Informe Final: Bot de Telegram aplicado al Microlearning

A. Moreno Gimeno, Estudiante Ingeniería Informática, *EE (UAB)*

Resumen—La necesidad de recibir conocimiento específico sobre un tema de forma útil y rápida, ha hecho que modalidades como el Microlearning ganen gran popularidad. El microlearning se adapta bien al repaso de contenidos fuera de las aulas, en momentos libres del dia a día. En este TFG se propone desarrollar un sistema de microlearning basado en un bot de Telegram con la finalidad de ayudar a los alumnos a ampliar y mejorar sus conocimientos sobre ciertas asignaturas. Mediante sus dispositivos móviles pueden realizar lecciones muy cortas de los temas de mayor interés de la asignatura. Esto facilita a los estudiantes el repaso de la materia en tiempos muertos que tengan a lo largo del día. Además, la actividad de los estudiantes queda registrada, permitiendo al profesorado conocer el seguimiento que se hace de los microcontenidos propuestos.

Palabras Clave—Bot de Telegram, Microaprendizaje, Python, Multi-plataforma

Abstract— The need to receive specific knowledge about a subject in a useful and fast way, has made modalities such as Microlearning gain great popularity. Microlearning is well suited to the review of content outside the classroom, in free moments of the day to day. In this TFG we propose to develop a microlearning system based on a Telegram bot to help students to expand and improve their knowledge of certain subjects. Through their mobile devices they can perform very short lessons of the most interesting topics of the subject. This makes it easier for students to review the subject matter in downtime throughout the day. In addition, the students' activity is recorded, allowing the teachers to know the follow-up of the proposed microcontents.

Keywords—Telegram Bot, Microlearning, Python, Multi-platform

1 Introducción

HOY en día hay cada vez más necesidad de recibir conocimiento sobre algún tema específico de forma útil y rápida. Es decir, adquirir conocimientos y competencias de forma eficiente, personalizada y en el momento adecuado (aprendizaje justo a tiempo). Entre las diferentes propuestas orientadas a conseguir esta necesidad, una de las que más popularidad ha ganado últimamente es el microlearning.

El término de "Microlearning" [1] o Microaprendizaje, comenzó a utilizarse ya en el año 2002 como forma de apoyar a los profesores en la mejora de la práctica de enseñanza, tanto en métodos como en contenidos. Sin embargo, fue en 2009 cuando T. Hug [2] definió este término como una modalidad de aprendizaje orientada en la fragmentación de contenidos didácticos, a través de los cuales se obtienen ciertas competencias útiles para casos específicos, y lo sitúa como un rol importante que siempre ha estado presente en el contexto del aprendizaje institucional.

Un aspecto muy importante a tener en cuenta del microaprendizaje es que se asume que los estudiantes tienen la suficiente motivación propia para buscar y completar de forma autónoma el aprendizaje que considere necesario, puesto que estas microlecciones son muy específicas y no siempre contendrán toda la información exhaustiva sobre un tema en concreto.

A continuación, se detallan sus características más importantes y cómo ayudan estas en el aprendizaje:

- El microaprendizaje se basa en la división del contenido a aprender en microcontenidos [3]. Este término fue acuñado por Anil Dash (programador e impulsor del movimiento blogger). Define toda aquella información publicada de forma abreviada que se restringe a un solo tema principal, al cual se accede por medio de un software y un dispositivo.
- La duración de estos microcontenidos es, por lo general, un periodo corto de tiempo que puede durar entre 5 y 15 minutos. Sin embargo, como bien describe Shannon Tipton [4]: No hay plazos para el microlearning. No hay una cantidad de tiempo mágica. Todo se trata de la necesidad y el contexto. Y dice que debemos de tener en mente algo muy importante: "El tiempo que sea necesario y tan corto como sea posible".
- Cada lección está diseñada para un objetivo de aprendizaje específico, por lo tanto, se pueden incluir todo tipo de temas y conceptos, pero sólo del aspecto en cuestión del cual se desea mejorar los conocimientos.
- Los microcontenidos se presentan en una gran variedad de formas: textos, imágenes, audios, videos, presentaciones, interacciones, juegos, escenarios, actividades evaluables, y un largo etcétera. Este es un punto muy relevante, puesto que dependiendo del tipo de contenido que se desea

- *Adrián Moreno, E-mail de contacto: Adrian.morenog@e-campus.uab.cat*
- *Mención: Ingeniería del Software*
- *Trabajo tutorizado por: Daniel Ponsa Mussarra (Departamento de Ciéncias de la Computación)*
- *Curso 2021/22*

enseñar se puede adaptar el método para que sea el más apropiado.
- Lo ideal es que el contenido sea accesible siempre que se necesite. Por ello, los teléfonos móviles [5] son una herramienta idónea para este tipo de aprendizaje, pues en el mundo actual, una gran mayoría de personas tienen acceso a este dispositivo, y es por este motivo principalmente que se ha decidido usar como base dicho dispositivo.

En este TFG se ha desarrollado un sistema de microlearning para asignaturas de grados universitarios. El sistema permite registrar la actividad de los alumnos con el sistema. Esto posibilita conocer el grado de seguimiento de los microcontenidos, así como identificar los conocimientos asentados y los temas que genera más confusión. Se propone desarrollar el sistema basandose en la aplicación de mensajeria Telegram [6], debido a la posibilidad de crear e implementar en la misma un bot; una pequeña aplicación software programada para interactuar con el usuario a través del chat. Sus acciones y respuestas dependen del objetivo con el que haya sido creado. En este caso, la finalidad de nuestro bot es ayudar a los estudiantes de una asignatura mediante el ya mencionado microlearning.

2 Objetivos

El objetivo principal del proyecto es desarrollar un sistema de micro diseminación de contenidos e implementar dicho sistema mediante un bot en Telegram.

Este objetivo se ha fragmentado en varios subobjetivos que permiten centrarse únicamente en un concepto más pequeño y, una vez todas las piezas estén completas, juntarlas para completar el objetivo inicial.

A. Ofrecer un mecanismo para añadir y gestionar los microcontenidos (dar de alta una microlección, un itinerario, …).
B. Limitar el acceso a los microcontenidos solo a alumnos registrados.
C. Registrar la actividad de cada alumno con el sistema.
D. Posibilitar una atención personalizada basada en las actividades registradas.
E. Permitir observar toda la actividad del sistema de forma sencilla.

3 Estado del arte

El sistema de aprendizaje del microlearning es conocido desde hace tiempo. Sin embargo, no fue hasta hace un par de años que su popularidad se disparó, a causa de la pandemia y el confinamiento obligatorio que afectó a todo el mundo.

Debido a su gran cohesión con el e-learning y el aprendizaje continuo, actualmente se utiliza en multitud de empresas como alternativa para formar a los trabajadores. Pero no solamente eso, sino que también se utiliza como forma de aprendizaje alternativo donde y cuando el usuario quiera.

