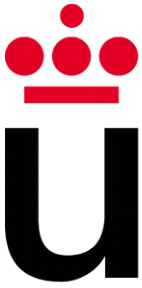

Universidad
Rey Juan Carlos

INGENIERÍA EN TECNOLOGÍAS DE LA TELECOMUNICACIÓN

Curso Académico 2024/2025

Trabajo Fin de Grado

# ANÁLISIS DE PATRONES DE ACTIVIDAD DE USUARIOS EN UNA PLATAFORMA WEB PARA DESARROLLO DE PROYECTOS DE ROBÓTICA

Autor/a : Alejandro Aguilera López
Tutor/a : José Felipe Ortega Soto

# Trabajo Fin de Grado

Título del Trabajo con Letras Capitales para Sustantivos y Adjetivos

**Autor/a :** Alejandro Aguilera López
**Tutor/a :** José Felipe Ortega Soto

La defensa del presente Proyecto Fin de Grado/Máster se realizó el día 3 de de 20XX, siendo calificada por el siguiente tribunal:

**Presidente:**

**Secretario:**

**Vocal:**

y habiendo obtenido la siguiente calificación:

**Calificación:**

Móstoles/Fuenlabrada, a de de 20XX

*Aquí normalmente*
*se inserta una dedicatoria corta*

# Agradecimientos

Aquí vienen los agradecimientos...

Hay más espacio para explayarse y explicar a quién agradeces su apoyo o ayuda para haber acabado el proyecto: familia, pareja, amigos, compañeros de clase...

También hay quien, en algunos casos, hasta agradecer a su tutor o tutores del proyecto la ayuda prestada...

# Resumen

Aquí viene un resumen del proyecto. Ha de constar de tres o cuatro párrafos, donde se presente de manera clara y concisa de qué va el proyecto. Han de quedar respondidas las siguientes preguntas:

- ¿De qué va este proyecto? ¿Cuál es su objetivo principal?
- ¿Cómo se ha realizado? ¿Qué tecnologías están involucradas?
- ¿En qué contexto se ha realizado el proyecto? ¿Es un proyecto dentro de un marco general?

Lo mejor es escribir el resumen al final.

# Summary

Here comes a translation of the “Resumen” into English. Please, double check it for correct grammar and spelling. As it is the translation of the “Resumen”, which is supposed to be written at the end, this as well should be filled out just before submitting.

# Índice general

# Índice de figuras

# Índice de fragmentos de código

# Capítulo 1

# Introducción

Debido al avance de las tecnologías, la robótica se encuentra en constante evolución, lo que hace que cada vez la podamos ver en más sectores, desde el mundo laboral hasta en nuestro día a día. Sin embargo, el aprendizaje de la robótica es un terreno complejo, ya que para poder empezar en este campo se requiere la instalación y configuración de distintas herramientas y entornos de programación, los cuales pueden resultar bastante enrevesados, en especial a nuevos usuarios que quiere empezar en el mundo de la robótica.

Otro gran problema del aprendizaje de la robótica es el coste de un robot en el cual ir probando nuestros códigos, ya que a diferencia de otras materias la robótica es un campo que su aprendizaje es a base de prueba y error.

Este trabajo de fin de grado gira entorno a UNIBOTICS una plataforma web, que nace como solución a estos problemas. UNIBOTICS permite a los usuarios acceder a ejercicios interactivos de robótica, en los cuales podrán programar robots y poder simularlos en escenarios 3D sin la necesidad de tener el robot de forma física y sin tener que instalar ningún tipo de programa.

Sin embargo, a pesar de sus ventajas, aún existen áreas de UNIBOTICS que podrían mejorarse. Actualmente, todos los datos de usuarios desde los códigos de programación, tiempo de actividad, tiempo de resolución de actividades y muchos otros datos más, se recogen en una base de datos del tipo PostgreSQL, pero estos datos únicamente se almacenan y no se les saca todo el potencial que podrían tener.

Dada esta situación, este proyecto quiere proporcionar una API que permita extraer y procesar estos datos. Además, se diseñarán unas Dashboards interactivas, donde se podrán ver de forma clara y detallada toda la información obtenida a través de la API.

Con la implementación de estas Dashboards se busca que los profesores puedan ver de

forma rápida y sencilla el trabajo realizado por sus alumnos, lo cual les podrá facilitar la enseñanza a los alumnos de robótica.

## 1.1 Objetivos

Esto es una sección, que es una estructura menor que un capítulo.

Por cierto, a veces me comentáis que no os compila por las tildes. Eso es un problema de codificación. Al guardar el archivo, guardad la codificación de "ISO-Latin-1" a "UTF-8" (o viceversa) y funcionará.

### 1.1.1 Objetivo general

Mi Trabajo Fin de Grado consiste en crear de una herramienta que permita extraer datos de la plataforma web UNIBOTICS y crear unos dashboards interactivos para representar esa información

### 1.1.2 Objetivos específicos

Los objetivos específicos se pueden entender como las tareas en las que se ha desglosado el objetivo general. Y, sí, también vienen en infinitivo.

Lo mejor suele ser utilizar una lista no numerada, como sigue:

- Un objetivo específico.
- Otro objetivo específico.
- Tercer objetivo específico.
- ...

## 1.2 Planificacion

Rellenar alfinal

## 1.3 Estructura del documento

Es conveniente que incluyas una descripción de lo que te ha llevado realizar el trabajo. Hay gente que añade un diagrama de GANTT. Lo importante es que quede claro cuánto tiempo has consumido en realizar el TFG/TFM (tiempo natural, p.ej., 6 meses) y a qué nivel de esfuerzo (p.ej., principalmente los fines de semana).

## 1.4 Estructura de la memoria

Por último, en esta sección se introduce a alto nivel la organización del resto del documento y qué contenidos se van a encontrar en cada capítulo.

- En el primer capítulo se hace una breve introducción al proyecto, se describen los objetivos del mismo y se refleja la planificación temporal.
- En el siguiente capítulo se describen las tecnologías utilizadas en el desarrollo de este TFM/TFG (Capítulo 2).
- En el capítulo 3 Se describe el proceso de desarrollo de la herramienta ...
- En el capítulo 4 Se presentan las principales pruebas realizadas para validación de la plataforma/herramienta...(o resultados de los experimentos efectuados).
- Por último, se presentan las conclusiones del proyecto así como los trabajos futuros que podrían derivarse de éste (Capítulo 5).

# Capítulo 2

# Tecnologías

En este capitulo describiré las tecnologías utilizadas para este proyecto.

## 2.1 Python

Python es un lenguaje de programación de alto nivel, interpretado, dinámico y fuertemente tipado. Python es un lenguaje multiparadigma, lo que quiere decir que permite desarrollar software utilizando distintos enfoques como la programación orientada a objetos, la programación funcional y la programación imperativa.

