

FitDiary

UN CUERPO MÁS FUERTE,
UN DIARIO MÁS INTELIGENTE



Realizado por
Àngel Pellicer Grau



Índice

01

INTRODUCCIÓN

02

HERRAMIENTAS Y MÉTODOS

03

PERSPECTIVA ESTÁTICA

04

PERSPECTIVA DINÁMICA

05

CONCLUSIONES

06

BIBLIOGRAFÍA Y
WEBGRAFÍA

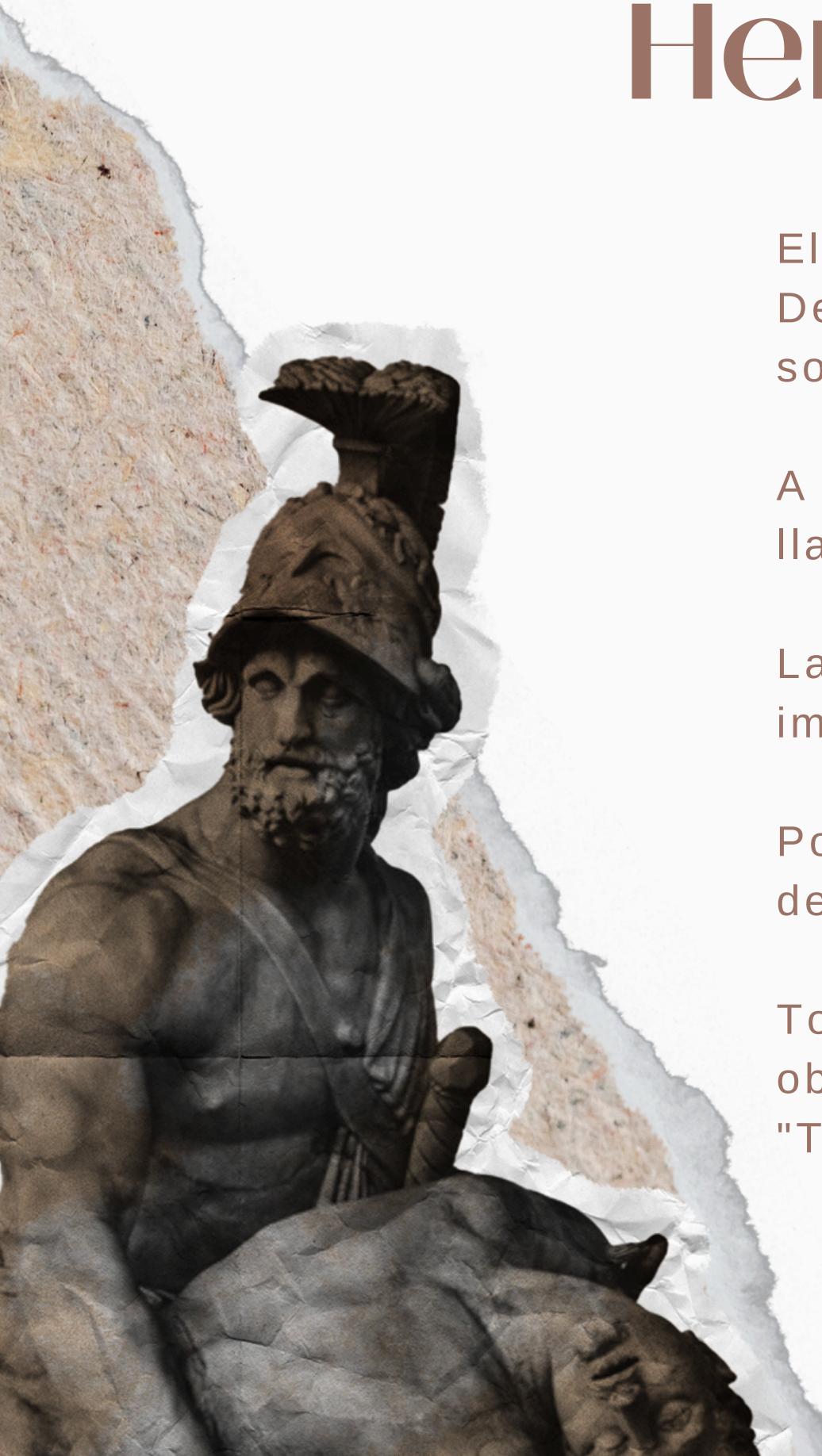
Introducción

FitDiary es una aplicación con la que podrás registrar tus entrenamientos, la duración de estos y tu progreso.