Respecto a los bots de Telegram, fue en junio de 2015 cuando la aplicación de mensajería comenzó a ofrecer las herramientas necesarias para que cualquiera pudiese crearlos y automatizar tareas en la aplicación. Aunque la combinación de ambos conceptos no es demasiado popular.

Con una búsqueda más exhaustiva se pueden encontrar pocos proyectos que combinen ambos conceptos, y ninguno que sea muy popular o que use la gente habitualmente.

Por el contrario, si existen proyectos bastante populares relacionados con el microlearning que son muy populares en dispositivos móviles como:

- Duolingo: Plataforma web creada en Estados Unidos destinada al aprendizaje gratuito de idiomas y a la certificación del nivel de inglés, que posteriormente fue ampliada con una aplicación móvil.
- W3Schools: Sitio web para aprender tecnologías web en línea. Contiene tutoriales de HTML, CSS, JavaScript, SQL, PHP, XML y otras tecnologías.

En definitiva, aunque hay propuestas similares con mucha popularidad, estas están desarrolladas para otro tipo de entornos y orientadas solo a un alumno. Por el contrario, este proyecto permite a un usuario como profesor proponer contenidos, y posibilita analizar el uso de los contenidos que hacen los alumnos. Lo cual implica que este proyecto es innovador y puede servir para ayudar tanto a profesores como a estudiantes en el futuro.

4 Metodología

Para planificar el proyecto y todas las tareas que se deberán realizar, se seleccionó la metodología ágil Kanban [7]. La palabra "Kanban" proviene del japonés y significa "tablero visual". Esta metodología se centra en los flujos de trabajo y en ir asignando y reordenando tareas a medida que avanza el proyecto. A diferencia de otras metodologías usadas en proyectos de equipo como Scrum, no se centraliza en el trabajo en equipo, ni depende de seleccionar roles específicos para los diferentes miembros. Además, el desarrollo es de forma incremental e iterativa, permitiendo ampliar y corregir errores durante el desarrollo del proyecto, mediante el feedback frecuente que se recibe por parte de los usuarios. Por estos motivos, es una opción ideal para un proyecto unipersonal.

Esta metodología es muy sencilla de aplicar, además de una herramienta visual en forma de tablero. En este se anotan todas las tareas a realizar durante el proyecto y se clasifican por columnas (Fig. 1). Las tareas propuestas en un inicio pueden aumentar a medida que va avanzando el proyecto, y se pueden ir añadiendo a la columna "To Do" nuevas tareas que se consideren necesarias para el correcto desarrollo del proyecto.

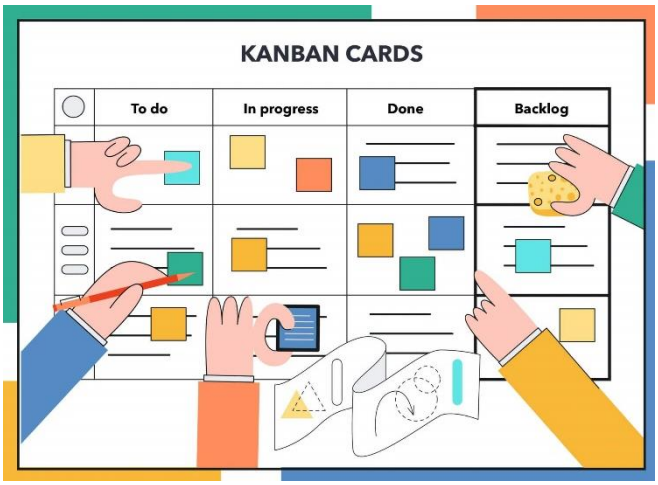


Fig. 1 Tablero Kanban

Además, este método tiene como lema "Stop starting, start finishing", es decir, se priorizan las tareas ya iniciadas, sobre las nuevas, de esta forma se limitan el total de tareas a asignar, evitando que se empiecen nuevas tareas hasta haber finalizado las previamente empezadas.

En definitiva, Kanban me ha ayudado a realizar la mayor parte de las actividades de gestión del proyecto más rápida y eficazmente con los mínimos recursos y sin invertir demasiado tiempo en este aspecto, obteniendo un excelente resultado.

5 Herramientas

A continuación, se describen las herramientas seleccionadas para el desarrollo del proyecto, y que han permitido seguir la metodología previamente detallada:

Asana: Para implementar la metodología Kanban, la herramienta de gestión de tareas que se utilizó es Asana [8], que permite asignar mediante post-its, las tareas que se van a realizar.

Esta herramienta es muy similar a Trello [9] (herramienta que había utilizado previamente y que por ello fue mi idea inicial). Sin embargo, Asana tiene una interfaz mucho más limpia y amigable que Trello, además de un enfoque más orientado al desarrollo de metodologías ágiles.

GitHub y GitKraken: Para el mantenimiento del código y el control de versiones se va a utilizar GitHub [10]. Habitualmente, como aplicación para interactuar con el repositorio, se utiliza el GitHub Desktop, que es una aplicación de gestión del código fuente basada en GitHub, tanto para subir actualizaciones del código al repositorio como para tener un historial con todas las versiones y modificaciones del código a lo largo del proyecto. Sin embargo, se ha decidido utilizar otra aplicación muy similar, pero con una interfaz mucho más amigable como es GitKraken [11].

Este software de control de versiones permite hacer un rollback en el caso de que haya algún problema durante el desarrollo de las nuevas versiones.

Microsoft Teams: Para las reuniones online con el tutor, se ha utilizado Microsoft Teams, que es una plataforma unificada de comunicación y colaboración que combina chat persistente en el lugar de trabajo, reuniones de video, almacenamiento de archivos e integración de aplicaciones.

Draw.io: Para los diagramas se ha utilizado Draw.io [12] debido a que es gratuita, permite a los usuarios crear y compartir diagramas online y offline, y guardar todo localmente. Además, la herramienta en línea funciona con Google Drive y Dropbox, y está profundamente integrada y es fácil de auditar en productos como Jira de Atlassian.

6 Planificación

La fase del desarrollo de dicho proyecto se ha desarrollado de manera incremental. En esta fase se realizan varios sprints de forma que la primera iteración contiene el prototipo inicial del proyecto. Y a medida que se van incluyendo más iteraciones, el bot va tomando forma e incluyendo todos los subobjetivos mencionados previamente.

La planificación del proyecto se ha dividido en cuatro fases, en la fase del desarrollo se pueden observar estos sprints incrementales:

(1) Fase Inicial:
   i. Análisis de las necesidades de los usuarios.
   ii. Identificación de los requisitos y tecnologías necesarias.
   iii. Estudio de la documentación sobre la funcionalidad del bot: cómo implementarlo con Telegram, y la aplicación del microlearning como estrategia educativa para nuestro bot.
   iv. Propuesta de un diseño que sirva de solución.