El principal motivo por el cual he elegido Python como lenguaje de programación para este trabajo es que la plataforma UNIBOTICS está desarrollada en este lenguaje, por lo que usar otro sería ineficiente y una complicación innecesaria. Además, Python destaca por su simplicidad y legibilidad, lo que facilita el desarrollo y mantenimiento del código, su gran comunidad que ofrece soporte constante y su gran variedad de bibliotecas en las cuales se apoya este proyecto. A continuación se enumeran y se detallan el funcionamiento de cada una:

### 2.1.1 Pandas

Pandas es una de las herramientas más poderosas para la manipulación y análisis de datos en Python. Esta biblioteca se diseñó para hacer que la limpieza, transformación y análisis de datos sean rápidos y eficientes.

Una de las razones por las que he elegido pandas en este proyecto es su capacidad para

manipular datos de manera flexible y eficiente. Sus estructuras principales, Series y DataFrame, facilitan la organización y el acceso a la información de manera intuitiva. La Series actúa como un array unidimensional con etiquetas asociadas, mientras que el DataFrame es una tabla bidimensional que permite realizar operaciones similares a las de bases de datos o herramientas como Excel.

En el contexto de este proyecto, pandas es la mejor opción porque la plataforma en la que se desarrollará ya está basada en Python y requiere una gestión eficiente de datos tabulares. Además, como se menciona en *Python for Data Analysis*, 3rd Edition, pandas no solo facilita la manipulación de datos, sino que también nos permite enfocarnos en la interpretación y visualización de la información, en lugar de perder tiempo en tareas repetitivas de procesamiento.

### 2.1.2 Dash

Dash es una herramienta de Python que permite crear aplicaciones web interactivas de forma sencilla.

En este proyecto, lo he utilizado porque facilita la creación de gráficos atractivos e interactivos para visualizar los datos de la plataforma UNIBOTICS. Con Dash, puedo generar visualizaciones que permiten a los usuarios interactuar con los datos de manera intuitiva, como hacer filtros o explorar diferentes métricas. Esto es ideal para representar información compleja de forma clara y visualmente atractiva. Además, Dash se integra muy bien con otras bibliotecas de Python, lo que hace que sea fácil crear gráficos de todo tipo, desde barras hasta mapas interactivos.

### 2.1.3 Psycopg2

Psycopg2 es una librería de Python que permite conectar aplicaciones Python con bases de datos PostgreSQL. Es ampliamente utilizada debido a su eficiencia y facilidad de uso para interactuar con bases de datos.

En este proyecto, he utilizado Psycopg2 para ejecutar consultas SQL directamente desde Python, lo que permite extraer, insertar y modificar datos de manera eficiente. Esta librería ofrece una interfaz sencilla y rápida para gestionar la conexión con la base de datos y ejecutar comandos SQL de forma segura, evitando vulnerabilidades como la inyección de SQL. Además, su integración con otras bibliotecas de Python, como Pandas 2.1.1, facilita la manipulación y análisis de los datos obtenidos desde la base de datos, lo que hace que sea más fácil la integración con Dash 2.1.2.

## 2.2 PostgreSQL

PostgreSQL es un sistema de base de datos muy seguro y popular que se utiliza para guardar y organizar grandes cantidades de datos. Es de código abierto, lo que significa que cualquiera puede usarlo y adaptarlo según sus necesidades.

Una de las razones por las que es tan utilizado es que permite hacer consultas complejas de manera rápida y segura, asegurando que los datos se mantengan intactos y bien organizados. En el caso de Unibotics, toda la información relacionada con los usuarios y sus interacciones en la plataforma se almacena en una base de datos PostgreSQL. Aunque se hablará más adelante sobre cómo está estructurada esta base de datos 3.1, lo importante es que PostgreSQL ofrece una forma eficiente y segura de gestionar toda esa información.

## 2.3 Docker

Docker es una plataforma de virtualización ligera basada en contenedores que permite empaquetar una aplicación junto con todas sus dependencias en una única unidad portátil. Cada contenedor comparte el núcleo del sistema operativo, pero se ejecuta de forma aislada, lo que garantiza que el entorno interno sea siempre el mismo, independientemente de la máquina donde se despliegue.

En este proyecto hemos utilizado Docker principalmente para levantar la base de datos PostgreSQL de Unibotics en un contenedor independiente. De esta manera, pude iniciar, detener o reiniciar la base de datos con un solo comando, sin necesidad de instalar PostgreSQL en el sistema anfitrión ni preocuparme por posibles conflictos con otras versiones de la base de datos. Además, al usar Docker, tengo la garantía de que el entorno de ejecución es idéntico en mi máquina local y en cualquier otro entorno de pruebas o producción donde se despliegue la plataforma.

## 2.4 Entorno de desarrollo: PyCharm

PyCharm es un Integrated Development Enviroment (Entorno de Desarrollo Integrado) (IDE) dedicado concretamente a la programación en Python y desarrollado por la compañía checa JetBrains.

Proporciona análisis de código, un depurador gráfico, una consola de Python integrada, control de versiones y, además, soporta desarrollo web con Django. Todas estas características lo convierten en un entorno completo e intuitivo, idóneo para el desarrollo de

proyectos académicos como el que nos ocupa.

## 2.5 Redacción de la memoria: LaTeX/Overleaf

LaTeX es un sistema de composición tipográfica de alta calidad que incluye características especialmente diseñadas para la producción de documentación técnica y científica. Estas características, entre las que se encuentran la posibilidad de incluir expresiones matemáticas, fragmentos de código, tablas y referencias, junto con el hecho de que se distribuya como software libre, han hecho que LaTeX se convierta en el estándar de facto para la redacción y publicación de artículos académicos, tesis y todo tipo de documentos científico-técnicos.

Por su parte, Overleaf es un editor LaTeX colaborativo basado en la nube. Lanzado originalmente en 2012, fue creado por dos matemáticos que se inspiraron en su propia experiencia en el ámbito académico para crear una solución satisfactoria para la escritura científica colaborativa.

Además de por su perfil colaborativo, Overleaf destaca porque, pese a que en LaTeX el escritor utiliza texto plano en lugar de texto formateado (como ocurre en otros procesadores de texto como Microsoft Word, LibreOffice Writer y Apple Pages), éste puede visualizar en todo momento y paralelamente el texto formateado que resulta de la escritura del código fuente.

# Capítulo 3

# Arquitectura

Aquí viene todo lo que has hecho tú (tecnológicamente). Puedes entrar hasta el detalle. Es la parte más importante de la memoria, porque describe lo que has hecho tú. Eso sí, normalmente aconsejo no poner código, sino diagramas.

