Coordinación de Ingeniería en Sistemas Computacionales

INSTITUTO TECNOLÓGICO SUPERIOR DEL SUR DE GUANAJUATO

Proyecto final

Materia:

Programación Web II.

Semestre:
7°A

Ing. Sistemas Computacionales

Elaborada por:

America Citlalli Lopez Lemus
Lizet Guadalupe Lopez Medina
Jesús Rosiles Gonzales
Paola Montserrat Ruiz Carmen

Profesor:
Gustavo Ivan Vega Olvera

Uriangato, Gto. a 9 de octubre del 2025

Materia:	Programación Web II		
Nombre Informe:	**Plataforma de Registro y Control para Gimnasios**		
Alumno(s):	America Citlalli Lopez Lemus Lizet Guadalupe Lopez Medina Jesús Rosiles Gonzales Paola Montserrat Ruiz Carmen	**Fecha:**	09/10/2025

Objetivo

El proyecto consiste en el desarrollo de una Plataforma Web de Registro y Control para Gimnasios utilizando el framework Django. El objetivo principal es automatizar la gestión de usuarios, suscripciones, asistencias y reportes, buscando solucionar la problemática de la gestión manual o con herramientas básicas que conduce a errores y pérdida de tiempo en la administración del negocio.

El sistema se basará en el patrón de desarrollo MVT (Modelo-Vista-Template) y utilizará herramientas robustas de Django como el ORM, el sistema de autenticación y soporte de seguridad.

Temas del plan de estudios

Base de Datos y Modelos

- Modelos
- Migraciones
- ORM (Object Relational Mapper)
- Relaciones de modelos (uno a uno, uno a muchos, muchos a muchos)

Unidad 3: Formularios y Administración

- 3.1 Formularios
- 3.5 Módulo de Administración de Django
- 3.6 Buscadores y Filtros

Unidad 4: Autenticación y Autorización

- 4.2 Sistema de autenticación de Django
- 4.4 Seguridad Web

Unidad 5: Desarrollo Avanzado y Despliegue

- 5.1 Django REST Framework

- 5.2 Peticiones Asíncronas (AJAX)
- 5.3 Reportes y Gráficos
- 5.4 Exportación de datos (PDF y Excel)
- 5.5 Despliegue a Producción

Material

El desarrollo de la Plataforma de Registro y Control para Gimnasios se basa en dos conjuntos de materiales principales: el Marco Teórico/Conceptual y el Stack de Herramientas y Tecnologías.

1. Marco Teórico y Conceptual

El proyecto surge de la necesidad de reemplazar los métodos manuales (como cuadernos y Excel) por una plataforma automatizada para gestionar ingresos, gastos, usuarios y asistencias en gimnasios y centros deportivos.

El enfoque metodológico y la base tecnológica son los siguientes:

- **Framework Principal:** Se utilizará Django, un framework de desarrollo web de alto nivel escrito en Python, elegido por promover el desarrollo rápido y un diseño limpio y pragmático.
- **Componentes Incluidos en Django:** Django ya incluye herramientas necesarias como un ORM robusto, un sistema de autenticación y soporte para seguridad, lo que lo hace idóneo para la aplicación requerida.
- **Patrón de Desarrollo:** La arquitectura se basará en el patrón MVT (Modelo-Vista-Template), donde las aplicaciones se construyen como componentes separados y reutilizables.

2. Herramientas y Tecnologías (Stack Técnico)

El proyecto requiere el uso de herramientas específicas para cumplir con los requerimientos técnicos y de seguridad definido.

A. Lenguajes y Frameworks Esenciales

- **Django:** Como framework central, implementando las cinco unidades temáticas principales del plan de estudios.
- **Django REST Framework:** Necesario para implementar el Web API y exponer datos de asistencia y usuarios.
- **AJAX:** Se usará para las Peticiones Asíncronas en la funcionalidad de registro de asistencia, permitiendo una validación en tiempo real sin recargar la página.