Podrás ver el progreso de tus días y el tiempo invertido en este.



Herramientas y Métodos



El proyecto comienza con la creacion de un esquema de entidad relacion. Después, este esquema se convierte en tablas que pueden ser leídas por el software.

A continuación, estas tablas se traducen a un lenguaje de programación llamado SQL, con el que gestionaremos la bases de datos.

La base de datos se construye utilizando un programa llamado SQLite que implementaremos con nuestro codigo de python.

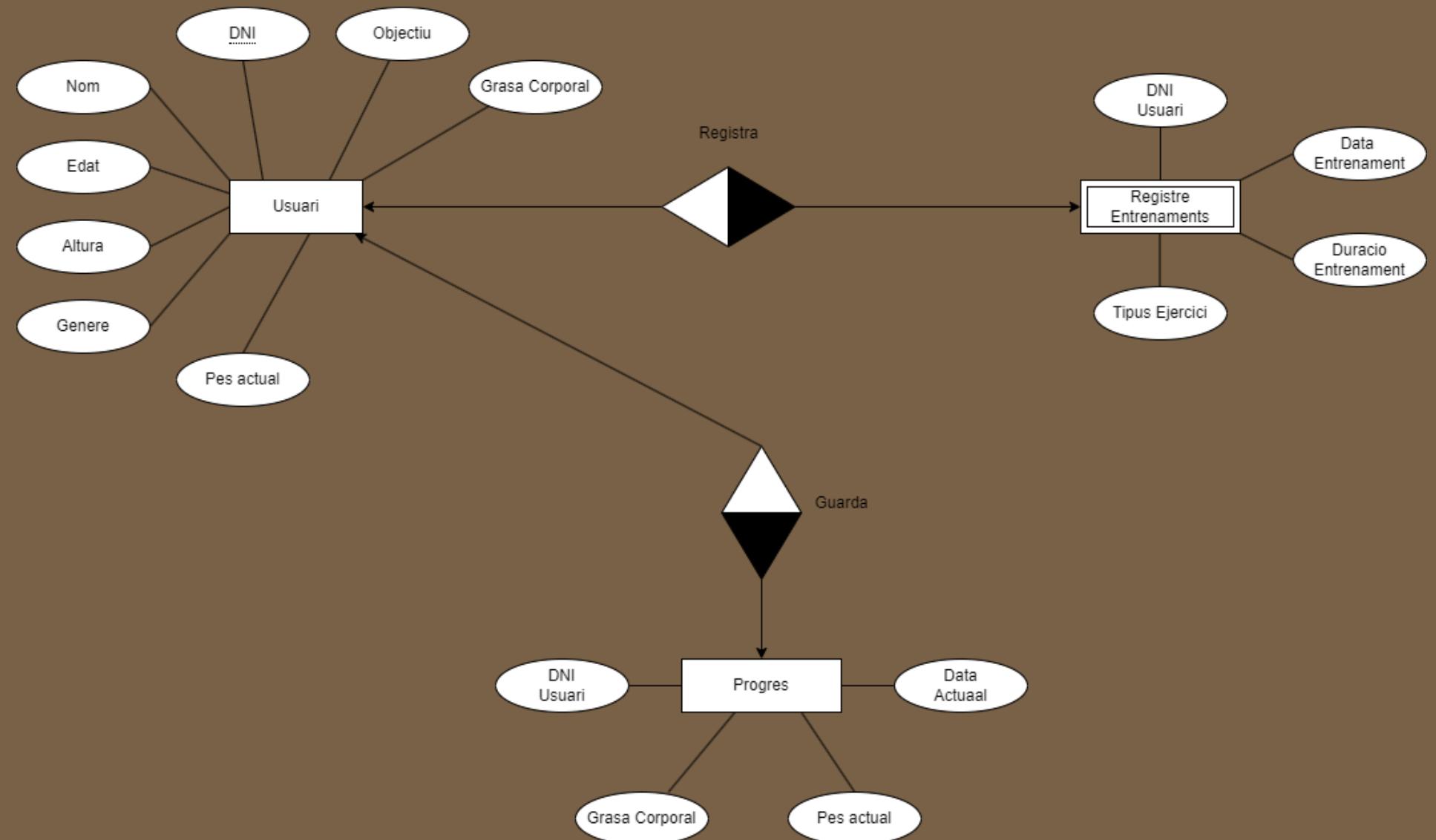
Posteriormente, se desarrolla una interfaz gráfica para interactuar con la base de datos y el programa en si.