## 3.1 Arquitectura general

Como ya dije antes, UNIBOTICS es una plataforma web diseñada con el fin de facilitar el aprendizaje práctico de la robótica, proporcionando a los estudiantes un entorno en el que pueden encontrar ejercicios y escenarios interactivos sin la necesidad de instalar o configurar entornos complejos de software.

Esta plataforma proporciona diferentes herramientas para trabajar con robots. La herramienta que gestiona el software robótico requerido, es RADI (Robotics Academy Docker Image). Se trata de unos contenedores Docker especiales en los que se llevan a cabo la ejecución de ejercicios y las simulaciones robóticas.

Estos contenedores integran ROS2 y Gazebo, además de otras dependencias necesarias para poder ejecutar el código de los usuarios. ROS2 (Robot Operating System 2) ofrece un middleware estándar para la programación robótica, permitiendo comunicar diferentes nodos y controlar así el flujo de datos entre los distintos sensores. Gazebo aporta un entorno de simulación física realista que proporciona entornos 3D donde se puede simular el comportamiento de los robots con el código previamente programado por el usuario.

Para comunicar el contenedor RADI con el navegador, se utiliza otra herramienta llamada RAM (Robot Application Manager), que actúa como puente entre el navegador y el contenedor. De esta forma, cuando el usuario modifica o ejecuta código, RAM lo recibe, lo

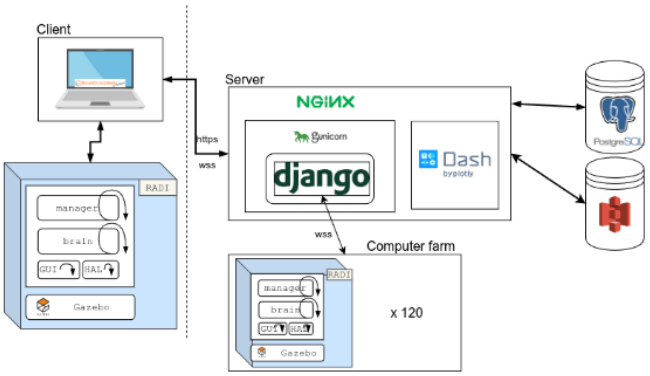


Figura 3.1: Estructura de Unibotics.

ejecuta en el contenedor y devuelve los resultados correspondientes. Estos resultados pueden ser desde imágenes del entorno y la posición del robot hasta cualquier dato relevante para el ejercicio.

La Figura 3.1 ilustra esta arquitectura, mostrando la relación entre los distintos componentes de la plataforma, desde la interacción del usuario a través del navegador hasta la gestión de simulaciones y datos en los servidores.

Para almacenar toda esta cantidad de datos que ofrece cada simulación, además de otros datos de Unibotics como los usuarios, datos estadísticos de ejercicios y datos del desempeño del usuario, se emplea una base de datos relacional del tipo PostgreSQL, lo que facilita la gestión y el análisis futuro de la información, de la cual obtendremos los datos a través de nuestra API para así poder representarlos en los dashboards.

Respecto a la parte del usuario, la interacción con esta plataforma se realiza a través de un navegador web, mediante el cual el usuario accede a una interfaz construida con React, donde se encuentran los ejercicios y escenarios interactivos.

Para implementar todas estas herramientas, UNIBOTICS se apoya en Django, un framework de alto nivel orientado al desarrollo web en Python, especialmente a la creación de aplicaciones web complejas. Gracias a Django, es posible estructurar de forma clara la lógica interna de la plataforma, lo que permite relacionar usuarios con ejercicios y almacenar información sobre las sesiones de trabajo.

Además de Django, emplea Nginx y Gunicorn, herramientas que mejoran la eficiencia de la plataforma. Estas permiten gestionar un gran volumen de usuarios de manera simultánea, asegurando que el sistema opere sin interrupciones.

La base de datos de Unibotics está organizada para almacenar toda la información necesaria para el funcionamiento de la plataforma, desde los datos de los usuarios hasta los resultados de las simulaciones. Aunque toda esta información se encuentra en una úni-

ca base de datos, está organizada en diferentes tablas que cubren varias áreas clave de la plataforma.

Por ejemplo, la información relacionada con los ejercicios y los universos de simulación está almacenada en tablas como exercises y exercise_universes, las cuales permiten asociar ejercicios a los distintos entornos de simulación disponibles. Esto asegura que un mismo ejercicio pueda ejecutarse en varios universos, proporcionando flexibilidad en las simulaciones.

También está la información sobre los robots y los mundos de simulación, que se guarda en tablas como robots y worlds. Estas tablas contienen detalles sobre la configuración de los robots y los mundos de simulación, lo que permite a la plataforma gestionar los recursos necesarios para ejecutar los ejercicios en el entorno correcto.

Por otro lado, las tablas de logs, como log_session y log_exercises, almacenan los registros más importantes sobre la actividad de los usuarios. Estos logs incluyen información detallada sobre la duración de las sesiones de los usuarios, los ejercicios realizados y las fechas en las que se realizaron. Estos datos son fundamentales para generar los dashboards y análisis sobre el comportamiento de los usuarios en la plataforma.

Además, la base de datos gestiona también los datos clave sobre los usuarios, los permisos de acceso y las máquinas de granja (contenedores Docker) que ejecutan los ejercicios. Aquí se guardan los detalles de los usuarios y cómo se relacionan con los ejercicios a los que tienen acceso, así como el estado de las máquinas que se utilizan para ejecutar los ejercicios.

En la Figura 3.2, se muestra el diagrama de entidad-relación (ER) que ilustra cómo todas estas tablas están conectadas entre sí y cómo se organiza la información en la base de datos. Este diagrama es esencial para comprender cómo fluye la información dentro del sistema y cómo se gestionan las relaciones entre los diferentes elementos.

Figura 3.2: Diagrama entidad-relación.

Finalmente, en el apendice A 6, encontrarás ejemplos de consultas SQL que te permitirán obtener información directamente de las tablas de logs y realizar análisis sobre la actividad de los usuarios y su interacción con los ejercicios.

## 3.2 Despliegue Local en Linux

Para poder probar y depurar todos los componentes de Unibotics de forma sencilla, instalé un entorno completo en mi máquina Linux (despliegue "D1"). En primer lugar, creé un entorno virtual de Python 3.8 para aislar las dependencias del proyecto sin afectar al sistema. A continuación, cloné el repositorio de `unibotics-webserver` junto con sus submódulos y ejecuté la instalación de los paquetes necesarios desde el fichero `utils/requirements.txt`.

Dado que Unibotics utiliza PostgreSQL como base de datos, levanté un contenedor Docker con esa imagen. Esto me permitió iniciar, detener o reiniciar la base de datos con total comodidad y sin instalar nada en el sistema. Tras arrancar el contenedor, apliqué los dumps de Robotics Infrastructure y Robotics Academy para cargar los universos y ejercicios, ejecuté las migraciones de Django para crear el resto de tablas y, finalmente, importé datos de prueba de cursos pasados para disponer de una muestra amplia sobre la que trabajar.

