

Modelado Back-End (Django)

1.1 Diagrama Entidad Relación (ER)

★ Actividad 1. Ronda de preguntas rápidas.

1. ¿Qué es un modelo de base de datos?
2. ¿Qué es un Diagrama Entidad-Relación (ER)?
3. Tipos de relaciones:
4. ¿En qué se diferencia ForeignKey de ManyToManyField?

ForeignKey:

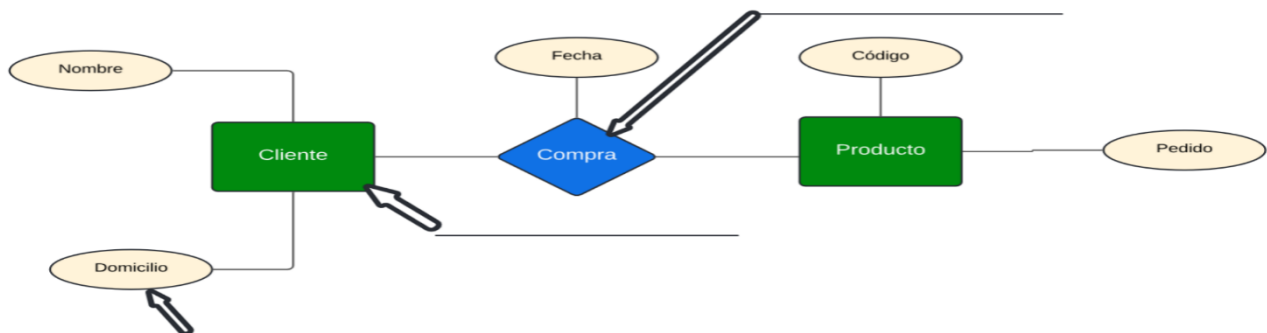
ManyToManyField:

5. ¿Para qué sirve django-extensions + graphviz? y ejemplos simples.

Actividad 1. Realiza una investigación de los siguientes conceptos en contexto de Bases de Datos:

Entidad	
Atributo	
Relación	

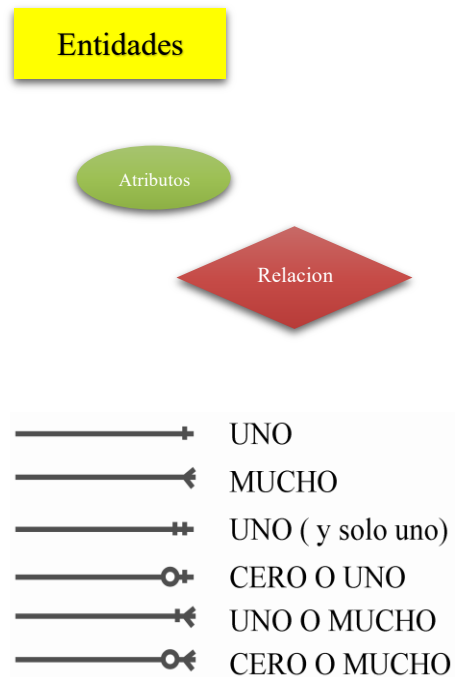
Actividad 2. En el diagrama siguiente se aplican los conceptos que investigaste en el ejercicio anterior, escribe el nombre de los conceptos en la parte que se aplique y explica su relación.



Símbolo	Representa	Símbolo	Representa

 Actividad 2. Diseño de un Modelo ER de un Proyecto Django para una "Tienda Online".

Dibujar de manera individual el diagrama ER en papel, las principales entidades son: Producto (nombre, precio, descripción), Cliente (nombre, email, dirección), Pedido (fecha, cliente, productos). Propón otras entidades y sus relaciones.

1. Entidad y atributos	2. Relaciones	3. Cardinalidad
Diagrama Entidad – Relación básico		Nomenclatura
		 Entidades Atributos Relacion  + UNO  MUCHO  UNO (y solo uno)  CERO O UNO  UNO O MUCHO  CERO O MUCHO

Prácticas. Herramientas para Generar Diagramas ER desde Django.

Para generar el diagrama anterior desde Django; existen dos alternativas:

- **Práctica 1 (Automatización):** django-extensions + graphviz
- **Práctica 2 (Creación Compendio):** Lucidchart / Draw.io

Revisión, retroalimentación y registro en archivo de Yo No Abandono.

Resumen de conceptos:

- **Modelo de Base de Datos:**
- **Diagrama ER:**
- **Django ORM:**
- **Herramientas:**



LECTURA: EL MUNDO ANTES DEL ORM



¿QUÉ ES UNA **BASE DE DATOS RELACIONAL**?

Imagina una Biblioteca Gigante

Una base de datos es como una BIBLIOTECA organizada:

```
BIBLIOTECA = {
  'LIBROS': [
    {'id': 1, 'titulo': 'Harry Potter', 'autor_id': 101},
    {'id': 2, 'titulo': 'El Señor de los Anillos', 'autor_id': 102} ],
  'AUTORES': [
    {'id': 101, 'nombre': 'J.K. Rowling'},
    {'id': 102, 'nombre': 'J.R.R. Tolkien'} ],
  'ESTUDIANTES': [
    {'id': 201, 'nombre': 'Ana', 'libro_prestado_id': 1}
  ]
}
```

Tipos de Bases de Datos:

- **MySQL** → Como una biblioteca pública grande 🏢
- **PostgreSQL** → Como una biblioteca universitaria 🎓
- **SQLite** → Como tu biblioteca personal en el celular 📱

¿Por qué se llama "**relacional**"?

