**MEMORIA FINAL DE PROYECTO**

**TÍTULO DEL PROYECTO**

**[EMPRESA, OPCIONAL]**

**[LOGOTIPO, OPCIONAL]**

**CICLO FORMATIVO DE GRADO SUPERIOR**

**DESARROLLO DE APLICACIONES WEB**

**AUTOR**

JORDI RUIZ ROMÁN

**TUTOR**

JOSÉ JUAN LÓPEZ VÉLEZ

**COORDINADOR**

LORENA ISABEL PALACIO LATORRE

ÍNDICE

[1. INTRODUCCIÓN 5](#_Toc183456229)

[2. INTRODUCCIÓN EN INGLÉS 5](#_Toc183456230)

[3. OBJETIVOS 5](#_Toc183456231)

[3.1. Objetivos fase actual 5](#_Toc183456232)

[3.2. Objetivos fases futuras 5](#_Toc183456233)

[4. PLANIFICACIÓN 5](#_Toc183456234)

[4.1. Tabla de hitos 5](#_Toc183456235)

[4.2. Diagrama de Gantt 6](#_Toc183456236)

[5. ANÁLISIS 6](#_Toc183456237)

[5.1. Estado del arte 6](#_Toc183456238)

[5.2. Funcionalidades 6](#_Toc183456239)

[6. DISEÑO 6](#_Toc183456240)

[6.1. Requisitos técnicos 6](#_Toc183456241)

[6.2. Arquitectura web 7](#_Toc183456242)

[6.3. Diseño back-end 7](#_Toc183456243)

[6.3.1. Modelo de datos 7](#_Toc183456244)

[6.3.2. Servicios REST 7](#_Toc183456245)

[6.3.3. Paquetes adicionales 7](#_Toc183456246)

[6.4. Diseño front-end 7](#_Toc183456247)

[6.4.1. Mock-ups 7](#_Toc183456248)

[6.4.2. Guía de estilos 7](#_Toc183456249)

[6.4.3. Paquetes adicionales 7](#_Toc183456250)

[7. IMPLEMENTACIÓN 8](#_Toc183456251)

[7.1. Servidor 8](#_Toc183456252)

[7.2. Cliente 8](#_Toc183456253)

[8. DESPLIEGUE 8](#_Toc183456254)

[8.1. Modelo de despliegue utilizado 8](#_Toc183456255)

[8.2. Datos iniciales y configuración 8](#_Toc183456256)

[8.3. Pasos para el despliegue 8](#_Toc183456257)

[8.4. Proveedores y servicios utilizados 8](#_Toc183456258)

[9. HERRAMIENTAS UTILIZADAS 8](#_Toc183456259)

# INTRODUCCIÓN

Este documento presenta el trabajo realizado para el módulo de Proyecto del Ciclo Formativo de Grado Superior en Desarrollo de Aplicaciones Web. El objetivo principal es aplicar de forma práctica los conocimientos adquiridos durante el ciclo, desarrollando una aplicación web completa desde su planificación hasta su implementación.

El proyecto consiste en una página web para un gimnasio, que ofrece información sobre sus servicios, rutinas de entrenamiento personalizadas, tienda online de suplementos deportivos y herramientas para el seguimiento del progreso físico de los usuarios. Se ha utilizado una combinación de tecnologías modernas para ofrecer una solución útil, atractiva y funcional.

# INTRODUCCIÓN EN INGLÉS

This document presents the work carried out for the Project module of the Advanced Vocational Training Course in Web Application Development. The main goal is to apply the knowledge acquired throughout the course by developing a complete web application, from planning to implementation.

The project consists of a website for a gym, which offers information about its services, personalized workout routines, an online store for sports supplements, and tools for tracking users’ physical progress. A combination of modern technologies has been used to create a useful, attractive, and functional solution.

# OBJETIVOS

El propósito de este proyecto es desarrollar una aplicación web funcional y estructurada para un gimnasio, que sirva como plataforma centralizada para informar, asistir y fidelizar a los usuarios. El objetivo general es implementar una solución que combine servicios informativos, rutinas personalizadas y funcionalidades básicas de comercio electrónico, aplicando las tecnologías y conocimientos adquiridos a lo largo del ciclo formativo..