En la parte de frontend, configuré los enlaces simbólicos que Webpack necesita para localizar los componentes de RoboticsAcademy, instalé las dependencias de Node y arrancé el modo desarrollo de Webpack. Con ello, cualquier cambio en el código React se recompila al vuelo y se refleja en el navegador.

Con este entorno ya puedo levantar la plataforma completa en local, probar cada cambio y comprobar al instante cómo funciona sin miedo a romper nada en producción. Esto me proporciona una zona de pruebas segura donde experimentar libremente y acelerar el ciclo de desarrollo y depuración.

# Capítulo 4

# Experimentos y validación

En esta sección expondré los distintos dashboards que he desarrollado con el objetivo de poder visualizar y analizar la información que está almacenada en la base de datos de Unibotics.

Cada dashboard se centra en distintas características del comportamiento del usuario, para su desarrollo he utilizado el framework Dash de Python junto con Plotly para la visualización gráfica y la información se extrae directamente desde la base de datos PostgreSQL mediante consultas SQL específicas.

En los siguientes apartados expondré de forma individual cada uno de los dashboard desarrollados, explicando su propósito, los datos que representa, el porqué ese tipo de visualización y las diferentes conclusiones que pueden obtenerse sobre el comportamiento de los usuarios.

A continuación, se presentan las tablas SQL de las que se extrae la información utilizada en los dashboards.

**Tabla 4.1**
*Tablas SQL utilizadas de la base de datos de Unibotics.*

| Tabla | Descripción |
|---|---|
| Log_session | Contiene información sobre las sesiones iniciadas por los usuarios. Entre los campos más relevantes se encuentran el nombre del usuario, la fecha y hora de inicio de la sesión, la duración total en segundos y el país desde el cual se conecta el usuario. |
| Log_exercises | Contiene información sobre los accesos de los usuarios a los distintos ejercicios disponibles. Entre los campos más relevantes se encuentran el nombre del usuario, el nombre del ejercicio al que accede, la duración de la interacción, la fecha y hora de inicio, así como otros datos relacionados con la actividad. |
| Common_user | Reúne información general sobre los usuarios registrados en la plataforma. Aunque cuenta con múltiples campos, el más relevante para este trabajo es el género del usuario. |
| Exercises | Contiene un listado con los distintos ejercicios disponibles en la plataforma |

*Nota.* La base de datos de Unibotics contiene mas tablas como vimos en 3.1, pero estas son las que uso para realizar los Dashboards.

## 4.1 DASHBOARD 1A

El Dashboard 1A nos permite analizar la duración total que un usuario ha dedicado a cada ejercicio de la plataforma.

Para poder obtener la información de duración por ejercicio, el dashboard se conecta directamente a la base de datos PostgreSQL de Unibotics mencionada anteriormente. Una vez establecida la conexión, la aplicación Dash ejecuta una consulta SQL parametrizada con el nombre de usuario introducido en el input.

La consulta SQL aprovecha la funcionalidad de agrupación y agregación de SQL: se filtran los registros por el usuario seleccionado y luego se agrupan por el identificador de ejercicio, calculando la suma de todos los tiempos asociados,la informacion se obtiene de la tabla log_session. La consulta tiene esta forma:

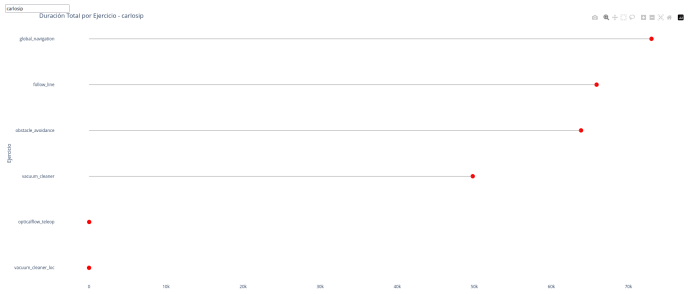


Figura 4.1: Dash 1a.

aqui pego codigo sigp sin ser capaz no se porque no lo veo

De este modo, la propia consulta realiza el cálculo de la duración total (en segundos) acumulada para cada ejercicio realizado por el usuario indicado. La cláusula GROUP BY agrupa las filas que tienen el mismo valor de ejercicio y aplica la función agregada SUM() para sumar los datos de cada grupo. En otras palabras, si el usuario ha realizado varias sesiones o intentos en un mismo ejercicio, todos esos tiempos se suman en un único resultado por ejercicio. Esta consulta retorna una tabla con dos columnas: el nombre de cada ejercicio y la suma de la duración total que el usuario ha invertido en él. Dichos resultados se cargan en un DataFrame de Pandas para su manipulación en la aplicación y posteriormente se emplean para generar la visualización.

Desde el punto de vista del usuario, el usuario ve un dashboard y un cuadro de texto donde poder escribir el nombre del usuario que deseas visualizar su duración total por ejercicio, una vez escrito un nombre de usuario carga el dashboard con la información de ese usuario.

El dashboard presenta los datos mediante un gráfico horizontal de líneas con puntos, también conocido como gráfico tipo lollipop. Este tipo de visualización es una variación de un diagrama de barras: la barra tradicional se transforma en una línea delgada y el valor se resalta con un marcador circular al final.

Explicar porque elegimos este tipo de gráfico (tambien importante que lo ordenamos de arriba a abajo porque era mejor visualmente)

La información presentada en el Dashboard 1A permite una rápida interpretación comparativa de la dedicación del usuario a los distintos ejercicios, gracias a ello podemos obtener varias conclusiones.

Identificación de ejercicios más y menos trabajados: Las longitudes de las líneas revelan de un vistazo cuáles son los ejercicios en los que el usuario ha invertido más tiempo y cuáles menos.

Detección de patrones de esfuerzo o dificultad: Una duración total muy elevada en cier-

to ejercicio podría indicar que el usuario tuvo dificultades significativas con ese ejercicio o que requirió múltiples intentos prolongados para completarlo,también podría significar que el ejercicio era muy extenso o el usuario dedicó tiempo extra explorando más allá de lo mínimo requerido. Por el contrario, ejercicios con duraciones muy bajas pueden sugerir que el usuario los completó rápidamente (quizá por ser sencillos o ya conocidos) o incluso que los abandonó pronto. En cualquier caso, los extremos en la distribución de tiempos señalan ejercicios que merecen una atención particular al evaluar el progreso del usuario.

