su calificación (si estuviera disponible) y los comentarios del profesor (si los hubiera). | |
| Precondiciones | • El estudiante ha iniciado correctamente sesión (CU04).<br>• El estudiante dispone de asignaturas visibles (CU05).<br>• Al menos una de las asignaturas del estudiante dispone de tareas visibles. | |
| Garantías de éxito | • Se muestra el listado de tareas visibles en la asignatura seleccionada.<br>• (Opcional) Si la tarea estuviese calificada, se mostraría la calificación de esta.<br>• (Opcional) Si la tarea estuviese calificada, se mostraría la calificación de esta. | |
| Flujo básico | **Paso** | **Acción** |
| | 1 | El estudiante inicia sesión o el sistema ha mantenido las credenciales |
| | 2 | El estudiante selecciona una asignatura |
| | 3 | Se muestra el listado de tareas disponibles y su nota (si estuviera calificada) |
| | 4 | (Opcional) El estudiante pulsa sobre el icono de comentario del profesor y se muestra una ventana emergente con el comentario del profesor en dicha tarea |

| CU05 - Refresco de información | | |
|---|---|---|
| Personal involucrado y objetivo | El **Estudiante** puede actualizar la información presentada en la pantalla de asignaturas mediante un gesto táctil. | |
| Precondiciones | • El estudiante ha iniciado correctamente sesión (CU04).<br>• El estudiante dispone de asignaturas visibles (CU05). | |
| Garantías de éxito | • Se muestra el contenido de la pantalla con la información actualizada. | |
| Flujo básico | **Paso** | **Acción** |
| | 1 | El estudiante inicia sesión o el sistema ha mantenido las credenciales |

| | | |
|---|---|---|
| | 2 | El estudiante fuerza una actualización del contenido mediante un gesto táctil |
| | 3 | Si hubiera algún cambio de la información mostrada, la pantalla actualizará su contenido para mostrar la información nueva |

| CU06 - Cierre de sesión | | |
|---|---|---|
| Personal involucrado y objetivo | El **Estudiante** decide cerrar la sesión. | |
| Precondiciones | • El estudiante ha iniciado correctamente sesión (CU04). | |
| Garantías de éxito | • Se muestra la pantalla de inicio de sesión con los campos para las credenciales vacíos, independientemente de si decidió guardar las credenciales o no.<br>• Si el usuario decidió mantener la sesión iniciada cuando ingresó sus credenciales por última vez, las credenciales dejan de ser persistidas por la aplicación, siendo necesario volver a introducirlas de nuevo para iniciar sesión. | |
| Flujo básico | **Paso** | **Acción** |
| | 1 | Con una sesión iniciada, el usuario selecciona la opción “Cerrar sesión. |
| | 2 | Se pide la confirmación al usuario |
| | 3.1 | El usuario confirma el cierre de sesión. Se muestra la pantalla de inicio de sesión. |
| | 3.2 | El usuario cancela el cierre de sesión. La aplicación se mantiene en la vista que estaba. |

| CU07 - Apertura de la aplicación por segunda vez y posteriores | | |
|---|---|---|
| Personal involucrado y objetivo | El **Estudiante**, tras haber iniciado sesión al menos una vez previamente y haber cerrado la aplicación, vuelve a abrir la aplicación. La aplicación debe comportarse acorde a la elección del usuario acerca de mantener la sesión iniciada (CU04) | |
| Precondiciones | • El estudiante ha iniciado correctamente sesión (CU04).<br>• El usuario ha cerrado la aplicación. | |
| Garantías de éxito | • Si el usuario seleccionó “Mantener la sesión iniciada”, se mostrará la pantalla de asignaturas.<br>• Si el usuario no seleccionó “Mantener la sesión iniciada”, se mostrará la pantalla de inicio de sesión. | |
| Flujo básico | **Paso** | **Acción** |
| | 1 | Con la aplicación cerrada previamente, se pulsa en el icono de la aplicación. |
| | 2.1 | Si el usuario seleccionó “Mantener la sesión iniciada”, se muestra la pantalla de asignaturas. |
| | 2.2 | Si el usuario no seleccionó “Mantener la sesión iniciada”, se muestra la pantalla de inicio de sesión. |

Anexo A2. Requisitos funcionales

| **RQF01 - Descarga de la aplicación** | |
|---|---|
| **Dependencia** | Ninguna |
| **Descripción** | El usuario debe poder descargar a través de la tienda de aplicaciones relativa a su dispositivo (App Store para iOS) |
| **Comentarios** | Las tiendas de aplicaciones podrían imponer condiciones o requisitos ajenos a la aplicación. |

| **RQF02 - Inicio de sesión** | |
|---|---|
| **Dependencia** | RQF01 |
| **Descripción** | La aplicación solicita al usuario un nombre de usuario y una contraseña para acceder al contenido de la aplicación. El usuario, adicionalmente, puede decidir mantener la sesión iniciada.<br>Si las credenciales son correctas, la aplicación muestra otra pantalla donde se identifican las asignaturas matriculadas si estuvieran disponibles.<br>Si las credenciales no son correctas, se informa al usuario de ello y la aplicación se mantiene en la pantalla de inicio de sesión. |
| **Comentarios** | Es necesario que el usuario disponga de una cuenta activa en la plataforma Moodle de Madrid Open University. |