(2) Fase de Diseño:
   i. Diseñar un prototipo de bot que permita acceder a microlecciones a diferentes usuarios simultáneamente.
   ii. Implementar el prototipo inicial y comprobar su funcionamiento.
   iii. Definir los posibles casos de uso y diagramas de flujo.
   iv. Diseñar la base de datos y la interfaz del propio bot.

(3) Fase de Desarrollo:
   i. Desarrollar el bot y testearlo.
   ii. Implementar la base de datos.
   iii. Gestionar los contenidos y el acceso de usuarios autenticados en el sistema.

En este punto se define la duración estimada de desarrollo mediante el tiempo de cada tarea.

Para la fase de desarrollo del proyecto, se propuso inicialmente como objetivo principal un MVP que contemplaba todas las funcionalidades del Sprint 1. Y a posteriori, añadir un par de sprints de mejora que finalmente se completaron exitosamente.

- Sprint 1:

i. Definir itinerario
ii. Dar de alta lección
iii. Modificar contenidos
iv. Consultar opciones
v. Consultar itinerario
vi. Realizar microlección

- Sprint 2:
i. Dar de alta estudiante
ii. Acceso de autenticación en el sistema
iii. Valorar itinerario
iv. Consultar valoraciones
v. Consultar comentarios

- Sprint 3:
i. Enviar notificaciones
ii. Modificar notificaciones
iii. Pedir recomendación
iv. Consultar progreso
v. Consultar progreso alumnos

Cabe destacar que, para cada funcionalidad de las descritas anteriormente, se realizó una prueba comprobando su correcto funcionamiento.

(4) Fase de Documentación:
i. Documentar los resultados y redactar la memoria del sistema.
ii. Ordenar la documentación del proyecto, y presentarlo.

Para una revisión más detallada de la planificación hecha, en el apéndice A se detalla el diagrama de Gantt del proyecto.

7 Requisitos

En el diagrama de casos de uso que se encuentra en el apéndice B, se muestran los roles del usuario como profesor y como estudiante, y las acciones que cada uno puede realizar con el bot.
Los roles son absolutos y están prefijados desde un inicio. Además, el diseño está pensado para que los estudiantes tengan unas limitaciones, y sea el profesor el que vaya dando acceso a nuevas partes a medida que se avance con la asignatura que se imparte.
Para una revisión más detallada del diseño, en el apéndice B se detalla el diagrama de casos de uso.

En base a estos casos de uso del diagrama que se encuentra en el apéndice B, se derivan una serie de requisitos a cumplir, tanto funcionales como no funcionales, que definen las restricciones que el sistema debe satisfacer. A continuación, se detallan los diferentes requisitos establecidos:

7.1. Requisitos funcionales

- RQF1. El sistema debe proveer a los usuarios la capacidad de registrar su id de usuario de Telegram, nombre, apellido y niu mediante un formulario.
- RQF2. El sistema registrará en la base de datos el login si es la primera vez que el estudiante inicia sesión.
- RQF3. El sistema mantendrá el inicio de sesión, hasta que se borren los datos de la base de datos.
- RQF4. El sistema registrará en la base de datos la microlección realizada por el estudiante indicando su correspondiente fecha.
- RQF5. El sistema quedará a la espera en determinado punto durante la realización de una microlección, permitiendo al estudiante cerrar Telegram. De esta forma, el usuario podrá continuar más tarde en el punto donde lo dejó.
- RQF6. El sistema permitirá al estudiante realizar microlecciones en formato, texto, imagen o vídeo, en función de la información que haya en la base de datos.
- RQF7. El sistema permitirá al estudiante introducir la valoración que considere oportuna sin límite de caracteres.
- RQF8. El sistema permitirá al profesor visualizar en la base de datos todas las valoraciones de los alumnos por fecha y niu de cada alumno.
- RQF9. El sistema registrará en la base de datos, en el perfil de un usuario cuando este active o desactive las notificaciones.
- RQF10. El sistema tendrá en cuenta las tareas realizadas y la fecha de estas para enviar la información cuando el usuario pida una recomendación.

7.2. Requisitos no funcionales

- RQNF1. El sistema debe tener una apariencia visual amigable e intuitiva.
- RQNF2. El sistema debe ser funcional en todo tipo de dispositivos que permitan instalar Telegram.
- RQNF3. Es necesario para utilizar el sistema que dispositivo posea conexión a internet.

8 Diseño

En esta fase se definieron el diseño general del prototipo, tanto su interfaz como funcionamiento. Para ello se realizó un diseño inicial de la interfaz, y un diseño de la arquitectura del sistema.

a. Diseño general de la base de datos

En el apéndice 4 se puede observar el diseño general de la base de datos que fue realizado en una sesión, con la participación de usuarios finales. El resultado se muestra en el apéndice D, donde hay entidades y relaciones clave para desarrollar el sistema de este TFG, así como entidades y relaciones de otros servicios de interés (sistema de flashcards, o de sistema de preguntas frecuentes) que no forman parte del sistema a desarrollar en este TFG.

En la imagen del apéndice D se pueden observar diversas tablas, entre ellas, cada asignatura se divide en temas, y dentro de estos temas, se incluyen las microlecciones, que son un conjunto de microcontenidos organizados por un índice. El orden que tiene que

seguir una asignatura viene definido por la tabla Itinerario. Esta base de datos global ha servido para entender el concepto y diseño de la base que cada estudiante aplicaría en su TFG.

b. Diseño de la interfaz

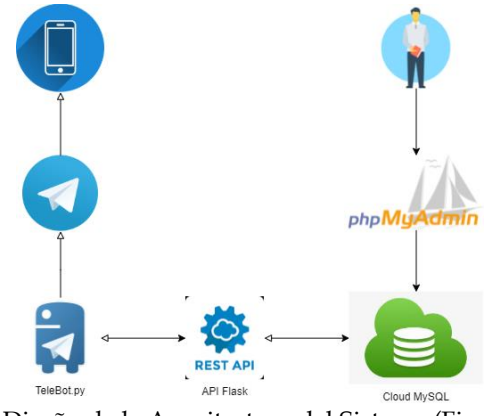
En Telegram, normalmente la interacción con un bot toma la forma de intercambio de mensajes. Esto no es muy pulido, porque se va acumulando una 'conversación' que en nuestro caso no tiene interés en que se mantenga. Para mejorar este aspecto del bot, en este trabajo se ha desarrollado un acceso a los microcontenidos mediante la pulsación de botones, donde la información mostrada se elimina una vez el usuario ya la ha revisado. De esta forma se resuelve el problema mencionado inicialmente.