Porque las tablas se **relacionan** entre sí, como los libros con sus autores.

✍️ Actividad 3. Tabla comparativa entre una tabla y una base de datos.

Tabla	Base de Datos
Partes de una Tabla en Base de Datos: • Filas: representa un registro. • Columnas: indica qué tipo de información contiene, como texto, número, fecha, etc. • Clave Primaria: columna especial que contiene valores únicos para cada fila en la tabla. Ayuda a identificar y acceder rápidamente a registros específicos. • Claves Externas: En algunas tablas, se utilizan claves externas para establecer relaciones con otras tablas en la base de datos. • Campo: Se refiere al nombre de la columna. Es un dato único y además se le establece un tipo de dato. • Registro: Se refiere a cada fila que conforma la tabla, dicho de otra manera, son los datos y registros que almacenamos.	 Una base de datos es un contenedor basado en software estructurado para recopilar y almacenar información para que pueda recuperarse, agregarse, actualizarse o eliminarse de forma automática. La estructura de una base de datos es la tabla. Las columnas identifican los datos (atributos) en la tabla y las filas son los registros de información. Las tablas se ven como una hoja de cálculo, pero las tablas se pueden manipular y actualizar de una manera que las hojas de cálculo no pueden, lo que hace que una base de datos sea una herramienta muy valiosa.



Dibuja y representa una tabla

Dibuja y representa una Base de Datos

🧠 ¿QUÉ ES SQL? EL LENGUAJE DE LAS BASES DE DATOS.

SQL es como dar órdenes a un robot asistente.

LAS 4 ÓRDENES BÁSICAS:

- Comandos básicos de SQL

SELECT: _____

INSERT: _____

DELETE: _____

UPDATE: _____

1. SELECT → "MOSTRAR"



Ejemplos

-- "Muéstrame todos los estudiantes" **SELECT * FROM** estudiantes;

-- "Muéstrame solo los nombres de los estudiantes" **SELECT nombre FROM** estudiantes;

-- "Muéstrame los estudiantes llamados Ana" **SELECT * FROM** estudiantes **WHERE** nombre = 'Ana';

Casos de la vida real:

" _____ " 

2. INSERT → "AGREGAR" +

-- "Agrega un nuevo estudiante" **INSERT INTO** estudiantes (nombre, edad, grado)
VALUES ('Carlos', 15, '9no');

Casos de la vida real:

" _____ " 

3. UPDATE → "ACTUALIZAR"



-- "Cambia la edad de Carlos a 16 años" **UPDATE** estudiantes

SET edad = 16

WHERE nombre = 'Carlos';

Casos de la vida real:

" _____ " 

4. DELETE → "ELIMINAR"



-- "Elimina al estudiante Carlos"

DELETE FROM estudiantes

WHERE nombre = 'Carlos';

Casos de la vida real:

" _____ " 



COMANDOS SQL

Comando	Función	Ejemplo
CREATE DATABASE		CREATE DATABASE escuela;
CREATE TABLE		CREATE TABLE alumnos (...);
INSERT INTO		INSERT INTO tabla VALUES (...);
SELECT		SELECT * FROM tabla;
UPDATE		UPDATE tabla SET campo=valor;
DELETE		DELETE FROM tabla WHERE...;
WHERE		WHERE Promedio > 8.0
ORDER BY		ORDER BY Nombre ASC
COUNT()		SELECT COUNT(*) FROM...
AVG()		SELECT AVG(Promedio) FROM...

Operadores de Comparación

Operador	Significado	Ejemplo
=		WHERE edad = 15
>		WHERE precio > 100
<		WHERE stock < 10
>=		WHERE promedio >= 8.0
<=		WHERE edad <= 18
<>		WHERE estado <> 'inactivo'

Operadores Especiales

Operador	Significado	Ejemplo
BETWEEN		WHERE edad BETWEEN 15 AND 18
LIKE		WHERE nombre LIKE 'A%'
IN		WHERE curso IN ('Matemáticas', 'Física')

FUNCIONES DE AGREGACIÓN (CÁLCULOS)

Función	Significado	Ejemplo
COUNT()		SELECT COUNT(*) FROM estudiantes
SUM()		SELECT SUM(ventas) FROM productos
AVG()		SELECT AVG(calificacion) FROM notas
MAX()		SELECT MAX(precio) FROM productos
MIN()		SELECT MIN(edad) FROM estudiantes

TIPOS ENTEROS - Para números sin decimales

Tipo Dato	Significado	Rango	Ejemplo de Uso
TINYINT	Número _____	-128 a 127	Edades, estados (sí/no)
SMALLINT	Número _____	±32,768	Códigos postales, cantidades
MEDIUMINT	Número _____	±8 millones	Población de ciudades
INTEGER	Número _____	±2.1 billones	IDs, contadores grandes
BIGINT	Número _____	±9.2×10 ¹⁸	IDs únicos globales

TIPOS DECIMALES - Para números con punto decimal

Tipo Dato	Significado	Precisión	Ejemplo de Uso
DECIMAL	Número _____	Precisión perfecta	Dinero, precios
NUMERIC	Igual que _____	Precisión perfecta	Mediciones exactas
FLOAT	Número _____	~7 decimales	Cálculos científicos
DOUBLE	Número _____	~15 decimales	Cálculos complejos