B. Base de Datos y Persistencia

- **Sistema de Gestión de Base de Datos:** Se requiere el uso de SQLite para la base de datos del proyecto.
- **ORM y Modelos:** Se utilizará el ORM de Django para manejar los Modelos y las Relaciones de modelos (uno a uno, uno a muchos, muchos a muchos).

Marco Teórico

El presente proyecto se fundamenta en la necesidad crucial de modernizar la gestión operativa de gimnasios y centros deportivos. El problema central que buscamos solucionar es la ineficiencia que genera el uso de métodos manuales, como cuadernos, o herramientas básicas como Excel, lo cual inevitablemente conduce a errores, demoras, y una pérdida de tiempo considerable en las tareas administrativas. Por esta razón, el desarrollo de una plataforma automatizada es la solución indispensable para garantizar la gestión precisa, ágil, y centralizada de ingresos, gastos, usuarios, y asistencias.

La plataforma será construida sobre Django, un *framework* de desarrollo web de alto nivel escrito en Python. Django es la elección tecnológica ideal porque promueve el desarrollo rápido y se adhiere a un diseño limpio y pragmático. Se trata de un *framework* que viene con "todo incluido", proporcionando un conjunto robusto de herramientas esenciales para el proyecto.

La arquitectura de la aplicación se adherirá al patrón Modelo-Vista-Template (MVT). Este patrón organiza el código en componentes separados y reutilizables, lo que garantiza un desarrollo modular y fácil de mantener.

El soporte nativo de Django es un pilar del proyecto:

- Un ORM (Object Relational Mapper) robusto facilita la manipulación de la base de datos (SQLite) utilizando código Python en lugar de SQL, lo que acelera el desarrollo y garantiza la coherencia.
- Un sistema de autenticación y autorización integrado es fundamental para gestionar los diferentes roles de usuario (Administrador y Usuario), protegiendo y limitando el acceso a los módulos según los permisos definidos.

Desarrollo

Requerimientos Funcionales (RF)

Los siguientes requerimientos cumplen con los 13 aspectos solicitados para el proyecto:

ID	Requerimiento Funcional	Aspecto Cubierto
RF1	El sistema debe permitir el registro, modificación y eliminación de usuarios y sus datos personales, incluyendo historial de membresías.	Base de datos (1), Formularios (2), Catálogos (4)
RF2	El sistema debe manejar diferentes tipos de suscripciones/membresías (entidades relacionadas) con sus precios y duración.	Base de datos (1), Catálogos (4)
RF3	Debe existir un módulo para el control de asistencias (registro de entrada/salida) de los usuarios, filtrado por fecha y usuario.	Base de datos (1), Búsqueda/Filtrado (12)
RF4	Los administradores deben acceder a un módulo de gestión completo (CRUD) para usuarios, suscripciones, pagos y reportes.	Módulo de administración (3)
RF5	Los usuarios deben poder ver su información de membresía, historial de pagos y asistencias en un módulo para el público en general.	Módulo público (3)
RF6	La plataforma debe implementar formularios para el registro, inicio de sesión y edición de datos del usuario, usando templates de Django.	Formularios/Templates (2), Autenticación (7)
RF7	Se debe implementar un Web API para exponer datos de asistencia y usuarios.	Web API (5)
RF8	La funcionalidad de registro de asistencia debe usar AJAX para una validación en tiempo real sin recargar la página.	Peticiones asíncronas (6)
RF9	El acceso a los módulos estará protegido por el sistema de Autenticación y Autorización de Django,	Autenticación/Autorización (7)

	limitando las funcionalidades según el rol (Administrador, Usuario).	
RF10	La aplicación debe configurarse para protegerse contra las vulnerabilidades más comunes soportadas por Django.	Seguridad (8)
RF11	La base de datos del proyecto debe ser SQLite.	Base de datos (10)
RF12	El sistema debe generar reportes de ingresos, asistencias y usuarios, incluyendo gráficos resumen.	**Reportes y Gráficos (11)**
RF13	Los reportes deben permitir la exportación de datos a formatos PDF y Excel.	**Exportación (13)**

Historias de Usuario

ID	Rol	Quiero...	Para poder...
HU01	Como Administrador	Registrar un nuevo usuario con sus datos personales y la suscripción elegida.	Mantener un control preciso de la base de usuarios.
HU02	Como Usuario	Registrar mi asistencia al gimnasio de forma rápida.	Validar mi acceso y llevar un control de mis visitas.
HU03	Como Administrador	Generar un reporte de ingresos mensuales con un gráfico.	Analizar el desempeño financiero del gimnasio y tomar decisiones (RF11, RF12).
HU04	Como Administrador	Buscar y filtrar usuariospor estado de membresía (activa/vencida).	Contactar rápidamente a usuarios con membresías por vencer (RF12).