## Objetivos fase actual

Durante la fase correspondiente al módulo de Proyecto, se plantea alcanzar una solución básica pero completamente funcional, con los siguientes objetivos:

* Crear una estructura web clara, usable y responsive.
* Mostrar información detallada sobre centros, horarios y tarifas.
* Permitir el registro y autenticación de usuarios.
* Habilitar una zona privada con acceso a contenidos personalizados.
* Implementar un sistema de visualización de ejercicios mediante un cuerpo humano interactivo.
* Diseñar e integrar un sistema de progreso con seguimiento en gráficos y estadísticas.
* Conectar la aplicación con una base de datos para gestionar usuarios, ejercicios y centros.

## Objetivos fases futuras

Los siguientes objetivos se consideran ampliaciones que podrían incorporarse en futuras fases del proyecto, una vez consolidada la base funcional:

* Visualización interactiva del cuerpo humano con ejercicios por grupo muscular.
* Sistema de seguimiento del progreso del usuario (gráficas, medidas, pesos, etc.).
* Integración de un sistema de gamificación (puntos, logros, recompensas).
* Valoraciones y comentarios en productos y rutinas.
* Mejora del sistema de búsqueda con filtros avanzados.
* Implementación de métodos de pago reales y notificaciones de pedidos.
* Optimización de la seguridad y protección de datos personales.

# PLANIFICACIÓN

Para garantizar el éxito del desarrollo durante el módulo de Proyecto, se ha definido una planificación que divide el trabajo en fases concretas. Cada fase incluye tareas específicas con una duración estimada, lo que permite organizar el tiempo de manera eficiente y detectar posibles desviaciones.

## Tabla de hitos

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Fase** |  |  | | --- | |  | | **Tareas asociadas** | **Duración estimada** | **Fecha inicio** | **Fecha fin** |
| |  | | --- | | **1. Análisis y diseño** |  |  | | --- | |  | | - Definición de requisitos - Diseño de estructura y base de datos | |  | | --- | | 7 días |  |  | | --- | |  | | |  | | --- | | 06/05/2025 |  |  | | --- | |  | | |  | | --- | | 12/05/2025 |  |  | | --- | |  | |
| 2. Maquetación web | - Diseño de la interfaz principal (HTML, CSS, JS)  - Zona pública básica | 7 días | 13/05/2025 | |  | | --- | | 19/05/2025 |  |  | | --- | |  | |
| |  | | --- | | **3. Backend básico** |  |  | | --- | |  | | - Configuración de Django - Modelado de datos - Registro/login | 10 días | |  | | --- | | 20/05/2025 |  |  | | --- | |  | | |  | | --- | | 29/05/2025 |  |  | | --- | |  | |
| |  | | --- | | **4. Zona privada** |  |  | | --- | |  | | - Visualización de rutinas - Acceso personalizado por usuario | |  | | --- | | 5 días |  |  | | --- | |  | | |  | | --- | | 30/05/2025 |  |  | | --- | |  | | |  | | --- | | 03/06/2025 |  |  | | --- | |  | |
|  |  |  |  |  |

## Diagrama de Gantt

Se pasará la tabla de hitos a un diagrama Gantt, para poder visualizar las fases y tareas a realizar a lo largo del tiempo de duración del proyecto.

# ANÁLISIS

## Estado del arte

Detalla aquí cualquier solución existente, parecida, que trate el mismo problema, o en la que te hayas basado para desarrollar tu idea. Es interesante detallar qué ofrecen las soluciones existentes, y en qué se diferencian de la solución desarrollada.

## Funcionalidades

A continuación, se enumeran las funcionalidades que formarán parte de esta primera fase del desarrollo del proyecto, garantizando una base sólida y operativa para la aplicación web del gimnasio:

1. Página informativa del gimnasio

Visualización de los diferentes centros disponibles.

Descripción de instalaciones y horarios.

Visualización de tarifas actualizadas.

2. Gestión de usuarios

Registro de nuevos usuarios con validación de datos.

Inicio de sesión seguro con autenticación.

Zona privada para el usuario con acceso personalizado a funcionalidades.

5. Cuerpo humano interactivo (versión inicial)

Visualización de figura anatómica con zonas musculares destacadas.

Selección de músculo y muestra de ejercicios recomendados asociados.