TIPOS DE TEXTO - Para almacenar caracteres

Tipo Dato	Significado	Tamaño Máximo	Ejemplo de Uso
CHAR	Texto de _____	255 chars	Códigos, siglas (RFC, ISBN)
VARCHAR	Texto de _____	65,535 chars	Nombres, direcciones, emails
TINYTEXT	Texto _____	255 chars	Títulos breves, nombres
TEXT	Texto _____	65 KB	Descripciones, comentarios
MEDIUMTEXT	Texto _____	16 MB	Artículos, documentos
LONGTEXT	Texto _____	4 GB	Libros, enciclopedias

TIPOS BINARIOS - Para archivos y datos crudos

Tipo Dato	Significado	Tamaño Máximo	Ejemplo de Uso
BINARY	Datos binarios longitud FIJA	255 bytes	Hashes fijos, códigos
VARBINARY	Datos binarios longitud VARIABLE	65,535 bytes	Contraseñas encriptadas
TINYBLOB	Archivo _____	255 bytes	Iconos, thumbnails
BLOB	Archivo _____	65 KB	Imágenes pequeñas, PDFs
MEDIUMBLOB	Archivo _____	16 MB	Fotos, música, videos cortos
LOB	Archivo _____	4 GB	Videos largos, backups

TIPOS ESPECIALES

Tipo Dato	Significado	Ejemplo	Uso
ENUM	Lista de opciones fijas	ENUM('soltero','casado')	Estado civil, categorías
SET	Múltiples opciones simultáneas	SET('deporte','musica','arte')	Intereses, etiquetas
BOOLEAN	Valor verdadero/falso	TRUE, FALSE	Estados activo/inactivo

=====

SCRIPT SQL: BASE DE DATOS CECYTEM

=====

-- 📁 1. CREAR BASE DE DATOS --

```
CREATE DATABASE 'cecytem';  
USE 'cecytem';
```

-- 📖 2. CREAR TABLA ALUMNOS --

```
CREATE TABLE 'alumnos' (  
  'NoControl' INT NOT NULL,  
  'Nombre' VARCHAR(50) NOT NULL,  
  'ApellidoPaterno' VARCHAR(50) NOT NULL,  
  'ApellidoMaterno' VARCHAR(50) NOT NULL,  
  'FechaNacimiento' DATE NULL,  
  'Genero' ENUM('M', 'F') NULL,  
  'Promedio' DECIMAL(3,1) NULL,  
  'Email' VARCHAR(100) NULL,  
  'Telefono' VARCHAR(15) NULL,  
  'Activo' TINYINT DEFAULT 1,  
  'FechaRegistro' TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  PRIMARY KEY ('NoControl')  
);
```

-- 📖 3. CREAR TABLA PROFESORES --

```
CREATE TABLE 'profesores' (  
  'IdProfesor' INT AUTO_INCREMENT PRIMARY KEY,  
  'Nombre' VARCHAR(50) NOT NULL,  
  'ApellidoPaterno' VARCHAR(50) NOT NULL,  
  'ApellidoMaterno' VARCHAR(50) NOT NULL,  
  'Especialidad' VARCHAR(100) NULL,  
  'Email' VARCHAR(100) NULL,  
  'Telefono' VARCHAR(15) NULL,  
  'FechaContratacion' DATE NULL,  
  'Activo' TINYINT DEFAULT 1  
);
```

-- 📁 4. CREAR TABLA MATERIAS --

```
CREATE TABLE 'materias' (  
  'ClaveMateria' VARCHAR(10) PRIMARY KEY,  
  'NombreMateria' VARCHAR(100) NOT NULL,  
  'Creditos' INT NULL,  
  'HorasSemana' INT NULL,  
  'Semestre' INT NULL  
);
```

-- 📖 6. CREAR TABLA ALUMNOS_GRUPOS (Relación Muchos a Muchos) --

```
CREATE TABLE 'alumnos_grupos' (  
  'NoControl' INT NOT NULL,  
  'IdGrupo' INT NOT NULL,  
  'FechaInscripcion' DATE NULL,  
  PRIMARY KEY ('NoControl', 'IdGrupo'),  
  FOREIGN KEY ('NoControl') REFERENCES 'alumnos'('NoControl'),  
  FOREIGN KEY ('IdGrupo') REFERENCES 'grupos'('IdGrupo')  
);
```

-- + 8. INSERTAR DATOS EN TABLA ALUMNOS --

INSERT INTO 'alumnos' ('NoControl', 'Nombre', 'ApellidoPaterno', 'ApellidoMaterno', 'FechaNacimiento', 'Genero', 'Promedio', 'Email', 'Telefono') **VALUES**
(59120083, 'Libeth', 'Alcantara', 'Blas', '2000-05-15', 'F', 8.5, 'libeth.alcantara@cecytem.edu.mx', '555-123-4567'),
(59120084, 'Carlos', 'Gómez', 'Pérez', '2001-03-20', 'M', 9.1, 'carlos.gomez@cecytem.edu.mx', '555-234-5678'),

