Descripción técnica de las tareas realizadas

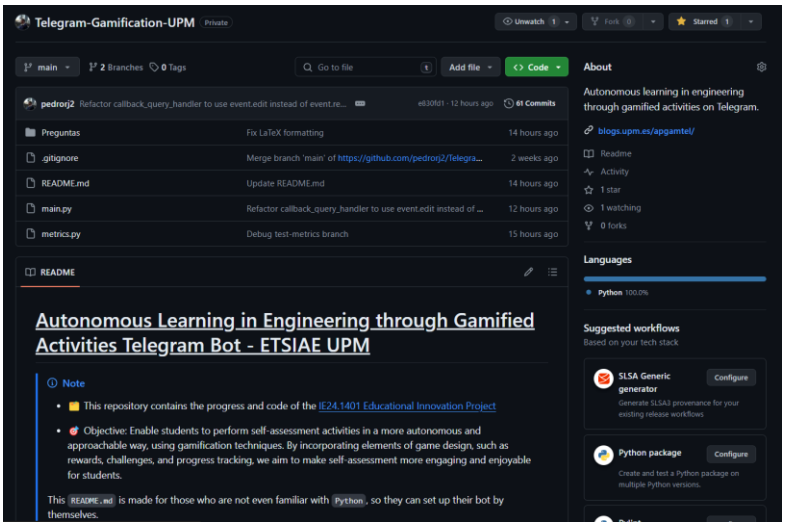
Descripción técnica, concreta y detallada, de todas las actividades, tareas y trabajos desarrollados en el Departamento de la entidad al que ha estado asignado. Debe de incluir una indicación aproximada de la dedicación horaria de cada una de las actividades expuestas. Se debe hacer mención explícita de la relación con los conocimientos y competencias adquiridos en los estudios de la titulación que ha cursado en las tareas realizadas.
Nota para completar el informe: el espacio proporcionado en cada recuadro es sólo indicativo. El alumno puede emplear el que necesite para completar su informe, hasta un máximo de 10 páginas. El formato es libre, pudiendo incorporar esquemas, fotos, gráficos, etc. Este apartado se empleará para valorar la comunicación oral y escrita y evaluar el trabajo realizado.

Durante mi período de prácticas en el Departamento de Aeronaves y Vehículos Espaciales de la Universidad Politécnica de Madrid, he trabajado en el desarrollo de un proyecto de innovación educativa centrado en la gamificación del aprendizaje mediante la plataforma de mensajería Telegram. Este proyecto busca modernizar y hacer más atractivas las actividades de aprendizaje y evaluación mediante la implementación de un Bot interactivo que facilita la participación activa de los estudiantes tanto dentro como fuera del aula. A continuación, se presenta una descripción detallada de las actividades, tareas y trabajos realizados, indicando la dedicación horaria aproximada y competencias adquiridas.

IMPORTANTE: *El progreso de estos tres meses puede verse en el repositorio GitHub del proyecto, donde este último mes de mayo hemos ido incorporando los avances del desarrollo.*

https://github.com/pedrorj2/Telegram-Gamification-UPM

Está restringido para los desarrolladores de este proyecto hasta las ponencias de este por parte de los tutores asociados.



1. Creación del canal de difusión, creación del Bot y gestión de sus credenciales. (4 horas)

Se crearon un canal y un grupo de discusión en Telegram para centralizar la comunicación y difusión de actividades entre los estudiantes. Utilizando el bot @BotFather de Telegram, se creó un bot específico para el proyecto, obteniendo el BOT_TOKEN necesario para su operación.

Competencias: Uso de herramientas de comunicación digital y gestión de comunidades virtuales. Configuración de bots en plataformas de mensajería y manejo de herramientas API de Telegram.

2. Acceso mediante API a Telegram (16 horas)

Descripción: Se estableció la conexión del bot con la API de Telegram usando las librerías de Python, configurando los parámetros API_ID, API_HASH y BOT_TOKEN. Se realizaron pruebas básicas para iniciar a entender el funcionamiento detrás del uso de APIs con diferentes pruebas para realizar difusión de actividades y ejercicios.

Esto fueron pruebas orientadas a iniciarse en las posibilidades que ofrecía la API de Telegram.