HU05	Como Administrador	exportar el listado de pagos del mes a un archivo PDF y Excel.	compartir la información financiera con contabilidad (RF13).
HU06	Como Usuario	iniciar sesión y ver mis datos de perfil.	confirmar la vigencia de mi suscripción y mis asistencias (RF9).

Casos de Uso

Caso de Uso 1: Registro de Asistencia

Campo	Descripción
Nombre	Registrar Asistencia del Usuario
Actor Principal	Usuario(Usuario autenticado con suscripción activa)
Objetivo	Marcar la hora de entrada del Usuario al gimnasio.
Precondición	El Usuario debe estar registrado y su suscripción activa.
Flujo Principal	1. El Usuario escanea su código de identificación o ingresa su ID. 2. El sistema realiza una petición AJAX (RF8) para validar el ID. 3. El sistema verifica la suscripción activa y los permisos. 4. El sistema registra la hora de entrada en la tabla de Asistencias (RF3). 5. Se muestra un mensaje de bienvenida.
Flujo Alternativo	A1: Suscripción Vencida/No Activa: Se muestra un mensaje de error y la razón, redirigiendo a la pantalla de pagos.
Postcondición	Se crea un nuevo registro de asistencia con la hora de entrada.

Caso de Uso 2: Generación de Reporte de Ingresos

Campo	Descripción
Nombre	Generar Reporte de Ingresos por Período
Actor Principal	Administrador
Objetivo	Obtener un informe detallado y gráfico de los ingresos en un rango de fechas.
Precondición	El Administrador debe estar autenticado (RF9) y tener ingresos registrados en la base de datos (RF11).
Flujo Principal	1. El Administrador accede al Módulo de Reportes (RF4). 2. Selecciona un rango de fechas para el reporte (RF12). 3. El sistema procesa los datos y muestra el reporte en formato tabular y con un gráfico resumen (RF12). 4. El Administrador hace clic en el botón "Exportar a PDF/Excel". 5. El sistema genera el archivo y lo descarga (RF13).
Flujo Alternativo	A1: Exportación Fallida: Se muestra un mensaje indicando que la generación del archivo no pudo completarse.
Postcondición	El Administrador tiene acceso al reporte en línea y/o un archivo de exportación del mismo.

Resultado

Esta sección documentará el estado final del producto desarrollado al concluir la fase de implementación, validando su correspondencia con los requerimientos funcionales y la visión arquitectónica definida.

1. Plataforma Desarrollada

El resultado principal del proyecto será la Plataforma Web de Registro y Control para Gimnasios, implementada utilizando el *framework* Django y siguiendo el patrón MVT. El sistema debe cumplir con el objetivo de automatizar la gestión de usuarios, suscripciones, asistencias y reportes.

2. Módulos Funcionales Implementados

Se presentará una descripción de cada módulo y su estado de funcionamiento:

- **Módulo de Administración (RF4):** Acceso protegido para el administrador para el manejo completo (CRUD) de usuarios, suscripciones, y pagos.
- **Módulo de Asistencia (RF3, RF8):** Sistema funcional para el registro de entrada/salida de usuarios, utilizando una petición AJAX para la validación en tiempo real de la suscripción activa.
- **Módulo Público (RF5):** Interfaz para que los usuarios consulten su información de membresía, historial de pagos y asistencias.
- **Módulo de Reportes (RF12, RF13):** Generación de informes de ingresos, asistencias, y usuarios con gráficos resumen, permitiendo la exportación a formatos PDF y Excel.