-- 9. INSERTAR DATOS EN TABLA PROFESORES --

INSERT INTO 'profesores' ('Nombre', 'ApellidoPaterno', 'ApellidoMaterno', 'Especialidad', 'Email', 'Telefono', 'FechaContratacion') **VALUES**
('María', 'Gutiérrez', 'López', 'Matemáticas', 'maria.gutierrez@cecytem.edu.mx', '555-111-2233', '2018-08-15'),
('Roberto', 'Martínez', 'Silva', 'Física', 'roberto.martinez@cecytem.edu.mx', '555-222-3344', '2019-01-20'),

-- 14. CONSULTAS DE EJEMPLO --

-- Consulta 1: Mostrar todos los alumnos

SELECT * FROM 'alumnos';

-- Consulta 2: Alumnos con promedio mayor a 8.0

SELECT 'NoControl', 'Nombre', 'ApellidoPaterno', 'Promedio'
FROM 'alumnos'
WHERE 'Promedio' > 8.0
ORDER BY 'Promedio' **DESC**;

-- Consulta 3: Contar alumnos por género

SELECT 'Genero', **COUNT(*) AS** 'Total'
FROM 'alumnos'
GROUP BY 'Genero';

-- 15. ACTUALIZACIONES DE EJEMPLO --

-- Actualizar promedio de un alumno

UPDATE 'alumnos'
SET 'Promedio' = 9.0
WHERE 'NoControl' = 59120083;

-- Actualizar email de un alumno

UPDATE 'alumnos'
SET 'Email' = 'libeth.nueva@cecytem.edu.mx'
WHERE 'NoControl' = 59120083;

-- Actualizar calificación

UPDATE 'calificaciones'
SET 'Calificacion' = 9.5
WHERE 'NoControl' = 59120083 **AND**
'ClaveMateria' = 'MAT101';

-- 16. ELIMINACIONES DE EJEMPLO --

-- Eliminar una calificación específica

DELETE FROM 'calificaciones'
WHERE 'NoControl' = 59120087 **AND** 'ClaveMateria' = 'RED101';

-- Eliminar un alumno (si no tiene registros relacionados)

DELETE FROM 'alumnos'
WHERE 'NoControl' = 59120087;

-- 17. CONSULTAS ESTADÍSTICAS

-- Total de alumnos

SELECT COUNT(*) AS 'TotalAlumnos' **FROM** 'alumnos';

-- Promedio general de todos los alumnos

SELECT AVG('Promedio') **AS** 'PromedioGeneral' **FROM** 'alumnos';

=====
FIN DEL SCRIPT
=====



Insertar instrucciones utilizando los comandos básicos de SQL

Crear Base de Datos: ControlEscolarCECyTEM	SQL
Insertar instrucciones utilizando los comandos básicos de SQL Crear las tablas	
Materias	
IdMateria Nombre	SQL
Semestre	
IdSemestre Semestre	SQL
Grupo	
Idgrupo Grupo	SQL
Alumnos	
NoControl Nombre ApellidoPaterno ApellidoMaterno Promedio	SQL
Docentes	
NoEmpleado Nombre ApellidoPaterno ApellidoMaterno Profesion	SQL

Insertar datos utilizando los comandos básicos de SQL

- Materias

Colocar nombre de las materias de 3ero

Colocar nombre de las materias de 3ero		
IdMateria	Materia	SQL
001		
002		
003		
004		
005		
006		
007		

- Semestre

IdSem	Semestre	SQL
1	Primero	
2	Segundo	
3	Tercero	
4	Cuarto	
5	Quinto	
6	Sexto	

- Grupo

IdGrupo	Grupo	SQL
01	301	
02	302	
03	303	
04	304	
05	305	
06	306	

- Docentes

Colocar el nombre de todos sus docentes, pueden inventar, NoEmpleado y la profesión.
SLQ: insertar

[illegible]



• **Alumnos: NoControl, Nombre, ApellidoPaterno, ApellidoMaterno, Promedio**

[illegible]

Operaciones básicas de la tablas.	
Seleccionar todos los registros	
Tabla: Materia	Tabla: Semestre
Tabla: Grupo	Tabla: Alumno
Tabla: Docente	
Ordenar por apellido paterno	
Tabla: Materia	Tabla: Semestre
Tabla: Grupo	Tabla: Alumno
Tabla: Docente	
Contar el número de registros	
Tabla: Materia	Tabla: Semestre
Tabla: Grupo	Tabla: Alumno
Tabla: Docente	

✿ PROGRAMACIÓN ORIENTADA A OBJETOS (POO)

RECUERDAS QUE POO es como usar MOLDES para crear cosas

1. CLASE → EI MOLDE 🍪

```
python
# Molde para crear Estudiantes
class Estudiante:
    def __init__(self, nombre, edad, grado):
        self.nombre = nombre
        self.edad = edad
        self.grado = grado

    def estudiar(self):
        print(f'{self.nombre} está estudiando 📖')

    def hacer_tarea(self):
        print(f'{self.nombre} está haciendo tarea 🍷')
```

2. OBJETO → La Galleta hecha con el molde 🍪

```
python
# Creando estudiantes (objetos) usando el molde (clase)
estudiante1 = Estudiante("Ana", 15, "9no")
estudiante2 = Estudiante("Luis", 14, "8vo")

# Usando los objetos
estudiante1.estudiar() # Output: "Ana está estudiando 📖"
estudiante2.hacer_tarea() # Output: "Luis está haciendo tarea 🍷"
```

