UNIDAD 1.

Introducción al desarrollo web.

(El estudiante identificará las herramientas y elementos para la estructura de un documento HTML.)

Tema I. Conceptos generales del desarrollo web.

Evidencia de aprendizaje Tema I –RESUMEN

Temas	Saber	Saber hacer	Resultado de Aprendizaje
Conceptos generales del desarrollo web	Identificar los conceptos de aplicaciones web: Internet y web, aplicación web, front end, back end, hosting, protocolos (http, https), modelo cliente - servidor.	Instalar y configurar el ambiente de desarrollo Web.	Los estudiantes identifican los conceptos relacionados con las tecnologías para el desarrollo web.
Evidencia de aprendizaje			
Examen de Opción Múltiple, Tema: Conceptos y Configuración de un Ambiente de Desarrollo Web. Este examen evalúa los conocimientos sobre la **instalación, configuración y verificación de un entorno de desarrollo web**. El estudiante deberá identificar conceptos básicos, reconocer el software necesario, comprobar su correcta instalación y mostrar los programas instalados en su PC relacionados con el entorno web.			

Información sobre el profesor

Profesor	Correo	Días de Clase
Bernardo Prado Díaz	Tenor_prado@yahoo.com.mx	Lunes y Martes

Introducción al Desarrollo Web con Python y Django

1 SABER – Dimensión Conceptual

Objetivo: Comprender los conceptos fundamentales del desarrollo web y su aplicación en entornos reales.

A) Conceptos Generales del Desarrollo Web

- **Internet:**

Definición: Red global que conecta millones de dispositivos para intercambiar información.

Ejemplo: Consultar un sitio como https://www.google.com.

¿Qué significa WWW? World Wide Web

Aplicación en Django: Django despliega aplicaciones en servidores accesibles por Internet.

• **Web:**
Definición: Conjunto de recursos accesibles mediante navegadores usando protocolos como HTTP/HTTPS.

Ejemplo: Página de noticias en línea.

Aplicación en Django: La interfaz generada por Django se accede vía navegador.

• **Aplicación web:** Un programa al que se puede acceder usando un navegador web, como Google Maps o Facebook. A diferencia de un programa de escritorio, no necesita ser instalado en tu computadora. Definición: Programa que se ejecuta en un servidor y se accede desde un navegador, sin necesidad de instalarse localmente.

Ejemplo: Gmail, Google Drive.

Aplicación en Django: Tu proyecto Django (por ejemplo, un sistema de ventas).

• **Front-end:** Es la parte de la aplicación web con la que el usuario interactúa directamente. Es todo lo que ves y usas en la pantalla, como los botones, imágenes y textos. Se crea con lenguajes como HTML, CSS y JavaScript.

Definición: Parte visual con la que interactúa el usuario. Incluye HTML, CSS, JavaScript.

Ejemplo: Botones, menús, formularios de registro.

Aplicación en Django: Django usa plantillas HTML (templates) para mostrar datos al usuario.

• **Back-end:** Es la parte de la aplicación que no ves. Se encarga de la lógica interna, el manejo de datos, la interacción con bases de datos y la autenticación de usuarios. Funciona en el servidor y está programado en lenguajes como Python, Java o Node.js.
Definición: Lógica y procesamiento que ocurre en el servidor.

Ejemplo: Validar usuarios, guardar datos en base de datos.

Aplicación en Django: Django gestiona el acceso a datos con views y models.

• **Hosting:** Es el servicio que te permite publicar un sitio web en Internet. Una empresa de hosting aloja los archivos de tu sitio en sus servidores, para que cualquier persona en el mundo pueda acceder a él.

Definición: Servicio donde se almacena y ejecuta tu aplicación web.

Ejemplo: Servidores como Heroku, PythonAnywhere.

Aplicación en Django: Subir el proyecto Django a un hosting para que sea accesible públicamente.

• **Protocolo HTTP (Hypertext Transfer Protocol):** Es el lenguaje que usan el cliente y el servidor para comunicarse. Define las reglas para solicitar y enviar datos, como páginas web, imágenes y otros archivos a través de Internet.
Definición: Protocolo de transferencia de hipertexto que gestiona la comunicación cliente-servidor sin cifrado.

Ejemplo: Una solicitud GET para ver una página.

Aplicación en Django: Django recibe peticiones HTTP y responde con HTML.

• **Protocolo HTTPS:**
Definición: Versión segura de HTTP que cifra los datos.