El diseño de pantallas de la aplicación se obtuvo a partir de refinar un esbozo (sketch) inicial. El esbozo definitivo se documenta en el apéndice C.

c. Diseño de la Arquitectura del sistema

Para establecer la conexión entre la base de datos y la aplicación, se ha realizado un diseño modular del sistema que muestra la figura 2, que se detallará en el apartado de desarrollo.

Esto ha implicado un cambio en el diseño de la arquitectura, ya que en vez de conectar el sistema directamente a Telegram, esto se realiza mediante un paso intermedio en el que se encuentra este sistema. El diseño final donde se puede apreciar este paso intermedio es el siguiente:



Diseño de la Arquitectura del Sistema (Fig. 2)

El alumno accede al sistema a través de Telegram, como se puede apreciar en la imagen del teléfono. Se comunica con un bot (Telebot.py), que mediante un sistema de botones gestiona la conversación con el usuario. Dicho bot consulta/registra toda la información que el usuario le pide/envía de la base de datos comunicándose mediante una Rest API. Es esta Rest API quien se comunica mediante consultas SQL con la base de datos. Además de convertir estos datos en un formato sencillo para que el bot se lo muestre al usuario de forma clara y ordenada.

Por otra parte, el profesor interactúa con el sistema a través de las utilidades que ofrece el sistema de la base de datos. A través de un ordenador de sobremesa o un portátil, puede añadir contenidos, así como revisar el listado de usos del sistema por parte de los alumnos.

Lo más interesante de utilizar una Rest API es que el programa se puede migrar con facilidad, aislando el bot del sistema de la base de datos que se utiliza, de forma que si se quiere cambiar de gestor de bases de datos no será necesario modificar el código del bot. Además, la REST API posibilita proveer información a otras posibles aplicaciones (por ejemplo, un sistema de micro-learning alternativo basado en una aplicación móvil).

9 Desarrollo y Resultados

Para el desarrollo del front-end del bot se ha utilizado Python [13] ya que es un lenguaje muy popular en bots y con bastante información al respecto.

Para el back-end del bot se empezó utilizando para el prototipo NeDB [14], una librería o almacén de datos NoSQL más pequeño que imita a MongoDB [15]. Esta permitía incorporar una base de datos local de forma sencilla y con pocos recursos.

Sin embargo, tras los primeros cambios y la adición de la Rest API (desarrollada también con Python) para interactuar con la nueva base de datos, se cambió a MySQL [16] por la facilidad de introducir datos desde PHPMyAdmin [17] y de enlazar las tablas de forma gráfica.

MySQL es de los sistemas de gestión de bases de datos relacional más utilizados por sitios web grandes y populares como Google, Facebook o Youtube.

El back-end actual basado en MySQL cubre todas las necesidades requeridas por el sistema. No obstante, se ha generado y comprobado también su gestión basada en NeDB por si en un futuro se quisiera utilitzar este sistema. Además, con la adición de la Rest API, se aisla el programa que controla el bot del sistema de la base de datos que se usa para gestionar los datos.

9.1. Desarrollo back-end

Para el back-end y relacionar la base de datos con el bot se ha desarrollado una Rest API con Flask. Consultar los datos mediante la Rest API es muy rápido y eficiente, además permite en caso de querer sustituir la base de datos, poder utilizar la misma API simplemente modificando la llamada a la nueva base de datos. Mediante esta API se filtran y envian los datos deseados. Y después desde el bot solamente se trabaja con esos datos.

Mediante PHPMyAdmin y MySQL se pueden editar de forma sencilla y visual todos los datos de la base. Además, permite a varios administradores acceder con su cuenta individual a dichos datos para, por ejemplo, dar de alta a estudiantes una vez envíen su User_id de Telegram.

Por estos motivos es una gran idea utilizar este tipo de base de datos.

9.2. Desarrollo front-end

El front-end del bot se ha desarrollado con Python. Además, se han utilizado librerías existentes como telebot

[18] y telegram, que han ido creando usuarios de forma altruista y poniendo a disposición de todos.

Como se ha comentado previamente, un usuario se comunica de forma habitual con un bot mediante comandos, sin embargo, esto implica que el usuario tenga que escribir y saber que hace cada comando, por lo que se ha reducido a un simple /start. Hay dos elementos que permiten que el bot se comunique con el usuario, el primero es mediante mensajes, que dan instrucciones y información al usuario, y el segundo son los botones, que es el elemento sustituto de los comandos mencionados al inicio, de esta forma es más intuitivo y sencillo para el usuario, además de ahorrarle tiempo.

i. Mensajes

Para evitar tener un listado enorme de mensajes entre el bot y el estudiante, se ha utilizado el "ConversationHandler()". Es un método de la clase Handler que permite mantener una conversación con uno o varios usuarios a través de las actualizaciones de Telegram gestionando cuatro colecciones de "handlers" diferentes.

Mediante este método, es posible crear una conversación persistente (que se mantiene pese a que se reinicie el bot), y que actualiza el mensaje inicial del bot con la siguiente respuesta. Además, la conversación será única y personalizada para cada usuario, de forma que no habrá cruces de datos.

ii. Botones

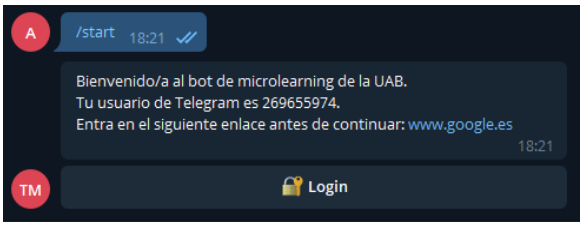
Para que los estudiantes puedan interactuar de una forma más sencilla y evitar que tengan que conocer mil comandos, se han añadido botones que proporciona la librería telegram a la mayor parte de mensajes. Así el estudiante solamente tiene que ir pulsando debajo de cada mensaje que el bot le envía para seguir la conversación.

Estos botones se crean gracias a los objetos InlineKeyboardMarkup y InlineKeyboardButton que como bien indican sus nombres representan un teclado en línea junto al mensaje al que pertenecen.

A continuación, se detalla como se han resuelto los casos de uso principales del sistema, que se han listado en cada Sprint en la parte de planificación:

CU.1 Hacer login en el bot

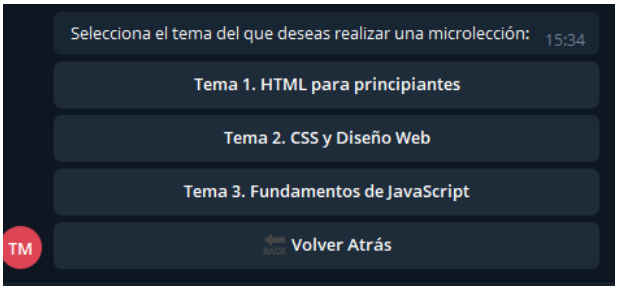
Para esta fase de prueba o prototipo, el método de autentificación parte implica dos fases. La primera es que los estudiantes introduzcan el comando /start para obtener su user_id personal y único de Telegram y enviarlo mediante un formulario al profesor (en este caso se ha introducido una URL estándar) (ver Fig.3).