3. HERENCIA → Moldes que se parecen a otros moldes 🧑 🧒

```
python
# Molde base: Persona
class Persona:
    def __init__(self, nombre, edad):
        self.nombre = nombre
        self.edad = edad

    def hablar(self):
        print(f'{self.nombre} está hablando 🗣️')

# Molde que hereda de Persona: Estudiante
class Estudiante(Persona): # 👉 ¡HERENCIA!
    def __init__(self, nombre, edad, grado):
        super().__init__(nombre, edad) # Usa el molde del papá
        self.grado = grado

    def estudiar(self):
        print(f'{self.nombre} está estudiando en {self.grado} grado 📖')
```

Crear objeto

```
ana = Estudiante("Ana", 15, "9no")
ana.hablar() # 🗣️ (heredado de Persona)
ana.estudiar() # 📖 (propio de Estudiante)
```

ANALOGÍA DE EJEMPLO CON VIDEOJUEGOS

Base de Datos = Inventario del Jugador

SQL	
TABLA: armas INSERT INTO armas (nombre, daño, rareza) VALUES ('Espada Legendaria', 100, 'Épica');	TABLA: jugadores INSERT INTO jugadores (nombre, nivel, arma_id) VALUES ('Player1', 50, 1);
POO = Sistema de Personajes en Python	
class Personaje: def __init__(self, nombre, salud): self.nombre = nombre self.salud = salud class Guerrero(Personaje): def atacar(self): print (f"{self.nombre} ataca con espada! 🗡️")	class Mago(Personaje): def lanzar_hechizo(self): print (f"{self.nombre} lanza bola de fuego! 🔥")
¿CÓMO SE CONECTA TODO ESTO? EL PROBLEMA ANTIGUO: # ¡Doble trabajo! 😞	
# Los programadores tenían que hacer esto: estudiante = Estudiante("Maria", 15, "9no")	# Y luego esto por separado: sql = "INSERT INTO estudiantes (nombre, edad, grado) VALUES ('Maria', 15, '9no')"
LA SOLUCIÓN MODERNA (ORM):	
# Con ORM solo haces esto: estudiante = Estudiante(nombre="Maria", edad=15, grado="9no") estudiante.save() # ¡Y se guarda automáticamente en la BD!	

RESUMEN

Concepto	Es como...	Para qué sirve
Base de Datos	Una biblioteca organizada 📚	Guardar información permanentemente
SQL	Órdenes a un asistente 🤖	Comunicarse con la base de datos
POO	Usar moldes para crear cosas 🧱	Organizar el código de forma lógica
ORM	Un traductor automático 🌐	Conectar POO con bases de datos

¡Ahora entiendes por qué el ORM es tan útil!

Representa la conexión entre los dos mundos que antes estaban separados.





Definición y ventajas

ORM

Tipos de campos

Modelos y sus relaciones

Migraciones y sintaxis

Lectura 1. ¿Qué es el diseño UI/UX?

Imagina esto: llegas a un sitio web para hacer un trámite rápido, pero es un caos. No encuentras lo que buscas, los botones no funcionan como esperas y te sientes frustrado. ¿Qué haces? Lo más probable es que lo abandones en segundos. Ahora, piensa en tu aplicación favorita. Es fácil de usar, se ve genial y logra exactamente lo que necesitas sin esfuerzo. **Esa es la magia de un buen diseño de UI y UX.**

En el mundo digital de hoy, **la experiencia del usuario es el producto.** No importa si tu código es perfecto; si la interfaz es confusa, los usuarios se irán. Por eso, las habilidades de diseño UI/UX son de las más buscadas en la industria tecnológica.



Primero, aclaremos los Conceptos: ¿UI o UX?

A menudo se usan juntas, pero significan cosas diferentes. Piénsalo así:

UI (Interfaz de Usuario)	VS	UX (Experiencia de Usuario)
Es el "Cómo se Ve" 🎨		Es el "Cómo se Siente" 😊
Los botones, colores, iconos, tipografía.		La facilidad para completar una tarea, la satisfacción al usarlo.
Es la capa superficial, la estética.		Es el viaje completo, la funcionalidad y emoción.
Ejemplo: <i>Un volante deportivo y elegante.</i>		Ejemplo: <i>Lo suave y responsive que es manejar ese auto.</i>

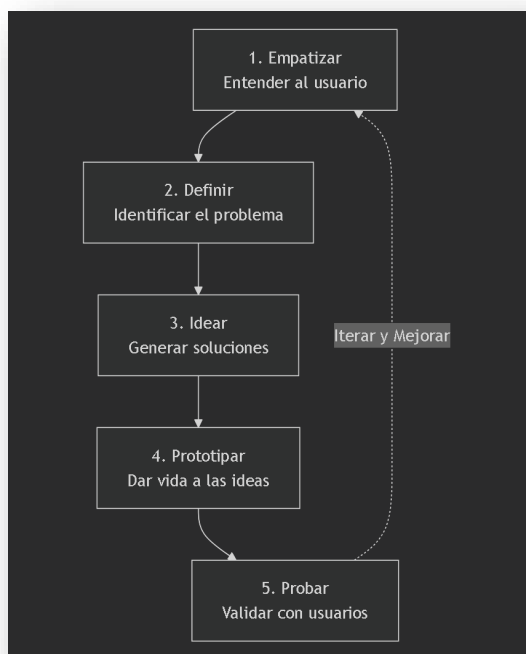
¿Cuál es más importante? _____

Ambas son cruciales. La UI atrae, la UX retiene.