3. Arquitectura y Stack Tecnológico

Se confirmará la correcta implementación de:

- La base de datos SQLite.
- La exposición de datos mediante el Web API (Django REST Framework).
- La aplicación desplegada en un servicio de *hosting* que emule un ambiente de producción (RF14).

Evaluación

La evaluación medirá el éxito del proyecto confrontando los resultados obtenidos con los objetivos, requerimientos y estándares de calidad definidos al inicio.

1. Cumplimiento de Requerimientos Funcionales

Se adjuntará una tabla de trazabilidad donde cada Requerimiento Funcional (RF) será evaluado, indicando si se cumplió, si está en progreso, o si no se cumplió.

- **Métrica Clave:** Porcentaje de RF implementados y funcionando correctamente (se espera el 100% de los 14 requerimientos).

2. Pruebas de Calidad y Rendimiento

- **Pruebas Unitarias:** Verificación de la lógica de negocio en Modelos y Vistas, así como la validación de suscripción activa para el registro de asistencia.
- **Pruebas de Integración:** Verificación de la comunicación entre módulos (ej., que el módulo de Pagos actualice correctamente el estado de la SuscripciónUsuario en el módulo de Membresías).

- **Pruebas de Seguridad (RF10):** Evaluación de la resistencia de la aplicación contra vulnerabilidades como CSRF y XSS.
- **Pruebas de Experiencia de Usuario:** Evaluación de la rapidez de respuesta del sistema, especialmente en el registro de asistencia mediante AJAX.

3. Criterios de Éxito de las Historias de Usuario

Se verificará que las historias de usuario (HU01 a HU06) se hayan implementado completamente, lo que implica que la funcionalidad es útil y usable para el Administrador y el Usuario.

Conclusiones

El avance del proyecto **"Plataforma de Registro y Control para Gimnasios"** ha permitido formalizar y estructurar la solución mediante una exhaustiva fase de análisis, la cual constituye un cimiento sólido para la etapa de desarrollo. Se ha logrado definir con precisión un conjunto claro de Requerimientos Funcionales, un entendimiento detallado de las Historias de Usuario, y la modelización de los flujos de trabajo a través de los Casos de Uso clave.

Actualmente, el proyecto se encuentra en la crucial fase de implementación, donde se busca materializar la arquitectura definida. Se mantiene firme el compromiso de lograr el objetivo principal: construir una plataforma que automatice por completo la gestión del gimnasio, erradicando los errores y la ineficiencia inherentes a los métodos manuales.

La robustez tecnológica del desarrollo está asegurada por el uso de Django y el patrón MVT, lo cual facilita la implementación modular de las funcionalidades más exigentes. Estamos enfocados en integrar la capacidad de un Web API, y los módulos de reportes con su respectiva exportación a PDF y Excel. La culminación de estas funcionalidades permitirá la entrega de un producto con alta ventaja competitiva y valor agregado para el negocio.

A pesar de que el desarrollo está en curso, se han identificado retos técnicos importantes que ya están siendo abordados, como la correcta configuración y migración a la base de datos SQLite y el aprendizaje práctico en la implementación de Django REST Framework. Estos desafíos, lejos de ser obstáculos, representan oportunidades clave de aprendizaje y consolidación de conocimientos para el equipo, asegurando que cuando la aplicación se lance a producción, cumpla cabalmente con todos los estándares de seguridad y funcionalidad planificados.

Bibliografía

Se citan las referencias principales para el desarrollo del *backend* y el diseño de la API:

- Django Documentation. (s. f.). Django Software Foundation. Recuperado el 4 de octubre de 2025, de https://docs.djangoproject.com/es/

- Django REST Framework. (s. f.). Home. Recuperado el 4 de octubre de 2025, de https://www.django-rest-framework.org/
- PostgreSQL Global Development Group. (s. f.). *Documentation*. Recuperado el 5 de octubre de 2025, de https://www.postgresql.org/docs/
- W3C. (s. f.). *AJAX - Asynchronous JavaScript and XML*. Recuperado el 6 de octubre de 2025, de https://www.w3.org/standards/webdesign/ajax (Referencia técnica para la implementación de peticiones asíncronas).