Distribución del tiempo de aprendizaje: El conjunto de todas las duraciones permite ver cómo el usuario ha distribuido su tiempo de aprendizaje en la plataforma. Un perfil equilibrado mostraría barras (líneas) de duraciones relativamente similares entre ejercicios, mientras que un perfil más desequilibrado tendría unos pocos ejercicios dominando la mayor parte del tiempo total.

## 4.2 DASHBOARD 1C

Este dashboard tiene como objetivo mostrar la duración promedio de las sesiones de los usuarios de Unibotics, agrupadas por país. A través de un mapa mundial interactivo, se representa gráficamente qué países presentan una mayor o menor media de tiempo por sesión, permitiendo identificar patrones geográficos en el uso de la plataforma.

La información utilizada en este dashboard se obtiene desde la tabla log_session,la consulta SQL utilizada agrupa los datos por país, calcula la suma total de duración y el número total de sesiones por país, y posteriormente se calcula en Python la duración promedio dividiendo ambos valores. La consulta tiene esta forma:

Aqui pego el codigo (pero nose como hacerlo bien)

A partir de esta consulta se genera un DataFrame de Pandas con el país y su respectiva duración promedio de sesión, que luego se utiliza para crear el gráfico. Este cálculo permite representar una métrica más equilibrada que la duración total (que podría estar sesgada por la cantidad de usuarios en un país), y proporciona una mejor medida de cómo interactúan los usuarios con la plataforma en promedio.

El dashboard utiliza un mapa coroplético (choropleth map), generado mediante Plotly Express, en el que cada país se colorea en función de la duración promedio de sesión. Cuanto mayor es la duración, más intenso es el color asignado, siguiendo una escala cromática progresiva desde tonos claros hasta colores más saturados. En el ejemplo mostrado (foto anterior), se emplea una paleta de colores continua (aclarar que color dejo) personalizada para resaltar visualmente los valores extremos.

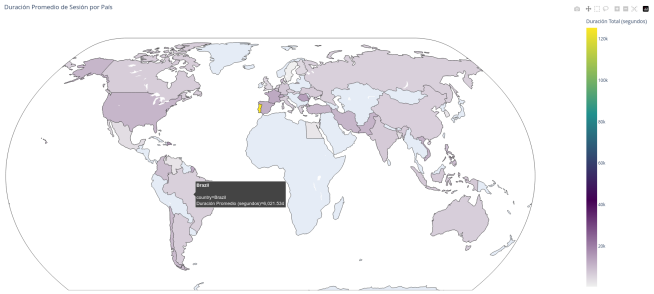


Figura 4.2: Dash 1c.

Explicar porque elegimos este tipo de gráfico

El dashboard ofrece información útil tanto a nivel técnico como estratégico.España aparece con una alta duración promedio, lo que sugiere un uso intensivo y posiblemente más prolongado por sesión, frente a países donde el uso puede ser más breve o esporádico.Sin embargo, países con bajas duraciones promedio pueden estar reflejando una adopción más reciente, una menor familiaridad con la plataforma o simplemente menor disponibilidad de tiempo por parte de los usuarios.

Desde el punto de vista de la administración de Unibotics, este dashboard puede ayudar a identificar mercados donde la plataforma tiene un mayor impacto o uso sostenido, lo cual puede influir en decisiones de soporte, localización de contenidos o expansión. Por el contrario, tambien puede ayudar a detectar regiones con bajo uso relativo, lo que puede motivar estrategias de difusión, colaboración educativa o mejora del acceso.

## 4.3 DASHBOARD 2A

Este dashboard tiene como objetivo mostrar la duración total acumulada de todas las sesiones de usuarios de Unibotics, agrupadas por país. A través de un mapa interactivo, permite observar en qué regiones geográficas se ha registrado un mayor tiempo de uso en conjunto, proporcionando una visión global de la distribución de la actividad en la plataforma.

La información utilizada en este dashboard se obtiene desde la tabla log_session, la consulta SQL que se utiliza agrupa los datos por país y calcula la suma total de la duración de todas las sesiones correspondientes a cada país. La consulta tiene esta forma:

Aqui pego el codigo (pero nose como hacerlo bien)

A diferencia de otros dashboards que analizan promedios, en este caso se calcula simplemente la acumulación del tiempo total de uso de la plataforma por país. Esto permite

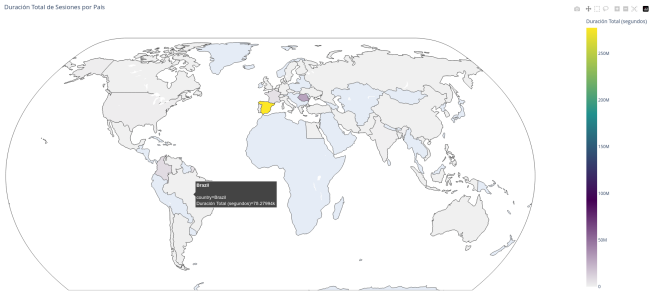


Figura 4.3: Dash 2a.

identificar los países donde el uso absoluto de Unibotics ha sido más intenso.

Una vez obtenidos los resultados, se cargan en un DataFrame de Pandas con el pais y su respectiva duración total de sesiones y se utilizan como base para la representación gráfica en el mapa.

El dashboard utiliza un mapa coroplético (choropleth map), generado mediante Plotly Express, en el que cada país se colorea en función de la duración total de sesión. Cuanto mayor es la duración, más intenso es el color asignado, siguiendo una escala cromática progresiva desde tonos claros hasta colores más saturados. En el ejemplo mostrado (foto anterior), se emplea una paleta de colores continua (aclarar que color dejo) personalizada para resaltar visualmente los valores extremos.

Explicar porque elegimos este tipo de gráfico

Del análisis del mapa (ver figura), se pueden obtener conclusiones como:

España destaca con el mayor volumen de tiempo total de sesiones, lo que refleja su fuerte implantación y uso intensivo de la plataforma, probablemente por ser el país de origen o principal mercado de Unibotics mientras que otros países europeos o latinoamericanos presentan niveles más bajos de duración acumulada, lo que puede interpretarse como una menor penetración o un uso más reciente de la plataforma en esas regiones.

Tambien se observan diferencias geográficas claras, algunos continentes como África o Asia presentan en general un uso más bajo, salvo excepciones, lo que podría señalar oportunidades de expansión o barreras de adopción actuales.