El Superpoder del Diseñador: El Pensamiento de Diseño

¿Cómo se crean productos que la gente ame? No es magia, es un proceso llamado **Pensamiento de Diseño.** Es un marco de 5 pasos que te mantiene centrado en el usuario:



1. Empatizar: Sal y habla con tus usuarios. ¿Qué necesitan? ¿Qué les frustra? Olvida tus suposiciones

2. Definir: Sintetiza tu investigación. ¿Cuál es el problema central que estás resolviendo? *Ejemplo:* "Las personas mayores necesitan zapatos que les brinden alivio y comodidad."

3. Idear: ¡Libera tu creatividad! Aquí todas las ideas son válidas. *Brainstorming*, sesiones de sketch, lo que sea.

4. Prototipar: Construye una versión barata y rápida de tu idea. Puede ser un dibujo en una servilleta o un modelo digital en *Figma*.

5. Probar: Dale tu prototipo a usuarios reales y observa cómo lo usan. Su *feedback* es oro puro. Luego, itera y vuelve al paso que necesites.

De la Idea a la Realidad: Wireframes y Prototipos

- **Wireframes:** Son como los **planos de un arquitecto**. Son esquemas simples en blanco y negro que definen la estructura y la jerarquía de la información. **¿El objetivo?** Organizar el contenido sin distracciones visuales. En el diseño web, un wireframe o un diagrama wireframe es una representación visual en escala de grises de la estructura y funcionalidad de una sola página web o pantalla de aplicación móvil.
- Los wireframes se usan en las primeras etapas del proceso de desarrollo con el fin de establecer la estructura básica de una página antes de agregar el contenido y el diseño visual, y se puede crear con papel, directamente en HTML/CSS o con aplicaciones de software.

Diseño de wireframes, interfaz de usuario (UI) y experiencia de usuario (UX)

Hay cuatro elementos fundamentales, según la [arquitectura de la Información \(IA\) para el World Wide Web](#):

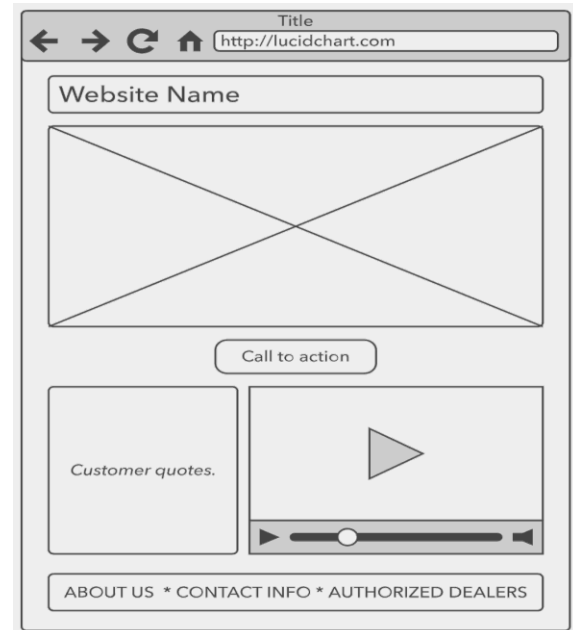
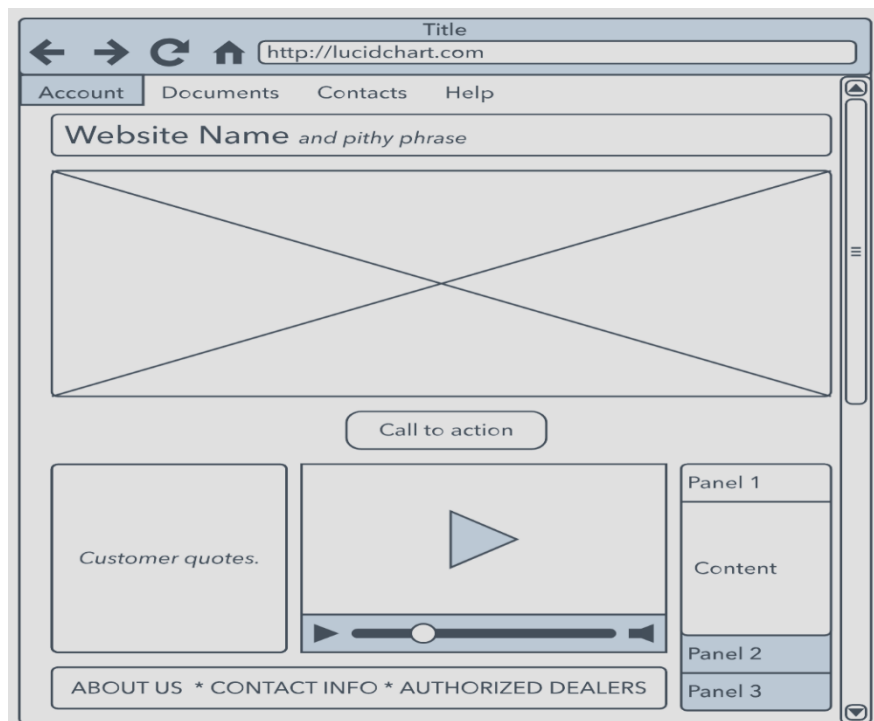
- **Estructuras y esquemas organizativos:** clasificación y estructuración de la información
- **Sistemas de etiquetado:** representación de la información
- **Sistemas de navegación:** desplazamiento por la información
- **Sistemas de búsqueda:** buscar y encontrar información

Cómo diseñar y crear un wireframe simple

1. Piensa en el objetivo principal de la página web y realiza el diseño con eso en mente. Ten en cuenta cómo interactuará el usuario con la interfaz.