Inicio del bot (Fig. 3)

En la segunda, el profesor valida e introduce en la base de datos la información recogida mediante el formulario, garantizando el acceso a los alumnos matriculados en su asignatura.

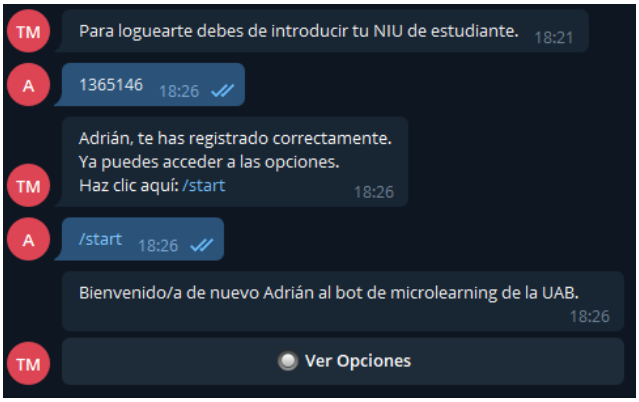
Una vez se han cumplido los requisitos anteriores, simplemente se inicia el bot mediante el comando /start, y este pide el NIU al estudiante para identificarlo, compara este NIU con el user_id que se ha introducido previamente en la base de datos y inicia sesión. Tras completar el inicio de sesión, el usuario queda logueado de forma permanente, y puede acceder a las funcionalidades primarias del bot (ver Fig.4).



Completar el login (Fig. 4)

CU.2 Ver opciones

Una vez se ha completado el login, se tiene acceso al menú de opciones donde se puede elegir entre las distintas funcionalidades que se muestran en la figura 5.

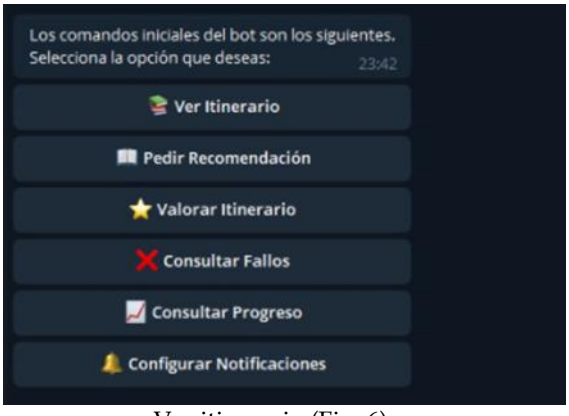


Opciones al finalizar el login (Fig. 5)

CU.3 Ver Itinerario

El itinerario lo define el profesor inicialmente y contiene todos los temas que pueden repasar los

alumnos para ampliar sus conocimientos sobre la materia.
Para acceder a este, se llama desde una Rest API a la base de datos para recuperar el total de temas disponibles. Y se muestra dicha información al usuario en forma de lista con botones (ver Fig. 6).

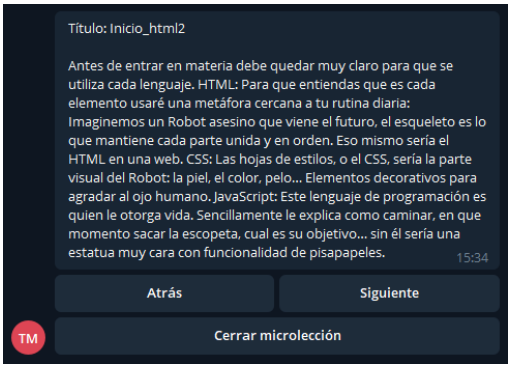


Ver itinerario (Fig. 6)

CU.4 Realizar Microlección

Cuando se selecciona un tema de los mostrados en la lista, se recuperan los datos de la misma forma que en itinerario y se obtienen el total de microlecciones disponibles mostrándose de nuevo como una lista.

En esta pantalla, el alumno selecciona la microlección a seguir, que se muestra como una secuencia de mensajes por los que puede navegar, teniendo la opción de cerrar la microlección cuando lo desee, tal y como se aprecia en la figura 7.



Ejemplo de una microlección (Fig. 7)

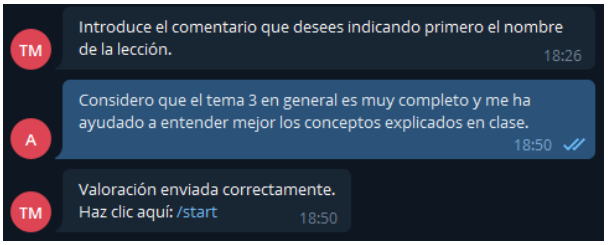
CU.5 Pedir Recomendación

Esta función comprueba en base a las microlecciones realizadas cual es la microlección que más tiempo lleva el usuario sin repasar y se la propone. Aunque como comentaré en los resultados, es una propuesta preliminar sin finalizar, como comento posteriormente en las conclusiones.

CU.6 Valorar Itinerario

Es una función bastante sencilla que permite introducir al usuario el comentario que desee sobre el tema/microlección deseada. Por ejemplo, avisar de un error, indicar que el contenido le ha generado dudas, o bien comentar si le ha resultado útil una microlección.

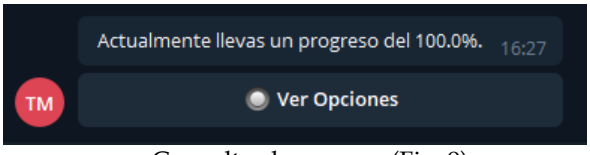
Dicho comentario no tiene un límite específico de caracteres, lo que permite al alumno explicarse en profundidad. Además, queda registrado en la base de datos el NIU del estudiante que ha introducido el comentario y la fecha, para su fácil identificación y clasificación (ver Fig. 8).



Valorar itinerario (Fig. 8)

CU.7 Consultar Progreso

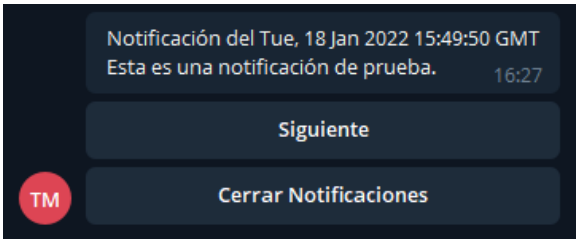
Dicha función permite al usuario consultar el progreso actual que lleva en la asignatura mostradola en base al porcentaje de temas del itinerario realizados. Es bastane sencilla y tiene bastante margen de mejora futura, como mostrar los temas que todavía faltan por realizar o los ya realizados, sin embargo, por motivos de tiempo durante el desarrollo del proyecto, se ha dejado como se muestra a continuación (ver Fig. 9).



