Práctica 3 – Diseño, desarrollo e implementación de un sistema informático Web: la capa de modelo

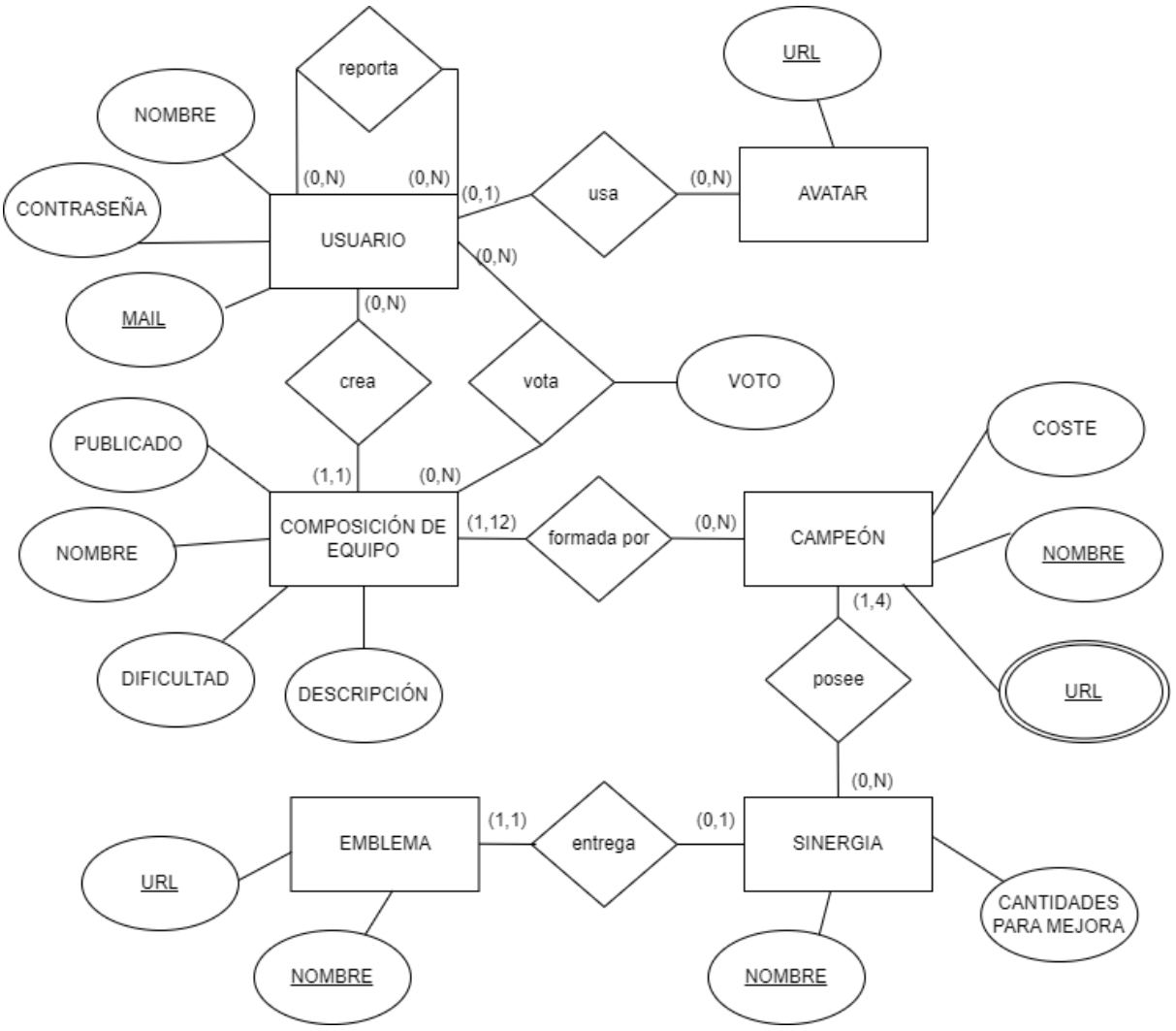
Peña Costa, Mateo (874162)
Cámara Domene, Raquel (875001)
Ibañez Sánchez, Marcos (869021)

martes, 8 de octubre de 2024

Resumen

En esta práctica desarrollaremos e implementaremos la base de datos de nuestra web. Para ello instalaremos y configuraremos nuestro entorno de trabajo para la base de datos, en nuestro caso esto será SQLite y Python con bibliotecas como Flask y Werkzeug. Además diseñamos el modelo Entidad-Relación para después transformarlo a un modelo Relacional. Por último crearemos los CREATE TABLES e INDEX necesarios para nuestra base y su capa de acceso en Python.