# Seguimiento del progreso (versión inicial)

Registro manual del progreso: peso levantado, repeticiones y fechas.

Visualización de registros mediante gráficos y estadísticas.

# DISEÑO

## Requisitos técnicos

1. Gestión de usuarios (registro/login)

Requisito técnico:

Uso del sistema de autenticación de Django (django.contrib.auth) para gestión de usuarios y sesiones.

Cifrado de contraseñas mediante el sistema de hashing seguro incluido en Django.

Middleware de autenticación y control de accesos a zonas privadas.

2. Página informativa del gimnasio

Requisito técnico:

Sistema de plantillas de Django (Django Templates).

Estructura modular con archivos HTML5, CSS3 y JavaScript para el frontend.

Almacenamiento de la información en modelos de base de datos gestionados por Django ORM.

5. Cuerpo humano interactivo

Requisito técnico:

Implementación inicial con JavaScript y archivo SVG seccionado por músculos.

Uso de videos de YouTube a modo de tutorial.

6. Seguimiento del progreso

Requisito técnico:

Modelos personalizados en Django para almacenar métricas del usuario (peso, repeticiones, fechas).

Uso de bibliotecas de gráficos como Chart.js para visualizar la evolución del usuario.

7. Arquitectura y despliegue

Requisito técnico:

Uso de contenedores Docker para el entorno de desarrollo y despliegue.

Separación de servicios (Django, Postgres) en un docker-compose.yml.

Configuración de entorno .env para gestión de credenciales y parámetros sensibles.

## Arquitectura web

La arquitectura web implementada en este proyecto se basa en el modelo MVC (Modelo-Vista-Controlador) proporcionado por el framework Django.

Además, se emplea tecnología basada en contenedores mediante Docker, lo que permite un entorno de desarrollo y despliegue más limpio, reproducible y aislado.

Elementos principales de la arquitectura

Frontend (Cliente)

HTML5, CSS3, JavaScript.

Django Templates para renderizado de vistas dinámicas.

Chart.js para la visualización de progreso del usuario.

Backend (Servidor)

Django (Python) actuando como servidor principal y controlador de lógica de negocio.

Sistema de rutas y vistas gestionadas con el patrón MVC.

Seguridad y control de acceso a través del sistema de autenticación integrado.

Base de datos

Postgres, gestionada por Django ORM para la persistencia de datos de usuarios, rutinas, productos, progreso, etc.

Servicios y tareas asincrónicas

Contenedores Docker

Contenedor para Django (app).

Contenedor para MySQL (base de datos).

Orquestación mediante docker-compose.

## Diseño back-end

El desarrollo del back-end se ha realizado utilizando el framework Django, basado en Python. Django ha sido elegido por su robustez, seguridad integrada y su enfoque en el patrón arquitectónico MVC (Modelo-Vista-Controlador), que permite mantener una estructura limpia y escalable.

Características destacadas del back-end:

Framework: Django 4.2.19

Lenguaje: Python 3.11

Base de datos: Postgres, gestionada mediante el ORM (Object-Relational Mapper) de Django, que facilita la interacción con la base de datos sin necesidad de escribir consultas SQL manuales.

Autenticación: Se ha utilizado el sistema de autenticación integrado de Django, permitiendo el registro, login y gestión de sesiones de usuarios.

Gestión de usuarios: Uso de UserModel extendido para almacenar información adicional del usuario (como progreso o rutinas).

Panel de administración: Se ha creado un panel de administración personalizado.

API REST (en fase futura): Se contempla la implementación de una API REST utilizando Django REST Framework, para posibilitar la conexión con futuras apps móviles o integraciones externas.

Seguridad: Protección contra ataques CSRF y XSS proporcionada por defecto. También se han implementado buenas prácticas como validación de formularios y control de permisos por rol.

Esta estructura permite una fácil ampliación, manteniendo la coherencia del código, reutilización de componentes y un mantenimiento más eficiente del sistema.

### Modelo de datos

La aplicación utiliza el ORM de Django sobre una base de datos relacional para estructurar y gestionar los datos. A continuación, se detallan los modelos principales definidos en el sistema, que permiten gestionar usuarios, entrenamientos y centros deportivos.