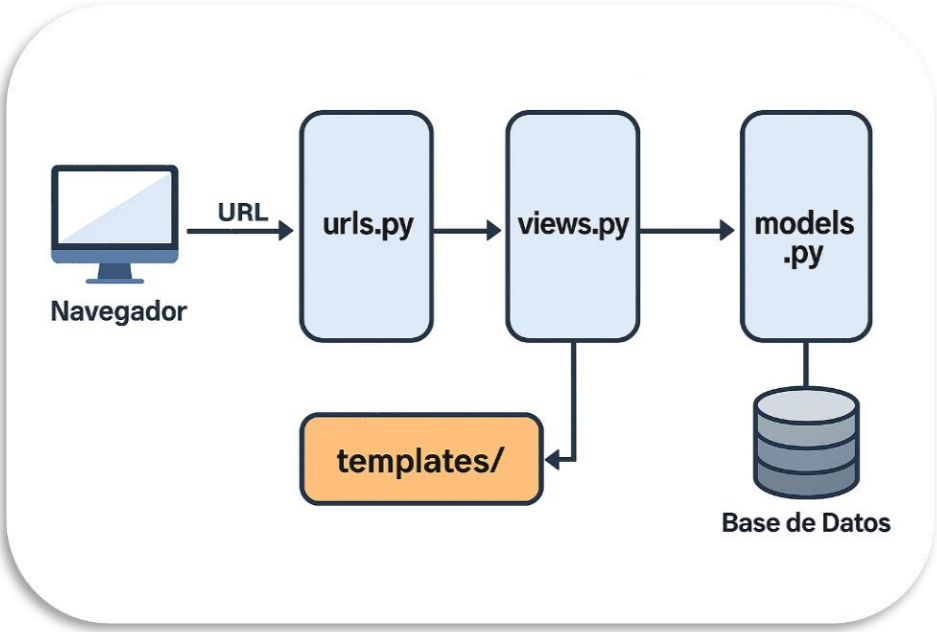
Ejemplo: Páginas de bancos o compras en línea.

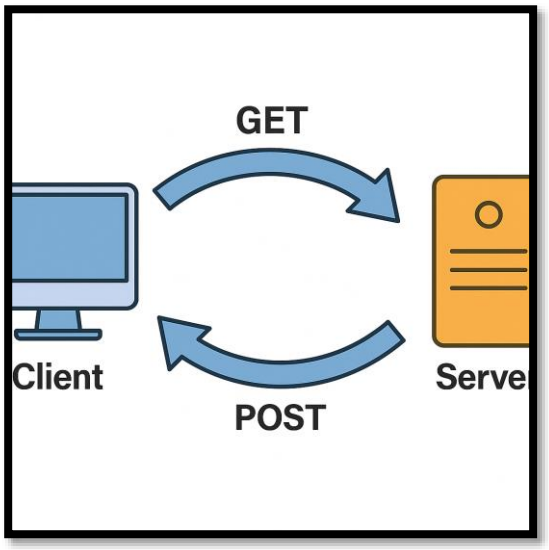
Aplicación en Django: Configurar Django para usar certificados SSL.

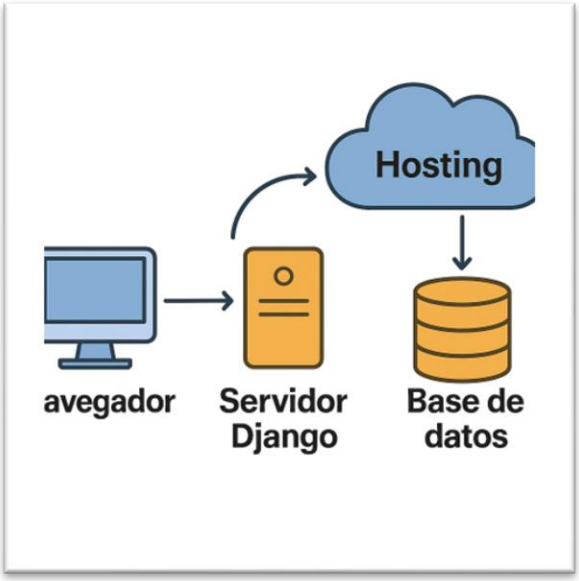
• **Modelo Cliente-Servidor:**
Definición: Arquitectura donde el cliente (navegador) solicita recursos y el servidor responde. Es un modelo de comunicación donde un "cliente" (por ejemplo, tu navegador web) le pide un servicio a un "servidor" (una computadora que aloja un sitio web). El servidor procesa la solicitud y envía una respuesta.

Ejemplo: Un usuario pide una página y el servidor la envía.

Aplicación en Django: Django actúa como servidor que recibe peticiones y envía respuestas.