Todo el desarrollo de este proyecto se ha realizado en un Windows 10, no obstante palabras clave como "SQL", "SQLite", "Python", "Visual Studio Code" y "TKinter" son elementos clave de este proyecto.

Perspectiva Estática

-
- E/R (Entidad-Relación)
 - Pasos a Tablas
 - DDL (Data Definition Language)
 - DML (Data Manipulation Language)
 - DQL (Data Query Language)

• E/R (Entidad-Relación)



• Paso a tablas

Usuario: DNI, Contraseña, grasa corporal, Nombre user, edat, altura, genero, peso actual.

Progres:DNI USER, data entrenamiento, Duracion entrenamiento, grasa corporal, tipo entrenamiento
C.Ali: DNI USER -> Usuario(Dni)
VNN: DNI USER

Registro Entrenamientos: DNI USER, Peso actual , grasa corporal, tipo entrenamiento, data actual.

C.Ali: DNI USER -> Usuario(Dni)

VNN: DNI USER

DDL (Data Definition Language)

```
def crear_tablas(self):
    cursor = self.conexion_bd.cursor()
    cursor.execute(''CREATE TABLE IF NOT EXISTS usuarios (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        usuario TEXT UNIQUE,
        contraseña TEXT,
        edad INTEGER,
        dni TEXT UNIQUE,
        altura REAL,
        genero TEXT,
        peso REAL,
        grasa_corporal REAL
    )'''')
```

Esta parte del código define una tabla llamada "usuarios" con columnas para el ID, nombre de usuario, contraseña, edad, DNI, altura, género, peso y grasa corporal.

```
cursor.execute(''CREATE TABLE IF NOT EXISTS registros_entrenamiento (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    usuario_id INTEGER,
    fecha TEXT,
    duracion INTEGER,
    tipo_entrenamiento TEXT,
    FOREIGN KEY(usuario_id) REFERENCES usuarios(id)
)''')
self.conexion_bd.commit()
```

Este fragmento crea una tabla llamada "registros_entrenamiento" con columnas para el ID. El ID del usuario al que pertenece este registro, la fecha del entrenamiento, la duración del entrenamiento en minutos y el tipo de entrenamiento.

Además, establece una clave externa que referencia el ID del usuario en la tabla de usuarios.

DML (Data Manipulation Language)

1. Registro de un nuevo usuario:

```
def registrar_usuario(self):
    usuario = self.usuario_entry.get()
    contraseña = self.password_entry.get()
    edad = self.edad_entry.get()
    dni = self.dni_entry.get()
    altura = self.altura_entry.get()
    genero = self.genero_combobox.get()
    peso = self.peso_entry.get()
    grasa_corporal = self.grasa_corporal_entry.get()

    cursor = self.conexion_bd.cursor()
    cursor.execute("INSERT INTO usuarios (usuario, contraseña, edad, dni, altura,
        (usuario, contraseña, edad, dni, altura, genero, peso, grasa_corporal)
    self.conexion_bd.commit()
    messagebox.showinfo("Éxito", "Usuario registrado correctamente.")
    self.mostrar_inicio_sesion()
```

2. Guardar modificaciones en los datos del usuario:

```
def guardar_modificaciones(self):
    nombre = self.nombre_entry.get()
    edad = self.edad_entry.get()
    altura = self.altura_entry.get()
    contraseña = self.contraseña_entry.get()
    peso = self.peso_entry.get()
    grasa_corporal = self.grasa_corporal_entry.get()

    cursor = self.conexion_bd.cursor()
    cursor.execute("UPDATE usuarios SET usuario=?, edad=?, altura=?,
        (nombre, edad, altura, contraseña, peso, grasa_corporal, self
    self.conexion_bd.commit()
    messagebox.showinfo("Éxito", "Datos modificados correctamente.")
```

DQL (Data Query Language)