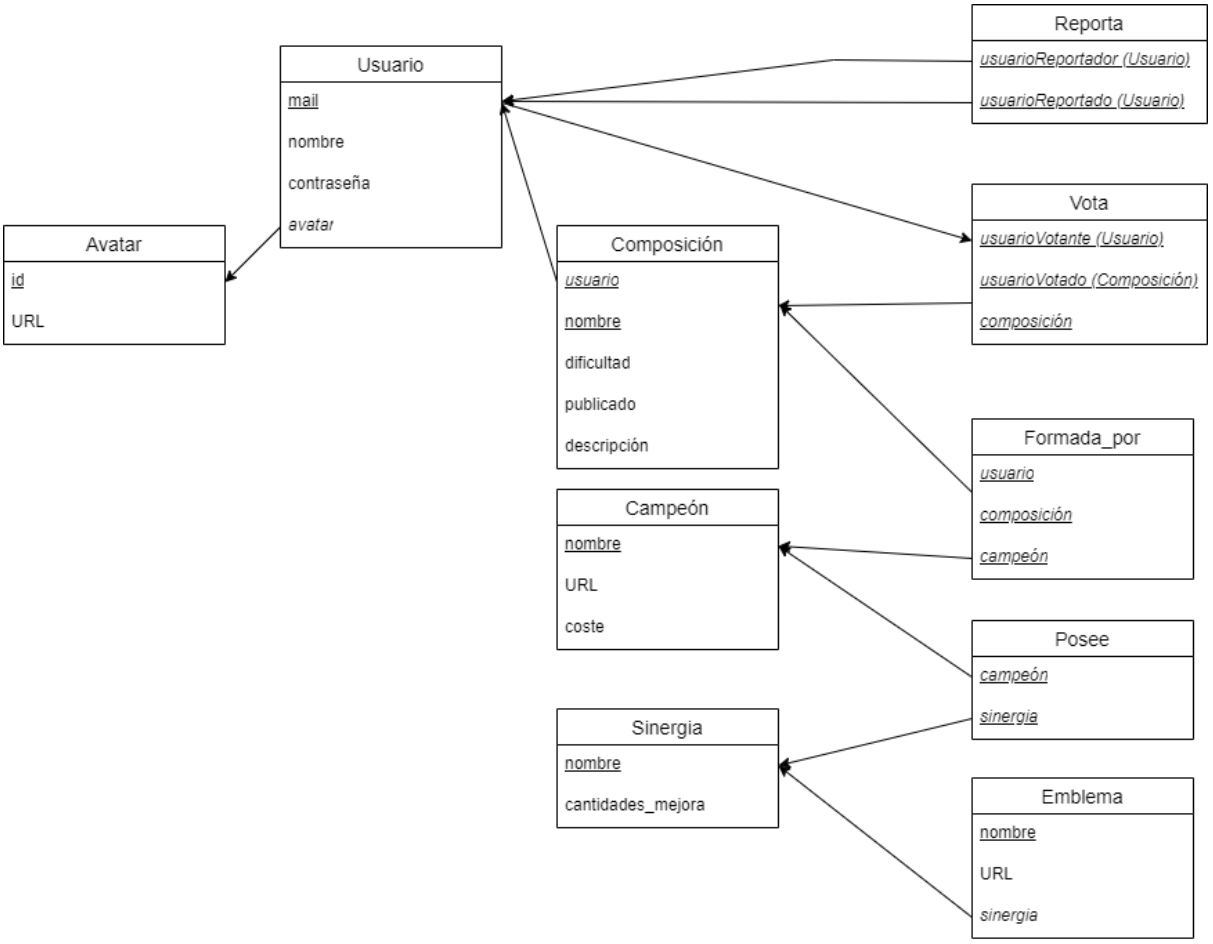
Modelo entidad-relación



Restricciones

- No pueden existir dos composiciones de equipo que tengan el mismo nombre y que hayan sido creadas por el mismo usuario.
- Un usuario no se puede reportar a sí mismo.