Consulta el progreso (Fig. 9)

CU.8 Mostrar Notificaciones

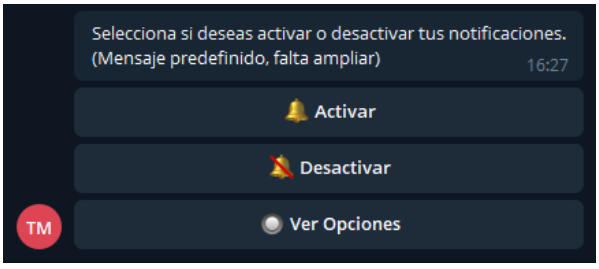
Esta función, junto con la siguiente, están relacionadas directamente con el profesor. Ya que dichas funciones son un apaño de lo que sería enviar notificaciones a los alumnos. Sin embargo, Telegram limita el formato para enviar notificaciones personalizadas mediante el bot, por lo que se ha hecho una adaptación que se puede mejorar en un futuro si se actualiza por ejemplo a aplicación móvil (ver Fig. 10).



Muestra una notificación (Fig. 10)

CU.9 Configurar Notificaciones

En relación a la funcionalidad anterior, esta permite activar o desactivar las notificaciones, de forma que el alumno puede elegir si desea o no recibir mensajes del profesor. Tiene también las limitaciones mencionadas anteriormente, por lo que es una adaptación con gran margen de mejora (ver Fig. 11).



Activar o desactivar notificaciones (Fig. 11)

10 Pruebas del sistema

Durante el desarrollo de cada funcionalidad de la aplicación, y tras la finalización del bot se han realizado una serie de pruebas para comprobar el correcto funcionamiento de las funcionalidades añadidas. De esta forma se iban buscando posibles problemas o fallos en el comportamiento del bot. Los tipos de pruebas utilizados son:

- Pruebas de unidad:

El ingeniero de pruebas, que en este caso coincide con el desarrollador, determinaba distintos valores de entrada para examinar los posibles flujos de ejecución del programa y comprobar de que se devuelven los valores de salida esperados.

Gracias a que el bot ha sido desarrollado con Python, este tipo de pruebas se han podido realizar con el editor de código fuente utilizado para el desarrollo. De forma que el testeo ha sido más sencillo y amplio, permitiendo pasar una mayor cantidad de pruebas.

- Exploratory testing:

También se ha realizado exploratory testing al finalizar el desarrollo principal del proyecto, comprobando un amplio abanico de posibles interacciones entre el alumno y el sistema. En este tipo de pruebas no se necesita una planificación tan detallada y funciona mejor cuando quien hace el testeo ya tiene un conocimiento previo del funcionamiento de la aplicación.

Gracias a este tipo de pruebas se han podido encontrar algunos errores funcionales, además de realizar cambios y ajustes basándome en los comentarios de las personas que realizaron este tipo de pruebas.

11 Conclusión

En este trabajo de final de grado se ha propuesto un sistema de microcontenidos basado en Telegram para estudiantes universitarios. El sistema permite al profesorado gestionar contenidos, controlar el acceso de los alumnos a los materiales y consultar quien utiliza el sistema. En el caso de los alumnos, el sistema les permite loguearse, acceder a los contenidos que ha definido previamente el profesorado, realizar las lecciones que deseen y valorarlas.

Se han logrado alcanzar los objetivos planteados en un producto mínimo viable completado. Además, se ha mejorado este mvp con algunas funcionalidades como permitir al estudiante que introduzca una valoración o consulte el progreso que lleva hasta ahora.

En general, se han seguido los diseños del diagrama de casos de uso durante el desarrollo del bot, además del diseño inicial de la interfaz, aplicando pequeños cambios y matices cuando era necesario. Esto ha permitido crear un bot fiel al planteado al inicio del proyecto.

Se ha seguido la planificación propuesta inicialmente, a excepción un retraso la fase inicial del desarrollo de la aplicación, que se compensó con más trabajo durante las siguientes semanas. La metodología elegida fue la correcta y permitió tener organizadas las tareas pendientes y tener un control sobre las cosas que quedaban por hacer.

A nivel personal, el desarrollo de este proyecto me ha permitido profundizar más sobre tecnologías como Python y MySQL, y como se enlazaban con Telegram. Ha sido una gran experiencia de la cual estoy convencido de que me será de utilidad de cara al futuro.

Durante el desarrollo del proyecto se ha observado que hay varias mejoras que se le pueden aplicar en un futuro a este proyecto.

La principal y más importante es migrarlo a una aplicación móvil, esto se debe a las limitaciones de Telegram; puesto que a medida que se desarrollaban algunas funcionalidades, se han tenido que adaptar de la forma más optima posible teniendo en cuenta las limitaciones. Estas funcionalidades serían mejorables en formato aplicación.

Entre estas opciones, podemos añadir la posibilidad de poner *pop-ups*, de forma que sea más intuitiva la navegación. Esto puede servir para mejorar funciones como valorar microlección una vez se haya acabado, dar al usuario más información o poner mensajes de error cuando alguna acción del usuario es incorrecta.

Otra de las funcionalidades a mejorar es la de Pedir Recomendación. Esta es bastante compleja y requiere de una base de datos más amplia, además de funciones que en Telegram no puedes implementar al 100%.

Es por este motivo que actualmente la implementación es básica, pero en un futuro se podría mejorar para que, junto a las flashcards, se tenga también en cuenta los tests que realiza el usuario. Y la recomendación sea más precisa y compleja, dando al usuario un mejor resultado.