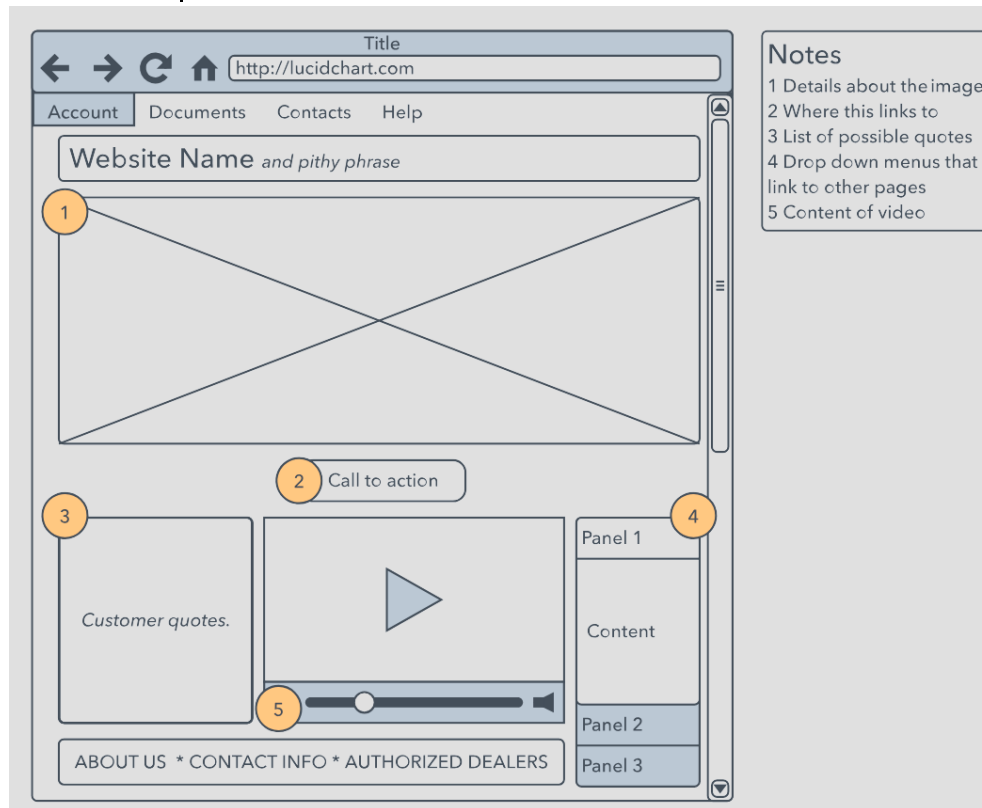
2. Comienza con una representación simple y de baja fidelidad de una página web dividida en tres partes: el encabezado (lo primero que los usuarios verán en la parte superior de la página), el cuerpo del sitio y el pie de página, que generalmente contiene información menos importante.

3. Luego, piensa en la navegación. Agrega botones y enlaces para que los usuarios visiten las principales secciones de tu sitio web, incluidas las áreas de contenido, búsqueda e inicio de sesión del usuario.



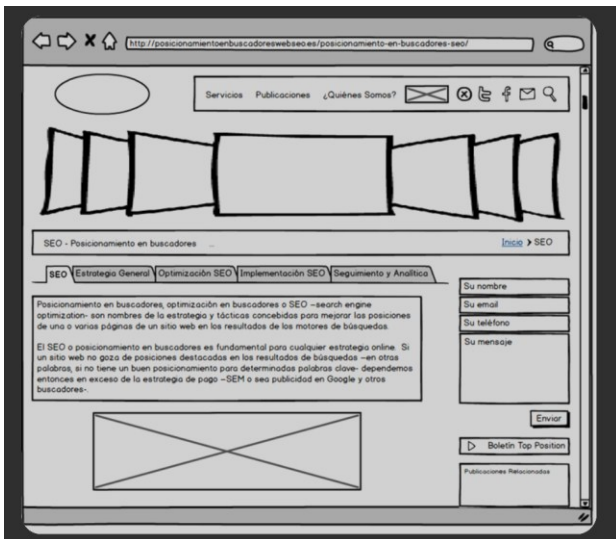
para que sea más claro.

4. Realiza anotaciones en tu wireframe



Una vez que llegues a esta etapa, podrás compartir el wireframe con los interesados antes de continuar con el siguiente paso. Puedes dibujar el wireframe inicial a mano en un papel o una pizarra, pero cuando llegues a la fase 4, es probable que necesites usar un software para seguir desarrollando el wireframe.

- **Prototipos:** Son la **maqueta interactiva** de tu producto. Los usuarios pueden hacer clic en los botones, navegar por los menús y simular la experiencia real. **¿El objetivo?** Probar el flujo y la funcionalidad antes del desarrollo. **Beneficio clave:** Encontrar y solucionar problemas aquí es 100 veces más barato y rápido que hacerlo cuando el producto ya está codificado.