Se hicieron muchas pruebas para ver posibles usos de cara a la implementación real.

```python
@client.on(events.NewMessage)
async def handle_new_message(event):
    sender = await event.get_sender()
    message_text = event.message.text
    if sender.username.lower() == 'profesor':
        for user in users_to_notify:
            if user.lower() != 'profesor':
                await client.send_message(user, f"Mensaje de {sender.username}: {message_text}")
    else:
        await event.respond(message_text)
```

```python
@client.on(events.NewMessage)
async def handle_commands(event):
    message_text = event.message.text
    if message_text.startswith('/actividad1'):
        await client.send_message('profesor', "Ejecutando actividad 1...")
        for user in users_to_notify:
            await client.send_message(user, f"{event.sender.username} ha iniciado la Actividad 1.")
```

Competencias adquiridas: Programación en Python y uso de APIs de servicios web.

3. Banco de preguntas e implementación de este (20 horas)

El Bot consiste en un banco de preguntas con un objetivo autodidacta para el alumno, estas preguntas deben de ser elaboradas en introducidas con una estructura a tratar por nuestro código, es por ello que debe de seguir una estructura definida y sencilla de elaborar por aquellos que quieran usar la herramienta con el perfil de aquel que quiera montar y gestionar su propio Bot.

Esta estructura ha sido seleccionada para facilitar la migración de cuestionarios de Moodle al Bot sin tener que volver a reescribir dichas preguntas, gracias a la función de exportar dentro de esta plataforma comentada, esto permite a un profesor una sencilla forma de tener sus preguntas ya creadas dentro de Moodle en el nuevo Bot.

```
// Pregunta 2
:: Introduction to Aircraft Design
:: The induced drag of an aircraft depends mainly on {
~%-100%the laminar or turbulent character of the boundary layer
~%50%the total lift coefficient
~%50%the wing shape
~%-100%the streamlined shape of the fuselage
}

// Pregunta 3
:: Introduction to Aircraft Design
:: A coordinate turn manuever is performed by using {
   ~Antisymmetric deflection of ailerons
   =Rotation of ailerons and rudder
   ~Rotation of ailerons and a particular flap setting
   ~Rotation of ailerons and spoilers
}
```

Aquí vemos que nuestro Bot ha de reconocer dos escenarios distintos, uno como la Pregunta 2, donde vemos una pregunta de respuesta múltiple con diferentes porcentajes, y otro como la Pregunta 3, donde tenemos una única respuesta correcta.

Nuestro Bot ha de hacer una gestión de estas preguntas y poder leer nuestro archivo.txt siendo capaz de diferenciar las diferentes preguntas y dentro de ellas, si es de respuesta múltiple o de respuesta única, ya que se tratarán de manera distinta.

Una de las funciones básicas de este banco de preguntas es:

```python
# Función para obtener las preguntas desde el archivo
def obtener_preguntas_desde_archivo(archivo):
    ruta_completa = os.path.join(preguntas_folder, archivo)
    with open(ruta_completa, 'r', encoding='utf-8') as file:
        contenido = file.read()

    preguntas = re.findall(r'::(.*?)\{(.*?)\}', contenido, re.DOTALL)
    preguntas_procesadas = []

    for titulo, opciones in preguntas:
        titulo_limpio = titulo.replace(':: ', '').strip()
        # Transformamos el formato multirespuesta de %positivo% y %negativo%
        opciones = re.sub(r'~%-\d+%', '~', opciones)
        opciones = re.sub(r'~%\d+%', '=', opciones)
        # Leemos la separación entre respuestas
        opciones = re.findall(r'([=~])(.*?)\s*(?=\n|\Z)', opciones)
        # Procesamos las respuestas correctas '='
        opciones_procesadas = [(opcion.strip()[0].upper() + opcion.strip()[1:].lower(), tipo == '=') for tipo, opcion in opciones]
        preguntas_procesadas.append((titulo_limpio, opciones_procesadas))

    return preguntas_procesadas
```

Ya con un output con un formato único para todas las preguntas y respuestas, podemos manejar a posteriori cómo tratar estas.