* **Modelos principales**
* **MyUser**
  + email (EmailField, único) — identificador principal del usuario.
  + username (CharField)
  + birthdate (DateField)
  + create\_date / update\_date (DateTimeField)
  + is\_staff, is\_superuser (BooleanField) — permisos administrativos.
* **Centro**
  + nombre (CharField)
  + direccion (TextField)
  + telefono (CharField)
  + horario (CharField)
  + imagen (ImageField) — imagen del centro.
* **Musculo**
  + nombre (CharField) — representa un músculo trabajado.
* **Ejercicio**
  + nombre (CharField)
  + musculo (ForeignKey a Musculo) — cada ejercicio se asocia a un grupo muscular.
* **Entrenamiento**
  + usuario (ForeignKey a MyUser) — quién realiza el entrenamiento.
  + ejercicio (ForeignKey a Ejercicio) — qué ejercicio se entrena.
  + fecha (DateField, auto) — fecha de realización.
  + Relación uno a muchos con Serie.
* **Serie**
  + entrenamiento (ForeignKey a Entrenamiento)
  + repeticiones (PositiveIntegerField)
  + peso\_levantado (FloatField)
* **Relaciones clave**
* Cada **usuario** puede registrar múltiples **entrenamientos**.
* Un **entrenamiento** contiene varias **series**.
* Cada **ejercicio** se asocia a un único **músculo**.
* Se puede registrar información de diferentes **centros deportivos** donde se pueden desarrollar las rutinas.

### Servicios REST

En este proyecto no se han implementado servicios REST. Toda la lógica de interacción entre el cliente y el servidor se gestiona mediante vistas tradicionales de Django y el renderizado de plantillas HTML en el servidor. Esto ha permitido mantener una estructura sencilla y coherente con los objetivos actuales del proyecto.

### Paquetes adicionales

Durante el desarrollo del proyecto se han utilizado varios paquetes adicionales de Python y Django, de los cuales destacan los siguientes por su impacto en la funcionalidad y experiencia del usuario:

* **django-allauth**: Facilita la autenticación de usuarios, incluyendo el registro, login y gestión de sesiones. Este paquete ha sido clave para implementar un sistema de usuarios robusto.
* **django-crispy-forms + crispy-bootstrap5**: Permiten una mejor presentación y personalización de los formularios utilizando Bootstrap 5, mejorando la experiencia visual y usabilidad del sitio sin necesidad de escribir mucho HTML adicional.
* **Pillow**: Biblioteca esencial para el tratamiento de imágenes, necesaria para la gestión de imágenes subidas por los usuarios (por ejemplo, imágenes de centros deportivos).
* **psycopg2-binary**: Driver utilizado para la conexión entre Django y la base de datos PostgreSQL, esencial para el funcionamiento de la aplicación.

## Diseño front-end

El diseño del frontend se ha realizado mediante plantillas HTML tradicionales de Django con el uso de CSS personalizado. La presentación se ha centrado en lograr una interfaz clara y funcional, sin el uso de frameworks modernos de JavaScript como React o Vue.

Se han usado formularios estándar de Django para la entrada de datos, y se ha procurado una organización lógica de las páginas para facilitar la navegación por parte del usuario.

### Mock-ups

Diagrama

El contenido generado por IA puede ser incorrecto.

### Guía de estilos

Gráfico, Gráfico en cascada

El contenido generado por IA puede ser incorrecto.

### Paquetes adicionales

6.4.4. Capturas de la aplicación

Capturas de las principales pantallas de la aplicación final en versión móvil, tablet y escritorio.

# IMPLEMENTACIÓN

## Servidor

Detalle de la implementación realizada en la parte servidor.

## Cliente

Detalle de la implementación realizada en la parte cliente.

# DESPLIEGUE

## Modelo de despliegue utilizado

Descripción del tipo de despliegue utilizado en el proyecto.

## Datos iniciales y configuración

Explicaicón de la carga de datos iniciales así como la configuración del sistema para su puesta en marcha.

## Pasos para el despliegue

Secuencia de pasos realizados para realizar el despliegue de la aplicación.

## Proveedores y servicios utilizados

Listado de proveedores y servicios usados en el despliegue.

# HERRAMIENTAS UTILIZADAS

Descripción de las herramientas usadas en el proyecto.