UNIDAD 3.

Desarrollo de aplicaciones web (Back end).

(El estudiante demostrará el proceso para el desarrollo de una aplicación web.)

Tema IV. Funciones básicas de base de datos.

Evidencia de aprendizaje Tema IV –RESUMEN

Temas	Saber	Saber hacer	Resultado de Aprendizaje
Funciones básicas de base de datos	Definir el proceso de conexión, inserción, modificación, eliminación y consulta de Bases de Datos desde aplicaciones Web.	Desarrollar aplicaciones Web que permitan la gestión de información en Bases de Datos.	Los estudiantes identifican los métodos de intercambio de información hacia el servidor y la base de datos.
Evidencia de aprendizaje			
A partir de un caso práctico realizar una aplicación web que contenga lo siguiente: Páginas web, hojas de estilos, gestión de contenido de base de datos desde formularios, implementación de seguridad por medio de sesiones.			

Información sobre el profesor

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

SABER: Fundamentos teóricos sobre funciones básicas de bases de datos

¿Qué es una base de datos?

Una base de datos es un sistema organizado para almacenar, gestionar y recuperar información. En aplicaciones web, se usa para guardar datos de usuarios, productos, transacciones, etc.

Funciones básicas

Las funciones esenciales que una aplicación web realiza con una base de datos son:

Función	Descripción
Conexión	Establecer un vínculo entre la aplicación y la base de datos
Inserción	Agregar nuevos registros (por ejemplo, registrar un usuario)
Modificación	Actualizar datos existentes (como editar un perfil)
Eliminación	Borrar registros (como eliminar una cuenta)
Consulta	Obtener información (como mostrar productos disponibles)

Estas operaciones se conocen como CRUD: ***Create, Read, Update, Delete*.**

¿Cómo se conectan las aplicaciones web a las bases de datos?

Las aplicaciones web usan drivers o ORMs (Object-Relational Mappers) para comunicarse con la base de datos. Ejemplos:

- PHP + MySQL: usa mysqli o PDO
- Python (Django): usa ORM de Django
- JavaScript (Node.js): usa Sequelize, Mongoose, etc.

Seguridad y buenas prácticas

- Usar consultas preparadas para evitar *SQL Injection*
- Validar los datos antes de insertarlos
- Cifrar información sensible (como contraseñas)
- Controlar los permisos de acceso a la base de datos

SABER HACER: Desarrollo práctico de funciones básicas

A continuación, te muestro cómo implementar cada función usando Python con Django como ejemplo. El enfoque es aplicable a cualquier lenguaje con sus respectivas herramientas.

1. Conexión a la base de datos

Django se conecta automáticamente usando la configuración en settings.py:

python

```
# settings.py

DATABASES = {

   'default': {

      'ENGINE': 'django.db.backends.mysql',  # o 'sqlite3', 'postgresql'

      'NAME': 'mi_base',

      'USER': 'usuario',

      'PASSWORD': 'contraseña',

      'HOST': 'localhost',

      'PORT': '3306',

   }

}
```

2. Inserción de datos

python

```
# models.py

from django.db import models
```

```
class Producto(models.Model):
    nombre = models.CharField(max_length=100)
    precio = models.DecimalField(max_digits=6, decimal_places=2)
```

 # views.py

```
def agregar_producto(request):
    nuevo = Producto(nombre="Laptop", precio=999.99)
    nuevo.save()  # Inserta en la base de datos
```

3. Consulta de datos

python

```
def mostrar_productos(request):
    productos = Producto.objects.all()  # SELECT * FROM producto
    return render(request, 'productos.html', {'productos': productos})
```

4. Modificación de datos

python

```
def editar_producto(request, id):
    producto = Producto.objects.get(id=id)
    producto.precio = 899.99
    producto.save()  # UPDATE producto SET precio=899.99 WHERE id=id
```

5. Eliminación de datos

python

```
def eliminar_producto(request, id):
    producto = Producto.objects.get(id=id)
    producto.delete()  # DELETE FROM producto WHERE id=id
```

6. Validación y seguridad

python

```
from django.core.exceptions import ValidationError

def agregar_producto(request):
    nombre = request.POST['nombre']
    precio = float(request.POST['precio'])

    if precio < 0:
        raise ValidationError("El precio no puede ser negativo")

    Producto.objects.create(nombre=nombre, precio=precio)
```

Resultado de aprendizaje

- **Comprender el ciclo completo de interacción entre una aplicación web y una base de datos.**
- **Implementar operaciones CRUD de forma segura y eficiente.**
- **Aplicar buenas prácticas de validación y protección de datos.**
- **Desarrollar sistemas web funcionales que gestionen información real.**

Funciones Básicas de Base de Datos con Python y Django

1 SABER – Dimensión Conceptual

Objetivo: Comprender el proceso de conexión, inserción, modificación, eliminación y consulta de bases de datos desde aplicaciones web utilizando Python y Django.

A) Conceptos Clave

• Conexión a Base de Datos:
Definición: Proceso mediante el cual la aplicación web se comunica con el gestor de base de datos.
Ejemplo: Ejemplo: Configurar la conexión en settings.py de Django.

• Inserción:
Definición: Agregar nuevos registros en una tabla de base de datos.
Ejemplo: Ejemplo: Añadir un nuevo usuario al sistema.

• Modificación:
Definición: Actualizar información existente en la base de datos.
Ejemplo: Ejemplo: Cambiar el correo electrónico de un usuario.

• Eliminación:
Definición: Borrar registros de una base de datos.
Ejemplo: Ejemplo: Eliminar un producto de un catálogo.

• Consulta:
Definición: Recuperar datos de la base de datos para mostrarlos o procesarlos.
Ejemplo: Ejemplo: Mostrar una lista de productos disponibles.

2 SABER HACER – Dimensión Actuacional

Objetivo: Implementar en Django las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para la gestión de información.

A) Configuración de Conexión en Django

```
# archivo settings.py
DATABASES = {
   'default': {
      'ENGINE': 'django.db.backends.mysql',
      'NAME': 'mi_base_datos',
      'USER': 'usuario_db',
      'PASSWORD': 'contraseña_db',
      'HOST': 'localhost',
      'PORT': '3306',
   }
```

}

B) Ejemplo de Modelo

```
# archivo models.py
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()
```

C) Ejemplos de Operaciones CRUD

```
# Inserción
nuevo_producto = Producto(nombre="Laptop", precio=1500.00, stock=10)
nuevo_producto.save()

# Consulta
productos = Producto.objects.all()

# Modificación
producto = Producto.objects.get(id=1)
producto.precio = 1400.00
producto.save()

# Eliminación
producto = Producto.objects.get(id=1)
producto.delete()
```

3 SER Y CONVIVIR – Dimensión Socioafectiva

Objetivo: Fomentar el trabajo colaborativo y la responsabilidad en el manejo de datos en aplicaciones web.

- Mantener copias de seguridad de la base de datos antes de modificaciones importantes.

- Definir roles claros para la gestión de datos (desarrollador, administrador de base de datos).

- Proteger el acceso a datos sensibles mediante autenticación y permisos.

- Documentar cambios realizados en la base de datos para un mejor control.

4 Glosario

- CRUD → Operaciones básicas: Crear, Leer, Actualizar, Eliminar.

- Modelo → Estructura de datos en Django que representa una tabla en la base de datos.

- QuerySet → Conjunto de registros recuperados de la base de datos.

- Migración → Proceso para aplicar cambios en la estructura de la base de datos.

- ORM → Mapeo objeto-relacional que permite interactuar con la base de datos usando objetos Python.