Desde el punto de vista de la gestión de Unibotics, este dashboard ayuda a ver en qué países la plataforma tiene más presencia. Esta información permite decidir mejor dónde hacer campañas, traducir contenidos o invertir en mejorar el servicio. Además, señala qué zonas podrían beneficiarse más de nuevas colaboraciones educativas

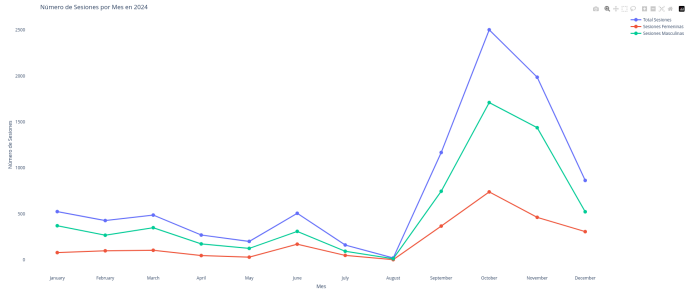


Figura 4.4: Dash 2c.

## 4.4 DASHBOARD 2C

Este dashboard muestra la evolución mensual del número de sesiones iniciadas en Unibotics a lo largo del año 2024, diferenciando también entre usuarios masculinos y femeninos. El objetivo es analizar cómo varía el uso de la plataforma mes a mes y si existen diferencias significativas por género.

Los datos provienen de las tablas log_session y common_user,la consulta SQL utilizada realiza una agregación mensual,teniendo en cuenta: el total de sesiones iniciadas en cada mes,el número de sesiones realizadas por usuarias (género femenino)y el número de sesiones realizadas por usuarios (género masculino). La consulta tiene esta forma:

Aqui pego el codigo (pero nose como hacerlo bien)

Una vez extraídos, los datos se procesan en Pandas para asegurarse de que los meses estén correctamente ordenados de enero a diciembre, y se formatea el nombre de los meses.

El dashboard utiliza un gráfico de líneas para representar la evolución de las sesiones durante el año. Se incluyen tres líneas:

Total de sesiones (en azul). Sesiones femeninas (en rojo). Sesiones masculinas (en verde).

Cada punto en el gráfico representa el número de sesiones registradas en ese mes para cada categoría.

Explicar porque elegimos este tipo de gráfico

Explicar conclusiones porque de aqui mas alla de que en verano nadie hace nada no se que mas decir.

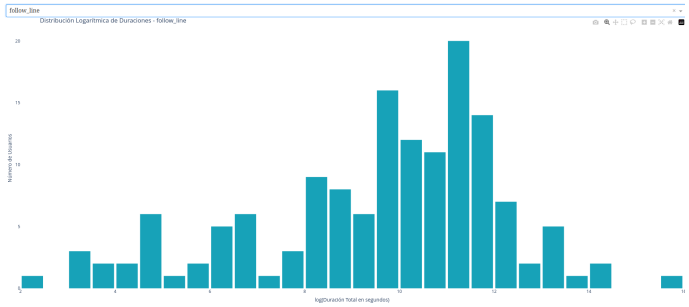


Figura 4.5: Dash 3a.

## 4.5 DASHBOARD 3A

Este dashboard analiza cómo se distribuyen los usuarios de Unibotics según el tiempo total que han dedicado a un ejercicio específico. Utiliza una escala logarítmica para poder representar mejor las diferencias en las duraciones, que pueden ser muy grandes entre unos usuarios y otros.

Los datos provienen de las tablas log_exercises y exercises.El usuario ve un desplagable de ejercicios que permite seleccionar un ejercicio concreto mediante un desplegable. Una vez seleccionado, se ejecuta una consulta SQL que suma el tiempo total que cada usuario ha dedicado a ese ejercicio. La consulta tiene esta forma:

Aqui pego el codigo (pero nose como hacerlo bien)

Después de obtener los datos, se aplica una transformación logarítmica al valor de duración total. Esto se hace para reducir la dispersión de los datos: como hay usuarios que pueden haber pasado desde unos pocos segundos hasta varias horas en un mismo ejercicio, el uso de una escala logarítmica permite ver toda la distribución de una forma más equilibrada.

El dashboard muestra los datos en un histograma, donde el eje horizontal representa el logaritmo de la duración total (en segundos) y el eje vertical representa el número de usuarios que tienen una duración dentro de cada rango.

Cada barra del histograma muestra cuántos usuarios cayeron en ese rango de duraciones. La escala logarítmica en el eje X permite visualizar mejor los datos, ya que de otra manera la mayoría de los usuarios quedarían agrupados en duraciones bajas y sería difícil distinguir diferencias.

Explicar porque elegimos este tipo de gráfico

El análisis del histograma permite entender cómo se comportan los usuarios al realizar un determinado ejercicio. Al observar la distribución de los tiempos, podemos ver si la

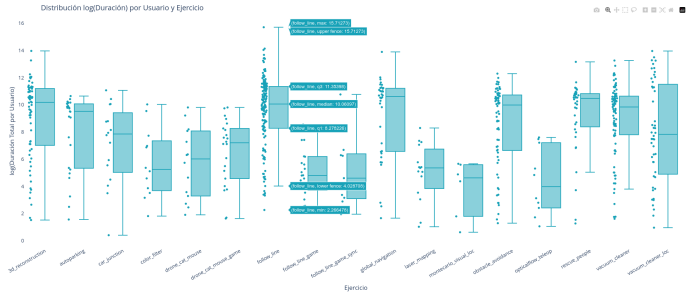


Figura 4.6: Dash 3b.

mayoría de los usuarios completan el ejercicio en un tiempo parecido o si existen grandes diferencias entre ellos.

Cuando los tiempos se agrupan principalmente en valores bajos, suele indicar que el ejercicio es sencillo o que muchos usuarios no llegaron a dedicarle demasiado tiempo, quizás porque lo abandonaron pronto. En cambio, cuando hay una gran variedad de tiempos, podemos interpretar que el ejercicio tiene un nivel de dificultad más variable o que, por su diseño, permite a los usuarios dedicarle más tiempo explorando distintas soluciones.

Para los profesores, ofrece una forma de detectar si el ejercicio está bien equilibrado en cuanto a dificultad, o si sería necesario ajustarlo para mejorar la experiencia de aprendizaje.

## 4.6 DASHBOARD 3B

Este dashboard muestra cómo se distribuye el tiempo que cada usuario ha dedicado a distintos ejercicios de Unibotics. Utiliza diagramas de caja (boxplots) aplicando una escala logarítmica para representar mejor las diferencias entre los usuarios.

Los datos se obtienen de la tabla log_exercises. Se agrupa la información por nombre de ejercicio y por usuario, sumando la duración total que cada uno ha dedicado a cada ejercicio. Solo se consideran aquellos casos donde el tiempo total registrado es mayor que cero. La consulta tiene esta forma:

Aqui pego el codigo (pero nose como hacerlo bien)

Una vez extraídos los datos, se calcula el logaritmo de la duración total para cada usuario. Esto se hace para normalizar los valores, ya que algunos usuarios pueden tener tiempos muy pequeños y otros muy grandes, y la escala logarítmica ayuda a visualizar mejor esa variabilidad.