12 AGRADECIMIENTOS

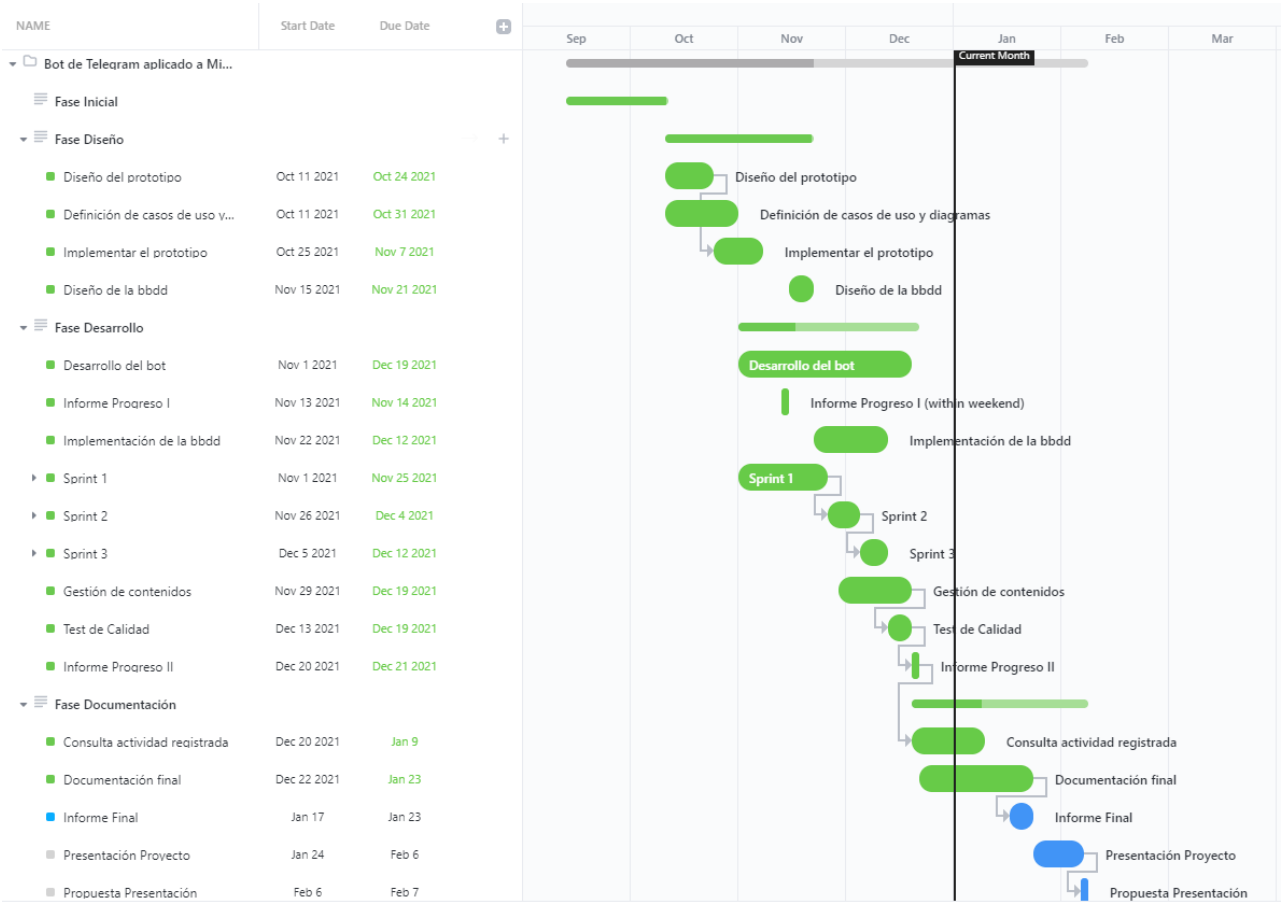
Me gustaría comenzar agradeciendo a mis padres y mi pareja por el apoyo que me han ofrecido, durante todos estos años de formación y con el proyecto de final de grado.
También quiero agradecer a mi tutor durante el proyecto, Daniel Ponsa, que tuvo la idea de crear dicho sistema e hizo su proposición, no solo me ha aconsejado en todo momento, sino que también me ha ayudado cuando lo necesitaba.
Finalmente, me gustaría agradecer también a todas esas personas y amigos que me han ayudado durante el desarrollo de este proyecto con sus opiniones, ideas o comentarios.

13 BIBLIOGRAFÍA

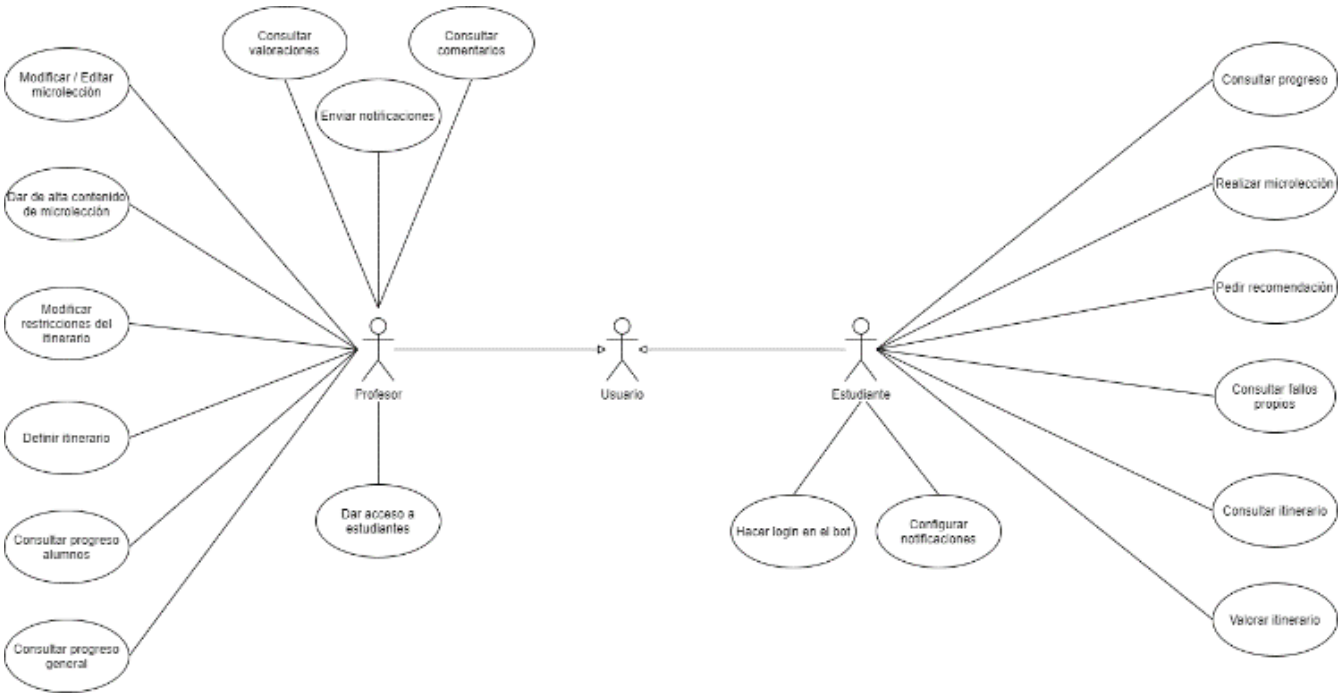
[1] Aquae ODS. El microlearning revoluciona la educación. Consultado el [05/10/2021] en https://www.fundacionaquae.org/microlearning/
[2] T. Hug, N. Friesen, (2009). "Outline of a Microlearning Agenda", eLearning Papers. Consultado el [06/10/2021] en http://www.elearningeuropa.info/files/media/media20252.pdf
[3] Martín, J., y Romero D. (2010). Ambiente de Aprendizaje Móvil basado en Micro-Aprendizaje. IEEE Revista Iberoamericana de Tecnologías del Aprendizaje Vol. 5, núm. 4.
[4] Tipton, S. Microlearning – LT18 Conference. Canal de youtube de LearningTechnologies. Consultado el [06/10/2021] en www.youtube.com/watch?v=J2z8GqQ2Hj8
[5] Salinas, J., y Marín, V. I. (2014). Pasado, presente y futuro del microlearning como estrategia para el desarrollo profesional. Campus Virtuales, Vol. III, núm. 2, pp. 46-61. Consultado el [06/10/2021] en www.revistacampusvirtuales.es
[6] Fernández, Y., Bots de Telegram: qué son, cómo funcionan y 17 recomendados para empezar. Consultado el [07/10/2021] en https://www.xataka.com/basics/bots-telegram-que-como-funcionan-recomendados-para-empezar/
[7] ¿Por qué utilizar la metodología Kanban? Consultado el [30/09/2021] en https://kanbantool.com/es/metodologia-kanban
[8] Asana. Consultado el [01/10/2021] en https://asana.com/es/product
[9] Trello, ¿Qué es Trello y cómo se usa? Consultado el [01/10/2021] en https://trello.com/c/RZExuHxu/6-qu%C3%A9-es-trello-y-c%C3%B3mo-se-usa
[10] Why GitHub? Consultado el [01/10/2021] en https://github.com/features
[11] Learn Git. Commands, Tutorials & More Consultado el [01/10/2021] en https://www.gitkraken.com/learn/git
[12] Calvo, B., Draw.io - La mejor herramienta para diseñar diagramas online. Consultado el [02/11/2021] en https://beatrizcalvo.com/tutorial-draw-io-herramienta-diagramas/
[13] Ramírez, O., (2021). Python a fondo. Editorial Marcombo.
[14] Introducción a NeDB: una base de datos javascript embebida. Consultado el [05/11/2021] en https://www.todojs.com/introduccion-a-nedb-una-base-de-datos-javascript-embebida/
[15] Sarasa, A., (2016). Introducción a las bases de datos. NoSQL usando MongoDB. Editorial UOC, S.L. Consultado el [05/11/2021].
[16] MySQL. Consultado el [01/12/2021] en https://es.wikipedia.org/wiki/MySQL
[17] phpMyAdmin. Consultado el [01/12/2021] en https://es.wikipedia.org/wiki/PhpMyAdmin
[18] pyTelegramBotApi. Consultado el [07/12/2021] en https://github.com/eternnoir/pyTelegramBotAPI