Principios de Diseño Visual para UI

Son directrices clave para crear interfaces efectivas y atractivas:

- **Alineación:** Organiza los elementos para crear orden y estructura.
- **Jerarquía:** Guía la atención del usuario hacia la información más importante mediante tamaño, color y ubicación.
- **Contraste:** Diferencia elementos para crear puntos focales y mejorar la legibilidad.
- **Repetición:** Crea consistencia y coherencia visual usando elementos recurrentes.
- **Proximidad:** Agrupa elementos relacionados para organizar la información.
- **Balance:** Distribuye los elementos para lograr estabilidad visual (simétrico, asimétrico, radial).
- **Espacio (Blanco/Negativo):** El espacio vacío que mejora la legibilidad, organiza el contenido y guía la vista del usuario.

6. Elementos y Aspectos Esenciales del Diseño UI

- **Elementos Básicos:** Líneas, formas, texturas y formas (volumen) son los bloques de construcción del diseño visual.
- **Componentes de la UI:** Imágenes, textos, botones, encabezados, pies de página y layouts.
- **Tipografía y Legibilidad:** Comunicación visual mediante fuentes. El uso de serif (para titulares) y sans-serif (para cuerpo de texto) mejora la legibilidad. Se compone de alineación, proximidad, medida, interlineado y peso.
- **Teoría del Color:** Sistema basado en tono (color), valor (luminosidad) y croma (pureza). Dirige la atención y transmite emociones.

7. Herramientas de Diseño: Figma

- **Tipos de Herramientas:** Edición de raster (Photoshop, GIMP), edición vectorial (Illustrator, Inkscape) y diseño de UI/Prototipos (Sketch, Adobe XD, **Figma**).
- **Figma:** Herramienta basada en la nube, colaborativa en tiempo real, con funciones robustas de diseño vectorial, prototipado interactivo, bibliotecas de componentes y diseño responsivo. Es la opción preferida por su accesibilidad y facilidad de uso.

8. Mejores Prácticas para UI Web y Móvil

- **Cualidades de un buen diseño UI:** Sencillez, uso adecuado del color y organización clara de la información por importancia.
- **Layouts (Rejillas):** Usar rejillas (columnas, modulares, jerárquicas) para crear diseños estructurados y familiares.
- **Navegación:** Debe ser intuitiva. Incluye navegación global (encabezados) y local (barra lateral). Usar migas de pan y filtros.
- **Barra Lateral:** Debe ser clara, con contraste y evitar duplicar el menú horizontal. Usar iconos con etiquetas.
- **Encabezados (Headers):** Mejor si son "fijos" (sticky) para un acceso constante a la navegación. Deben ser compactos y con buen contraste.
- **Pies de Página (Footers):** Incluir información de contacto, enlaces legales y recordatorios de las ofertas de la empresa.

Resumen de la lectura:

EXPOSICIONES EN EQUIPO

Instrucciones generales:

Organizados en **6 equipos**, los estudiantes prepararán una exposición sobre el tema asignado, relacionado con el diseño de interfaces de usuario para aplicaciones móviles.

Distribución de temas

Equipo 1. Componentes básicos de una interfaz de usuario

- Header (encabezado).
- Body (cuerpo).
- Footer (pie de página).
- Contenedores y organización del contenido.
- Elementos interactivos: botones, campos de texto y formularios.

Equipo 2. Patrones de navegación en aplicaciones móviles

- Stack: navegación mediante pantallas apiladas.
- Tabs: navegación mediante pestañas.
- Drawer: menú lateral desplegable.
- Ejemplos visuales de cada patrón.

Equipo 3. Análisis de interfaces de aplicaciones populares

- Estructura visual de Instagram.
- Estructura visual de WhatsApp.
- Identificación de encabezados, menús, botones y áreas de contenido.
- Comparación de los elementos de navegación de ambas aplicaciones.

Equipo 4. Principios del diseño visual

- Jerarquía visual: tamaño, color y contraste.
- Consistencia: tipografía, colores y estilos uniformes.
- Feedback visual: cambios que indican una acción del usuario.
- Ejemplos de interfaces con buenas prácticas de diseño.

Equipo 5. Accesibilidad en interfaces de usuario

- Tamaño adecuado de textos y botones.
- Contraste de colores para facilitar la lectura.
- Legibilidad y organización de la información.
- Diseño inclusivo para diferentes usuarios.

Equipo 6. Representaciones estáticas de interfaces

- Wireframe: diseño de baja fidelidad.
- Mockup: diseño de alta fidelidad.
- Características y diferencias entre ambos.
- Ejemplos visuales de un wireframe y un mockup de una misma aplicación.

Productos que deberá entregar cada equipo

1. **Una infografía educativa** sobre el tema asignado, con conceptos, características, ejemplos e imágenes.
2. **Un tríptico informativo** que sintetice los conceptos más importantes de la exposición.
3. **Una actividad lúdica** para reforzar los aprendizajes de sus compañeros (Kahoot, memorama, crucigrama, cuestionario interactivo, relación de columnas, entre otras).

Desarrollo de las exposiciones

Cada equipo realizará su presentación utilizando la infografía como recurso visual de apoyo.

Al finalizar, deberá aplicar una actividad lúdica en la que participen sus compañeros, con el propósito de comprobar y reforzar la comprensión de los conceptos expuestos.

Importante: Todos los integrantes deberán participar tanto en la preparación de los materiales como en la exposición y la aplicación de la actividad lúdica.