1. Inicio de sesión para verificar las credenciales
2. Obtener datos del usuario actual: del usuario:

```
def iniciar_sesion(self):
    usuario = self.usuario_entry.get()
    password = self.password_entry.get()

    cursor = self.conexion_bd.cursor()
    cursor.execute("SELECT * FROM usuarios WHERE usuario = ? AND contraseña = ?"
    usuario_encontrado = cursor.fetchone()

    if usuario_encontrado:
        self.usuario_actual = usuario
        self.mostrar_menu()
    else:
        messagebox.showerror("Error", "Usuario o contraseña incorrectos.")
```

```
def obtener_datos_usuario_actual(self):
    cursor = self.conexion_bd.cursor()
    cursor.execute("SELECT * FROM usuarios WHERE usuario = ?", (self.usuario_actual,))
    usuario_data = cursor.fetchone()
    return usuario_data[1], usuario_data[4], usuario_data[3], usuario_data[5], usuario_data[6]
```

DCL (Data Control Language)

```
def crear_tablas(self):
    cursor = self.conexion_bd.cursor()
    cursor.execute(''CREATE TABLE IF NOT EXISTS usuarios (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        usuario TEXT UNIQUE,
        contraseña TEXT,
        edad INTEGER,
        dni TEXT UNIQUE,
        altura REAL,
        genero TEXT,
        peso REAL,
        grasa_corporal REAL
    )'''')
```

Esta parte del código define una tabla llamada "usuarios" con columnas para el ID, nombre de usuario, contraseña, edad, DNI, altura, género, peso y grasa corporal.

```
cursor.execute(''CREATE TABLE IF NOT EXISTS registros_entrenamiento (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    usuario_id INTEGER,
    fecha TEXT,
    duracion INTEGER,
    tipo_entrenamiento TEXT,
    FOREIGN KEY(usuario_id) REFERENCES usuarios(id)
)''')
self.conexion_bd.commit()
```

Este fragmento crea una tabla llamada "registros_entrenamiento" con columnas para el ID. El ID del usuario al que pertenece este registro, la fecha del entrenamiento, la duración del entrenamiento en minutos y el tipo de entrenamiento.

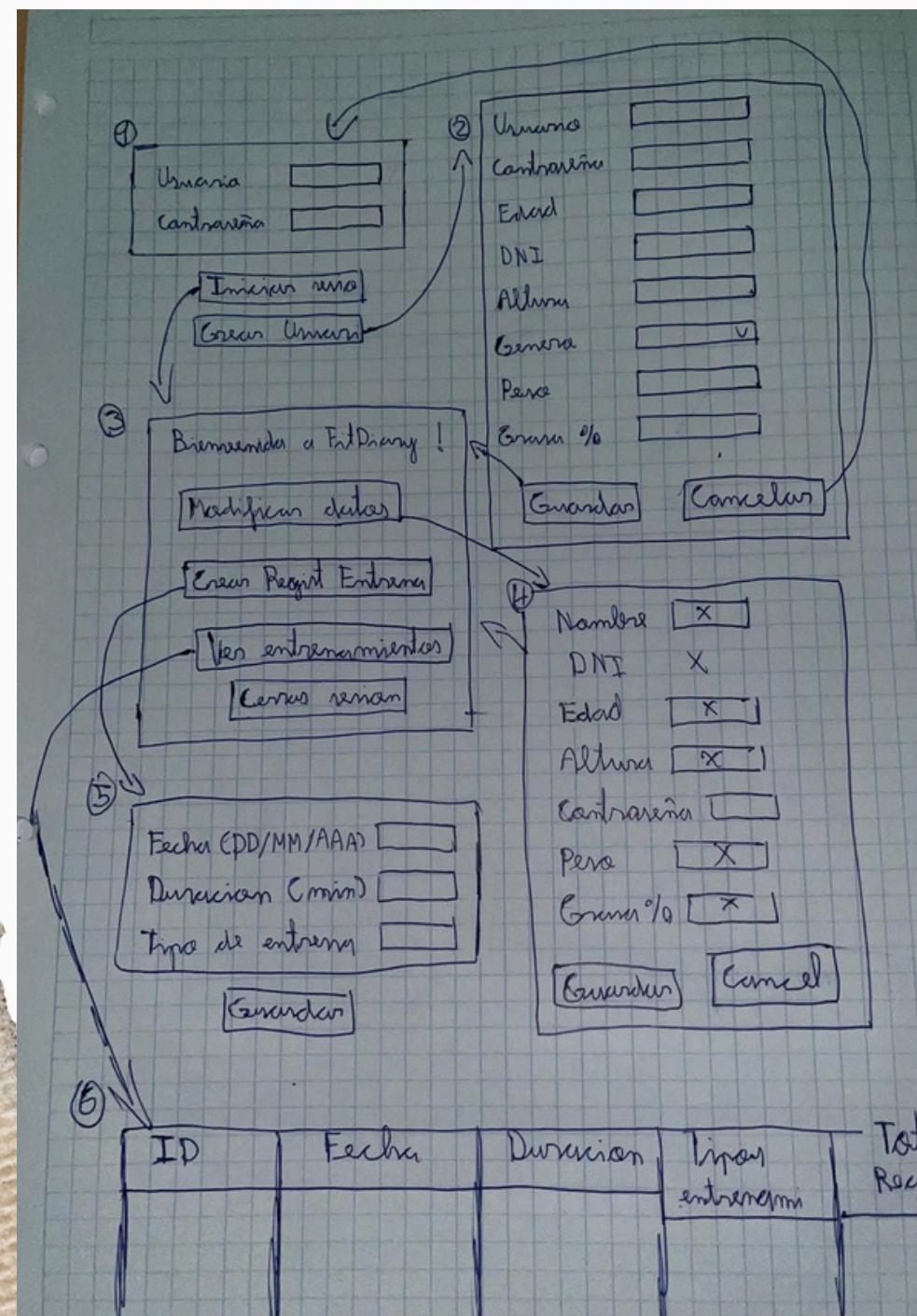
Además, establece una clave externa que referencia el ID del usuario en la tabla de usuarios.