Cada ejercicio se representa en el eje horizontal, mientras que en el eje vertical se mues-

tra el logaritmo del tiempo total invertido por los usuarios. Para cada ejercicio, el boxplot muestra:

La mediana de las duraciones (línea central del rectángulo). El rango intercuartílico (el ancho de la caja), que representa la mayoría de los usuarios. Los valores extremos o atípicos (puntos dispersos fuera de la caja).

Gracias a este tipo de gráfico es posible visualizar de forma clara cómo varía la dedicación de los usuarios en cada ejercicio, y detectar si hay muchos usuarios que dedican tiempos muy distintos o si la mayoría se concentra en valores similares. Además, se muestran todos los puntos individuales sobre los diagramas, lo que permite ver mejor la dispersión real de los datos.

Este dashboard permite comparar de manera rápida cómo se comportan los usuarios en cada ejercicio. Se puede ver si la mayoría de los alumnos dedican un tiempo similar o si hay mucha diferencia entre unos y otros. También ayuda a detectar ejercicios en los que algunos usuarios invierten mucho más tiempo, lo que podría significar que son más complicados o que resultan especialmente interesantes. Además, permite identificar actividades donde hay muchos casos extremos, lo que podría indicar que el ejercicio no está bien planteado o que su dificultad varía demasiado entre estudiantes. Para los profesores, esta información es útil para saber qué ejercicios podrían necesitar cambios o mejoras, y para entender mejor cómo se enfrentan los alumnos a las tareas. Para quienes diseñan los contenidos, el análisis sirve para ver si los ejercicios provocan el tipo de trabajo esperado o si sería buena idea añadir ayudas o dividir las actividades.

## 4.7 DASHBOARD 3C

Este dashboard es muy similar al anterior (Dashboard 3B), ya que también utiliza diagramas de caja (boxplots) para representar los datos, pero con una diferencia importante: en lugar de analizar el tiempo total que cada usuario dedica a un ejercicio, aquí se analiza la duración de cada sesión individual.

Mientras que en el 3B (esto lo cambiare cuando esten todos) sumábamos todas las sesiones de un usuario para un ejercicio determinado, en este dashboard se estudia cada sesión por separado, sin agruparlas. Además, también se aplica una escala logarítmica para representar mejor las diferencias en los tiempos, ya que algunas sesiones pueden durar solo unos pocos segundos y otras mucho más.

Para obtener estos datos se utiliza una consulta SQL sencilla que selecciona, para cada registro de sesión, el nombre del ejercicio y la duración de esa sesión concreta:

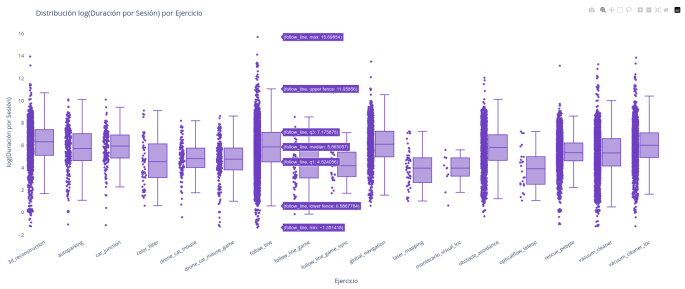


Figura 4.7: Dash 3c.

aqui pongo el codigo

El objetivo de este dashboard es ofrecer una visión más detallada sobre cómo varía el tiempo de trabajo en cada intento o sesión concreta, algo que no era visible en el Dashboard 3B, donde sólo veíamos el esfuerzo acumulado de cada usuario. Aquí podemos detectar, por ejemplo, si un ejercicio tiene sesiones generalmente cortas o si hay mucha variación entre los intentos de los distintos usuarios.

Esta representación es útil para entender mejor el comportamiento real dentro de cada ejercicio: permite ver si los estudiantes tienden a resolver los ejercicios de forma rápida o si, por el contrario, dedican mucho tiempo en intentos sucesivos. Además, puede ayudar a detectar ejercicios que generan más dispersión en el tiempo por sesión, lo que puede ser una pista sobre su dificultad o sobre la necesidad de mejorar su planteamiento o las instrucciones dadas.

## 4.8 como poner codigo

Es bastante habitual que se reproduzcan fragmentos de código en la memoria de un TFG/TFM. Esto permite explicar detalladamente partes del desarrollo que se ha realizado que se consideren de especial interés. No obstante, tampoco es conveniente pasarse e incluir demasiado código en la memoria, puesto que se puede alargar mucho el documento. Un recurso muy habitual es subir todo el código a un repositorio de un servicio de control de versiones como GitHub o GitLab, y luego incluir en la memoria la URL que enlace a dicho repositorio.

Para incluir fragmentos de código en un documento LaTeXse pueden combinar varias herramientas:

- El entorno  permite crear un marco en el que situar el fragmento de código (parecido al generado cuando insertamos una tabla o una figura). Podemos insertar también una descripción (*caption*) y una etiqueta para referenciarlo luego en el texto.

- Dentro de este entorno, se puede utilizar el paquete [1], que utiliza el paquete Python Pygments para resaltado de sintaxis (coloreando el código). Como se puede ver en el siguiente ejemplo, hay muchas opciones de configuración que permiten controlar cómo se va a mostrar el código (incluir números de línea, saltos de línea, tamaño y tipo de fuente, espaciado, código de colores para resaltado, etc.).

Código 4.1: Lectura de un fichero *.csv y tipado de datos.

Otra ventaja del entorno `listing` es que se puede generar automáticamente un índice (con entradas hiperenlazadas) de fragmentos de código, para incluirlo al comienzo del documento junto con los índices de figuras, tablas, etc.

### 4.8.1 Fuentes monoespaciadas

A veces se incluyen nombres de archivos, paquetes, etc. como texto monoespaciado, utilizando el comando LaTeX. Sin embargo, esto puede generar un problema cuando las palabras en fuente monoespaciada alcanzan el final de una línea. En ese caso, el compilador rehusa muchas veces romper la palabra y deja la línea demasiado larga respecto al resto.

Para evitar esto, especialmente en párrafos más cortos de lo habitual (como en una lista no numerada), se puede utilizar el comando , como se muestra a continuación con un ejemplo real.

- Los valores contenidos en las columnas `genres`, `spoken_languages`, `production_companies` y `production_countries`, clasificados originalmente como `np.objects`, se corresponden en realidad con listas de objetos JSON que han sido almacenadas como cadenas de caracteres. A través de la función `get_values(obj, key)` definida específicamente para ello, se transformará dicha cadena de caracteres en una lista de diccionarios a través de la función `json.loads(obj)` y se devolverá una tupla que recopile los valores de los mismos para la clave indicada, un objeto de Python mucho más manejable de cara a realizar consultas sobre el *dataset*.