APÉNDICE

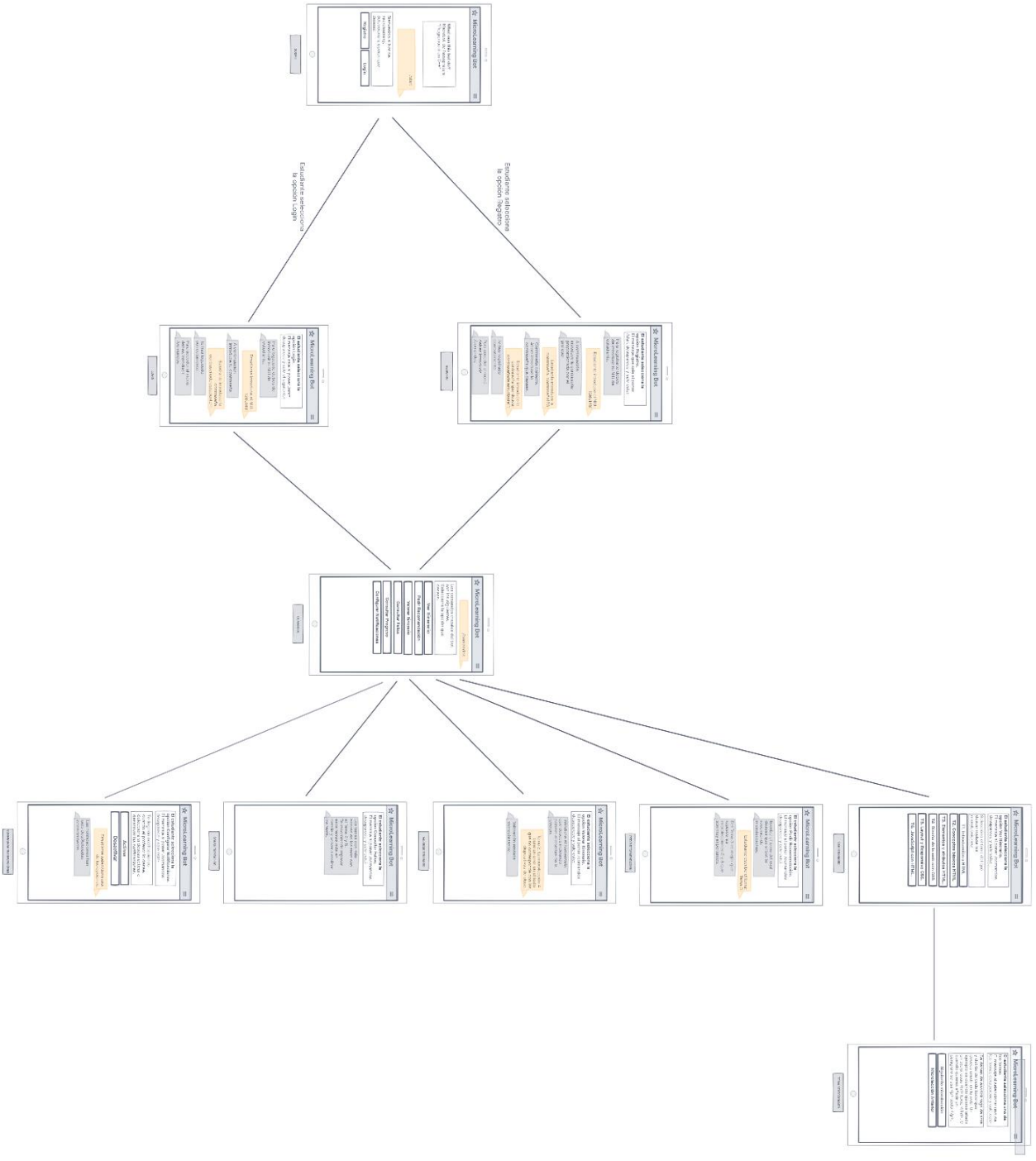
A. Diagrama de Gantt



B. Diagrama de Casos de Uso



C. Sketch de la Aplicación



D. Diseño General de la Base de Datos