Modelo relacional



Para comprobar que nuestro modelo está normalizado hemos comprobado que cumplía la 2FN ya que no hay dependencias con parte de la clave, teniendo en cuenta que la única clave compuesta que no es todos los atributos es la clave compuesta de 'Composición', y está normalizada.
En cuanto a la 3FN, los atributos de composición no tienen ninguna relación entre ellos, los de la tabla 'Usuario' tampoco, y en 'Campeón' o 'Emblema', el atributo URL solo fue clave candidata, por lo que no hay dependencias.

Creación de tablas

Para la creación de tablas y su propia población hemos utilizado el lenguaje de sqlite.

Para ello hemos creado un archivo app.py que ejecuta la funcion 'init_db', lee del archivo 'schema.sql' que contiene el esquema de las tablas a crear con sus determinados 'CREATE TABLE' y va creando la base de datos 'database.db'.

Estamos trabajando con la herramienta de 'Docker Desktop' por lo que cuando ejecutamos el programa 'app.py' ya se crearon las tablas en el contenedor, y por lo que no hará falta que se vuelva a ejecutar el archivo cada vez que se quiera ejecutar el contenedor.

Diseño de las clases

Para la implementación de la capa de acceso a datos de la aplicación Web hemos decidido utilizar las clases DAO y VO. Hemos optado por trabajar con python para crear la base de datos, y sus clases. Para organizar mejor el código, por cada tabla creada hemos creado un archivo que incluye su clase VO, y la clase DAO que realiza la comunicación entre esta y la base de datos mediante consultas.
El nombre de los archivos de cada clase tiene el formato de 'nombre_de_la_tabla.py'

Metodología de trabajo

Durante el desarrollo de la práctica realizamos el modelo entidad-relación, primero haciendo un esquema a papel de cuáles serían las entidades y cuáles serían sus atributos correspondientes. Para la realización del esquema final de la memoria utilizamos la herramienta de Drawio que nos permitió trabajar conjuntamente.

Luego realizamos el modelo relacional teniendo en cuenta el esquema entidad-relación diseñado previamente, y los normalizamos para eliminar redundancias, mejorar la integridad de los datos, y optimizar las consultas.

Por último nos pusimos a hacer el programa 'app.py' que inicializará la base de datos, utilizando python, Flask y sqlite3. Para el diseño de clases creamos los archivos correspondientes a cada clase con su VO y DAO para interactuar con la base de datos.

Dificultades de trabajo

En el transcurso de la práctica nos surgieron una serie de dificultades.
El principal problema fue el hecho de no saber elegir muy bien qué lenguaje iríamos a utilizar para crear la base de datos, pero después de investigar nos decantamos con sqlite3 y python (con flask) ya que vimos que era una opción sencilla y manejable.

Otro problema que nos surgió fue el uso de Docker ya que como se comentó al principio de la asignatura iba a ser una herramienta muy útil para no tener que descargar todos los lenguajes o herramientas necesarias para la ejecución de nuestra página web, y que esto se llevara a cabo dentro de un contenedor. Pero se nos complicó más de lo esperado al no saber muy bien cómo utilizarla.

La última dificultad que nos surgió fue el hecho de tener que repasar la asignatura de bases de datos para hacer el esquema entidad-relación y el modelo relacional, pero tampoco nos llegó a suponer un problema.

Planificación y distribución de tareas

En cuanto a la organización que hemos llevado en esta práctica ha sido de manera equitativa. En esta práctica el líder ha sido Marcos. Nos hemos repartido el trabajo de manera que uno ha hecho el esquema entidad-relación, otro el modelo relacional, y otro los códigos en python para la creación de la base de datos.
Además, investigamos y recabamos información entre todos para entender cómo funcionaba Docker y como se podía sacar su máximo potencial para estas prácticas.

Horas trabajadas

	RAQUEL	MATEO	MARCOS
Resumen	-	15min	-
Modelo entidad-relación	15min	15min	2h
Modelo relacional	1h	15min	15min
Creación de tablas	15min	2h	15min
Diseño de clases	2h	2h	2h
Memoria	2h	1h	1h
TOTAL	5h30min	5h45min	5h30min

Bibliografía

- Mobalytics: https://mobalytics.gg/
- TFTactics: https://tftactics.gg/
- SQLite: https://www.sqlite.org/
- Flask: https://flask.palletsprojects.com/en/3.0.x/
- Werkzeug: https://werkzeug.palletsprojects.com/en/3.0.x/
- Flask sessions: https://testdriven.io/blog/flask-sessions/
- sqlite3: https://docs.python.org/3/library/sqlite3.html
- Documentación Docker: https://docs.docker.com/