| RQF03 - Consulta de asignaturas | |
|---|---|
| **Dependencia** | RQF02 |
| **Descripción** | El usuario visualiza las asignaturas y, si estuviera disponible, su calificación parcial/final. |
| **Comentarios** | Es necesario que el usuario disponga de asignaturas visibles en su aula virtual. |

| RQF04 - Consulta de tareas | |
|---|---|
| **Dependencia** | RQF03 |
| **Descripción** | El usuario visualiza las tareas asociadas a una determinada asignatura, su calificación (si estuviera disponible) y los comentarios del profesor (si los hubiera). |
| **Comentarios** | Es necesario que el usuario disponga de asignaturas visibles.<br>Para visualizar el comentario del profesor, es necesario que el profesor haya realizado un comentario sobre alguna entrega.<br>Para visualizar las notas es necesario que el profesor haya calificado la tarea. |

| RQF05 - Refresco de información | |
|---|---|
| **Dependencia** | RQF03 |
| **Descripción** | El usuario puede actualizar la información presentada en la pantalla de asignaturas mediante un gesto táctil. |
| **Comentarios** | Es necesario que el usuario disponga de asignaturas visibles en su aula virtual. |

| RQF06 - Cierre de sesión | |
|---|---|
| **Dependencia** | RQF02 |
| **Descripción** | El usuario puede cerrar la sesión iniciada, mostrándose de nuevo la pantalla de inicio de sesión. |
| **Comentarios** | • Se muestra la pantalla de inicio de sesión con los campos para las credenciales vacíos, independientemente de si decidió guardar las credenciales o no.<br>• Si el usuario decidió mantener la sesión iniciada cuando ingresó sus credenciales por última vez, las credenciales dejan de ser persistidas por la aplicación, siendo necesario volver a introducirlas de nuevo para iniciar sesión. |

| RQF07 - Apertura de la aplicación por segunda vez y posteriores | |
|---|---|
| **Dependencia** | RQF02 |
| **Descripción** | El usuario, tras haber iniciado sesión al menos una vez previamente y haber cerrado la aplicación, vuelve a abrir la aplicación. La aplicación debe comportarse acorde a la elección del usuario acerca de mantener la sesión iniciada. |
| **Comentarios** | • Si el usuario seleccionó “Mantener la sesión iniciada”, se mostrará la pantalla de asignaturas.<br>• Si el usuario no seleccionó “Mantener la sesión iniciada”, se mostrará la pantalla de inicio de sesión. |

Anexo A3. Casos de prueba

| Identificador caso de prueba | CP01 |
|---|---|
| **Caso de uso probado** | CU01 |
| **Descripción** | Descarga de la aplicación |
| **Propósito** | Comprobar que es posible descargar la aplicación desde App Store |
| **Acciones** | Un usuario con cuenta de App Store de Apple busca la aplicación Aula Virtual y la descarga. |
| **Datos de entrada** | Usuario App Store: usuario_ficticio@me.com<br>Contraseña App Store: Contraseña1234 |
| **Salida esperada** | OK → La instalación se ha realizado correctamente |
| **Salida obtenida** | OK → La instalación se ha realizado correctamente |
| **Resultado** | Éxito |
| **Información adicional** | Al depender de un sistema externo como es la App Store de Apple, únicamente es posible probar la correcta instalación de la aplicación haciendo uso de Xcode y un terminal iPhone validado para pruebas. |

| Identificador caso de prueba | CP02 |
|---|---|
| **Caso de uso probado** | CU02 |
| **Descripción** | Inicio de sesión correcto |
| **Propósito** | Comprobar que la aplicación valida las credenciales correctamente. |
| **Acciones** | Tras la descarga de la aplicación, se lanza la misma y se realiza un intento de inicio de sesión. |
| **Datos de entrada** | nombre de usuario = "ssanchez"<br>password = "Prueba.2020" |
| **Salida esperada** | OK → El inicio de sesión es correcto y se muestra la pantalla de asignaturas del usuario. |
| **Salida obtenida** | OK → El inicio de sesión es correcto y se muestra la pantalla de asignaturas del usuario. |
| **Resultado** | Éxito |
| **Información adicional** | |

| Identificador caso de prueba | CP03 |
|---|---|
| **Caso de uso probado** | CU02 |
| **Descripción** | Inicio de sesión incorrecto |
| **Propósito** | Comprobar que la aplicación valida las credenciales correctamente. |
| **Acciones** | Tras la descarga de la aplicación, se lanza la misma y se realiza un intento de inicio de sesión. |
| **Datos de entrada** | nombre de usuario = "ssanchez"<br>password = "Prueba.2018" |
| **Salida esperada** | KO → El inicio de sesión no es correcto y se muestra una alerta que informa de ello al usuario, manteniéndose en la pantalla de inicio de sesión. |
| **Salida obtenida** | KO → El inicio de sesión no es correcto y se muestra una alerta que informa de ello al usuario, manteniéndose en la pantalla de inicio de sesión. |
| **Resultado** | Éxito |
| **Información adicional** | |