4. Back-end del Bot (100 horas, a repartir en las diferentes implementaciones citadas a continuación)

- *Procesamiento de los títulos y opciones y gestión de respuestas del usuario (24h):* El procesamiento de la estructura planteada en la base de datos ha de ser procesada para almacenar en directorios las diferentes preguntas asociadas a sus posibles opciones y a su opción correcta, al igual que comprobar la correcta respuesta del usuario.
- *Layout de botones e interacción entre eventos (12h):* Con las preguntas y opciones procesadas, han de mostrarse en pantalla con un menú interactivo, donde el alumno elija el tema y pregunta a responder y el Bot muestre la pregunta y los botones de respuesta, los cuales deben de dar una retroalimentación al responder.
- *Guardado y cargado a la base de datos con las respuestas de los usuarios (16h):* Las respuestas almacenadas en directorios (variables en forma de vector con información dentro, incluso otros directorios anidados) se pierden en caso de para la ejecución del Bot al inicializarse vacías al ejecutar el Bot, han de guardarse en una base de datos. La gestión ha de ser sencilla para profesores sin conocimientos de tratados de estos datos, así que se exportarán a un sencillo csv. Este csv además deberá ser cargado al ejecutar el Bot para que los directorios carguen las respuestas almacenadas en otras sesiones.
- *Funciones para obtener métricas de las respuestas de los usuarios (12h):* Diferentes funciones para realizar el estudio de las respuestas de los alumnos para posterior ejecución y evaluación por el profesor.

- *Comandos de gestión activa del Bot (20h):* El alumno puede ejecutar ciertos comandos que aparecen en un menú al interactuar con el bot. El profesor tiene comandos adicionales para seguir las métricas importantes para hacer un seguimiento de la evaluación individualizada y de manera colectiva con métricas de interés como la media, intentos realizados, ranking de puntuaciones, etc.
- *Gestión de progreso del usuario (16h):* Aprovechando la base de datos, podemos llevar un conteo de las preguntas no respondidas, aquellas que sí lo han sido, tanto de manera correcta como incorrecta, para que el alumno navegue por los menús del Bot sabiendo qué le falta por completar o qué debe revisar.

Competencias adquiridas: Programación back-end avanzada en Python uso de APIs de servicios web, desarrollo de aplicaciones ligadas a bases de datos, tratamiento de datos.

5. Despliegue del servidor que aloja el Bot (20h)

El Bot está alojado temporalmente en un ordenador antiguo, de cara al curso que viene (curso lectivo 2024/25), estará corriendo 24/7 en un servidor. Ahora está encendido para pruebas internas.

Competencias adquiridas: Programación en Python, despliegue de un servidor.

Desarrollo de las competencias adquiridas:

Desarrollo de Conocimientos Avanzados y Pensamiento Crítico: Este proyecto me ha permitido desarrollar un conocimiento avanzado y un pensamiento crítico en la implementación del back-end de una aplicación. Un bot es, en esencia, una aplicación web donde el front-end es gestionado por la plataforma Telegram. Este enfoque ha sido fundamental para comprender la importancia de la separación de responsabilidades en el desarrollo de software.

Control de Versiones con Git y GitHub: He adquirido habilidades significativas en el uso de herramientas de control de versiones como Git y plataformas de colaboración como GitHub. Este conocimiento es invaluable para el día a día de un desarrollador, permitiéndome llevar un registro detallado de los cambios, colaborar de manera efectiva con otros desarrolladores y mantener un desarrollo más estructurado y controlado del código.

Integración de APIs y Manejo de Eventos Asíncronos: A través de este proyecto, he aprendido a integrar de manera efectiva APIs y a manejar eventos asíncronos, lo cual es crucial para el desarrollo de aplicaciones que interactúan con servicios externos en tiempo real.

Diseño de Interfaces de Usuario en Plataformas de Mensajería: La creación de interfaces interactivas utilizando botones en Telegram me ha proporcionado una comprensión profunda del diseño de experiencias de usuario intuitivas en plataformas de mensajería.

Gestión de Proyectos y Trabajo en Equipo: Además de las habilidades técnicas, este proyecto ha fortalecido mis capacidades en la gestión de proyectos y el trabajo en equipo. La coordinación con compañeros y profesores, así como la gestión de plazos y entregables, ha sido una experiencia valiosa para mi desarrollo profesional.

Aplicación de Conocimientos en un Entorno Real: La posibilidad de aplicar conocimientos teóricos adquiridos durante mis estudios y proyectos en un proyecto práctico ha sido extremadamente enriquecedora.