Perspectiva Dinámica



-
- Sketch
 - Casos de Uso (Métodos)
-

Sketch y Casos de Uso (Métodos)



MODIFICAR DATOS

| Caso de Uso | MODIFICAR DATOS | Identificador: MODIFICAR DATOS |
|---------------|--|-----------------------------------|
| Actores | Usuario | |
| Tipo | Primario | |
| Precondición | Que el usuario este creado | |
| Postcondición | Modificación de los datos | |
| Descripción | En caso de que algún dato cambie con el tiempo el usuario lo podrá modificar aquí. | |

CREAR REGISTRO ENTRENAMIENTO

| Caso de Uso | CREAR REGISTRO ENTRENAMIENTO | Identificador: CREAR REGISTRO ENTRENAMIENTO |
|---------------|--|--|
| Actores | Usuario | |
| Tipo | Primario | |
| Precondición | Que el usuario este creado | |
| Postcondición | Crea un registro de entrenamiento | |
| Descripción | Crea un registro del entrenamiento realizado | |

VER ENTRENAMIENTOS

| Caso de Uso | VER ENTRENAMIENTOS | Identificador: VER ENTRENAMIENTOS |
|---------------|--|--------------------------------------|
| Actores | Usuario | |
| Tipo | Primario | |
| Precondición | Que el exista algún entrenamiento | |
| Postcondición | Visualizar los entrenamientos | |
| Descripción | Ver el registro de entrenamientos que tienes creados y supuestamente realizados. | |

CERRAR SESION

| Caso de Uso | CERRAR SESION | Identificador: CERRAR SESION |
|---------------|---|---------------------------------|
| Actores | Usuario | |
| Tipo | Primario | |
| Precondición | Que la sesión este iniciada | |
| Postcondición | Se cerrara la sesion | |
| Descripción | Al terminar de usar la aplicación el usuario cerrara la sesión con esta opción. | |

Conclusiones

- Resumen de los resultados obtenidos.
- Reflexiones sobre el proceso y posibles mejoras futuras.

El programa FitDiary es un programa creado para la gente que le gusta medir su progreso y ver la evolución de su cuerpo. De esta forma el mismo usuario se motiva para seguir utilizando la aplicación.

El programa FitDiary se podría mejorar tanto en interfaz gráfica como en algunos aspectos para facilitar a los usuarios la forma de usarlo. También podría implementar una sección de dietas con los usuarios Prime, etc.

Bibliografía y Webgrafía

- Apunts del Tema 4 sobre base de dades en SQL.
- Youtube.
- ChatGpt.



MUCHAS GRACIAS

REALIZADO POR ÀNGEL
PELLICER GRAU

Contacto:

Gmail: pellicergrauangel@gmail.com

Tel: +34 684 454 683

GitHub:

<https://github.com/PellicerAngel/FitDiary-.git>