| Identificador caso de prueba | CP04 |
|---|---|
| **Caso de uso probado** | CU03 |
| **Descripción** | Consulta de asignaturas |
| **Propósito** | Comprobar que las asignaturas matriculadas se muestran correctamente. |
| **Acciones** | Tras iniciar sesión, se muestra una pantalla con las asignaturas visibles para el usuario. |
| **Datos de entrada** | nombre de usuario = "ssanchez"<br>password = "Prueba.2020" |
| **Salida esperada** | OK → Se muestra la asignatura "Ingeniería de Requisitos" |
| **Salida obtenida** | OK → Se muestra la asignatura "Ingeniería de Requisitos" |
| **Resultado** | Éxito |
| **Información adicional** | |

| Identificador caso de prueba | CP05 |
|---|---|
| **Caso de uso probado** | CU04 |
| **Descripción** | Consulta de tareas |
| **Propósito** | Comprobar que las tareas disponibles en una asignatura matriculada se muestran correctamente. |
| **Acciones** | Tras iniciar sesión, se selecciona la asignatura "Ingeniería de Requisitos" |
| **Datos de entrada** | nombre de usuario = "ssanchez"<br>password = "Prueba.2020" |
| **Salida esperada** | OK → Se muestra un listado de tareas en la que aparece "AEC2: Casos de uso" |
| **Salida obtenida** | OK → Se muestra un listado de tareas en la que aparece "AEC2: Casos de uso" |
| **Resultado** | Éxito |
| **Información adicional** | Si la tarea estuviese calificada, se muestra la nota de esta. Si la tarea tuviese algún comentario del profesor, se muestra un icono que permite leer el comentario. |

| Identificador caso de prueba | CP06 |
|---|---|
| **Caso de uso probado** | CU05 |
| **Descripción** | Refresco de información |
| **Propósito** | Comprobar que es posible actualizar la información de las asignaturas. |
| **Acciones** | Con la sesión iniciada, se realiza el gesto táctil que permite actualizar la información. |
| **Datos de entrada** | nombre de usuario = "ssanchez"<br>password = "Prueba.2020" |
| **Salida esperada** | OK → Se muestra el cambio realizado por el profesor si lo hubiera. |
| **Salida obtenida** | OK → Se muestra el cambio realizado por el profesor si lo hubiera. |
| **Resultado** | Éxito |
| **Información adicional** | Para ver nueva información, el profesor debe de haber realizado cambios en algún elemento de la asignatura. |

| Identificador caso de prueba | CP07 |
|---|---|
| **Caso de uso probado** | CU06 |
| **Descripción** | Cierre de sesión |
| **Propósito** | Comprobar que el usuario puede cerrar la sesión iniciada en cualquier momento. |
| **Acciones** | Con la sesión iniciada, se cierra la sesión. |
| **Datos de entrada** | nombre de usuario = "ssanchez"<br>password = "Prueba.2020" |
| **Salida esperada** | OK → Se muestra la pantalla de inicio de sesión con los campos de nombre de usuario y contraseña vacíos. |
| **Salida obtenida** | OK → Se muestra la pantalla de inicio de sesión con los campos de nombre de usuario y contraseña vacíos. |
| **Resultado** | Éxito |
| **Información adicional** | |

| Identificador caso de prueba | CP08 |
|---|---|
| **Caso de uso probado** | CU07 |
| **Descripción** | Apertura por segunda vez y posteriores con credenciales persistidas |
| **Propósito** | Comprobar que la aplicación recuerda las credenciales tras ser cerrada cuando el usuario así lo ha seleccionado. |
| **Acciones** | Con la sesión iniciada, se cierra la aplicación, y se vuelve a abrir. |
| **Datos de entrada** | nombre de usuario = "ssanchez"<br>password = "Prueba.2020" |
| **Salida esperada** | OK → Se muestra la pantalla de asignaturas matriculadas sin solicitar de nuevo las credenciales al usuario. |
| **Salida obtenida** | OK → Se muestra la pantalla de asignaturas matriculadas sin solicitar de nuevo las credenciales al usuario. |
| **Resultado** | Éxito |
| **Información adicional** | Las credenciales son almacenadas de forma encriptada. |

| Identificador caso de prueba | CP09 |
|---|---|
| **Caso de uso probado** | CU07 |
| **Descripción** | Apertura por segunda vez y posteriores con credenciales no persistidas |
| **Propósito** | Comprobar que la aplicación no recuerda las credenciales tras ser cerrada cuando el usuario así lo ha seleccionado. |
| **Acciones** | Con la sesión iniciada, se cierra la aplicación, y se vuelve a abrir. |
| **Datos de entrada** | nombre de usuario = "ssanchez"<br>password = "Prueba.2020" |
| **Salida esperada** | OK → Se muestra la pantalla de inicio de sesión con los campos de nombre de usuario y contraseña vacíos. |
| **Salida obtenida** | OK → Se muestra la pantalla de inicio de sesión con los campos de nombre de usuario y contraseña vacíos. |
| **Resultado** | Éxito |
| **Información adicional** | |