[1] https://es.overleaf.com/learn/latex/Code_Highlighting_with_minted

# Capítulo 5

# Conclusiones y trabajos futuros

## 5.1 Consecución de objetivos

Esta sección es la sección espejo de las dos primeras del capítulo de objetivos, donde se planteaba el objetivo general y se elaboraban los específicos.

Es aquí donde hay que debatir qué se ha conseguido y qué no. Cuando algo no se ha conseguido, se ha de justificar, en términos de qué problemas se han encontrado y qué medidas se han tomado para mitigar esos problemas.

Y si has llegado hasta aquí, siempre es bueno pasarle el corrector ortográfico, que las erratas quedan fatal en la memoria final. Para eso, en Linux tenemos aspell, que se ejecuta de la siguiente manera desde la línea de *shell*:

## 5.2 Aplicación de lo aprendido

Aquí viene lo que has aprendido durante el Grado/Máster y que has aplicado en el TFG/TFM. Una buena idea es poner las asignaturas más relacionadas y comentar en un párrafo los conocimientos y habilidades puestos en práctica.

1. a

2. b

## 5.3 Lecciones aprendidas

Aquí viene lo que has aprendido en el Trabajo Fin de Grado/Máster.

1. Aquí viene uno.
2. Aquí viene otro.

## 5.4 Trabajos futuros

Ningún proyecto ni software se termina, así que aquí vienen ideas y funcionalidades que estaría bien tener implementadas en el futuro.

Es un apartado que sirve para dar ideas de cara a futuros TFGs/TFMs.

# Capítulo 6

# Apendice A

En este apéndice se presentan ejemplos de las consultas SQL utilizadas para extraer datos de las tablas que conforman la base de datos de Unibotics. A continuación, se describe cómo se organizan los datos en las tablas de la base de datos y se presentan ejemplos visuales para ayudar a comprender la estructura de la información.

Una base de datos está organizada en tablas, donde cada fila representa un registro (por ejemplo, un ejercicio o un usuario), y cada columna representa un campo de ese registro (por ejemplo, el nombre de un usuario o la duración de un ejercicio). A continuación, se detallan algunas de las tablas más relevantes para este proyecto.

La tabla 6.1 muestra los registros de las sesiones de ejercicios realizadas por los usuarios. Cada fila representa una sesión, y los campos incluyen información sobre el usuario, el ejercicio realizado, la fecha de inicio y finalización, y la duración de la sesión.

La tabla 6.2 contiene información sobre los registros de visitas de los usuarios, incluyendo detalles como el usuario, la fecha de inicio y finalización de la visita, y el navegador y país desde donde se accedió.

La tabla 6.3 almacena información sobre los diferentes ejercicios disponibles en la

| id [PK] integer | username text | start_date timestamp without time zone | end_date timestamp without time zone | duration double precision | exercise text |
|---|---|---|---|---|---|
| 298 | amgurjc | 2021-12-01 17:24:08.512187 | 2021-12-01 17:25:52.618119 | 104.105932 | obstacle_avoidance |
| 299 | mariamh | 2021-12-01 17:23:49.164972 | 2021-12-01 17:25:59.4145 | 130.249528 | rescue_people |
| 300 | mariamh | 2021-12-01 17:26:04.868734 | 2021-12-01 17:28:15.07145 | 130.202716 | rescue_people |
| 301 | fgomezl | 2021-12-01 17:28:33.600277 | 2021-12-01 17:30:53.251092 | 139.650815 | obstacle_avoidance |
| 302 | mariamh | 2021-12-01 17:28:23.518706 | 2021-12-01 17:31:45.124709 | 201.606003 | rescue_people |
| 303 | amgurjc | 2021-12-01 17:28:07.886411 | 2021-12-01 17:32:38.985946 | 271.099535 | obstacle_avoidance |
| 304 | amgurjc | 2021-12-01 17:32:39.026606 | 2021-12-01 17:39:19.10798 | 400.081374 | obstacle_avoidance |
| 305 | amgurjc | 2021-12-01 17:39:19.17405 | 2021-12-01 17:41:18.162209 | 118.988159 | obstacle_avoidance |
| 306 | carlosip | 2021-12-01 17:22:12.76852 | 2021-12-01 17:42:07.383919 | 1194.615399 | obstacle_avoidance |
| 307 | chuismiguel | 2021-12-01 16:44:58.295889 | 2021-12-01 17:46:26.431287 | 3688.135398 | obstacle_avoidance |
| 308 | mariamh | 2021-12-01 17:32:19.370519 | 2021-12-01 17:46:46.955035 | 867.584516 | rescue_people |

Figura 6.1: Tabla log_exercises.

Figura 6.2: Tabla log_session.

Figura 6.3: Tabla exercises.

plataforma, como el nombre del ejercicio, su descripción, los tags asociados y su estado. Cada fila de esta tabla corresponde a un ejercicio específico.

La tabla 6.4 contiene información sobre los usuarios registrados en la plataforma, incluyendo su nombre de usuario, nombre real, género, y la última fecha de inicio de sesión. También se almacena si el usuario tiene privilegios de superusuario.

A continuación, ya teniendo en contexto como se almacena la información en las tablas de la base de datos, se presentan ejemplos de consultas SQL que permiten extraer información relevante de las tablas mencionadas anteriormente. Estas consultas pueden ser utilizadas para analizar el comportamiento de los usuarios, las sesiones de ejercicio y el desempeño en los diferentes ejercicios.

y aqui cuando sepa poner codigo pongo querys y explico.

| id [PK] integer | last_login timestamp with time zone | is_superuser boolean | username character varying (150) | first_name character varying (150) | gender character varying (10) |
|---|---|---|---|---|---|
| 275 | 2024-04-02 09:56:01.352868+00 | false | i.linares | Ismael | M |
| 276 | 2023-04-12 15:09:51.994983+00 | false | jdpul_ | Jorge | M |
| 277 | 2022-04-19 10:26:08.507501+00 | false | enrique | Enrique | M |
| 278 | 2022-04-26 08:45:27.213432+00 | false | v.gilabert.2021 | VICENTE | M |
| 280 | 2022-04-19 15:29:00.179681+00 | false | hurta2 | Sergio | M |
| 281 | 2022-04-19 15:28:43.646237+00 | false | nacho9gs | Ignacio | M |
| 282 | 2022-04-19 23:38:44.874175+00 | false | alonsocn | Alonso | M |
| 283 | 2022-04-19 22:17:17.513394+00 | false | kcoutinho | Katherine | F |
| 284 | 2022-03-20 18:04:25.378575+00 | false | cv.lungu | Costin Valentin | M |
| 285 | [null] | false | PaulaST | Paula | F |
| 286 | 2022-12-12 01:08:10.920153+00 | false | PaulaS | Paula | F |

Figura 6.4: Tabla common_user.