B) Glosario Detallado

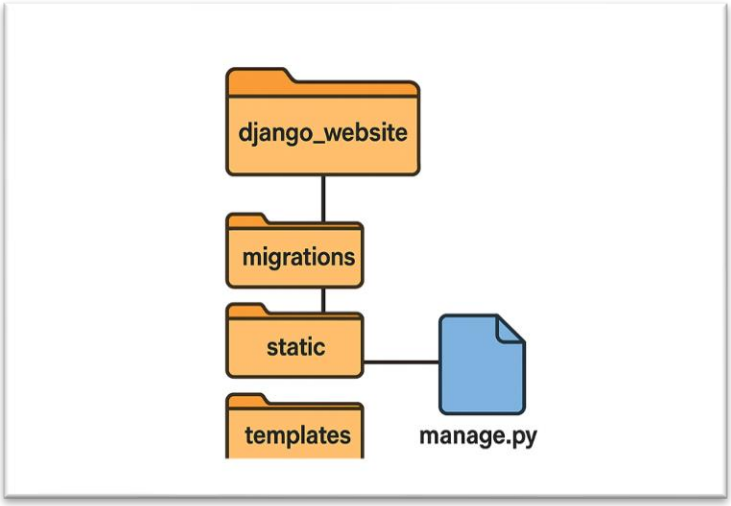
- Servidor web → Computadora que entrega contenido a través de Internet.

- Framework → Conjunto de herramientas y librerías que facilitan el desarrollo. (Django es un framework de Python).

- URL → Dirección de un recurso en la web.

- Base de datos → Sistema para almacenar información estructurada (Ej: MySQL, PostgreSQL).

- Renderizado → Proceso de transformar datos en una interfaz visual para el usuario.



2 SABER HACER – Dimensión Actuacional

Objetivo: Poner en práctica la instalación y configuración del ambiente de desarrollo web con Django.

Pasos Prácticos: Configuración del Ambiente

- Instalar Python (mínimo versión 3.10).

- Instalar Django: pip install django

- Crear un Proyecto: django-admin startproject mi_proyecto

- Ejecutar Servidor Local: python manage.py runserver

- Crear una App Dentro del Proyecto: python manage.py startapp blog

- Configurar Plantillas y Base de Datos en settings.py.

- Ejecutar Migraciones: python manage.py migrate

- Acceder al Proyecto en el navegador: URL local: http://127.0.0.1:8000

Ejemplo Real en Django

Sistema de registro de alumnos:

Front-End: Formulario HTML para ingresar datos.

Back-End: views.py procesa la información y la guarda en la base de datos con un modelo Alumno.

```
from django.db import models

class Alumno(models.Model):
```

```
    nombre = models.CharField(max_length=100)
    correo = models.EmailField()
    carrera = models.CharField(max_length=100)

    def __str__(self):
        return self.nombre
```

3 SER Y CONVIVIR – Dimensión Socioafectiva

Objetivo: Fomentar habilidades de colaboración y comunicación en el desarrollo web.

Buenas Prácticas de Trabajo en Equipo

- Comunicación clara: Usar herramientas como Trello, Slack o Discord.

- Control de versiones: Usar Git y GitHub para colaborar en el código.

- Roles definidos: Front-End Developer, Back-End Developer, Full-Stack Developer.

- Resolución de conflictos: Escuchar, proponer soluciones y documentar acuerdos.

- Respeto y empatía: Valorar el trabajo de los compañeros.

Ejercicio Socioafectivo en Clase

Dividir a los estudiantes en equipos:
1. Un grupo se encarga del Front-End (HTML/CSS/JS con plantillas Django).
2. Otro grupo del Back-End (vistas y modelos en Django).
3. Integrar el trabajo y hacer una presentación final.

Mejoras con Imágenes y Prácticas Reales

1. Imágenes ilustrativas

Se agregan diagramas para visualizar la arquitectura de Django, el flujo de peticiones y la estructura del proyecto.

- ✓ Arquitectura Cliente-Servidor (insertar imagen de diagrama).
- ✓ Estructura de un Proyecto Django (insertar imagen de carpetas).
- ✓ Flujo de Peticiones HTTP (insertar esquema de GET y POST).
- ✓ Ciclo MVT en Django (insertar diagrama).

2. Prácticas Reales Paso a Paso

Ejercicio 1 – Registro de usuarios con autenticación.

Ejercicio 2 – Blog simple con CRUD de publicaciones.

Ejercicio 3 – Mini API con Django REST Framework.

Ejemplo de Código:

```
from django.db import models

class Producto(models.Model):
    nombre = models.CharField(max_length=100)
    precio = models.DecimalField(max_digits=10, decimal_places=2)
    stock = models.IntegerField()

    def __str__(self):
        return f"{self.nombre} (${self.precio})"
```

3. Actividades Socioafectivas

Trabajo en equipo: cada grupo desarrolla un módulo (ejemplo: alumnos, materias, calificaciones).

Presentación final tipo demo: integrar los módulos en un sistema completo.

/api/productos


```
{
  "id":1,
  "nombre":"Laptop",
  "precio":1200.0,
  "stock":5
}
```