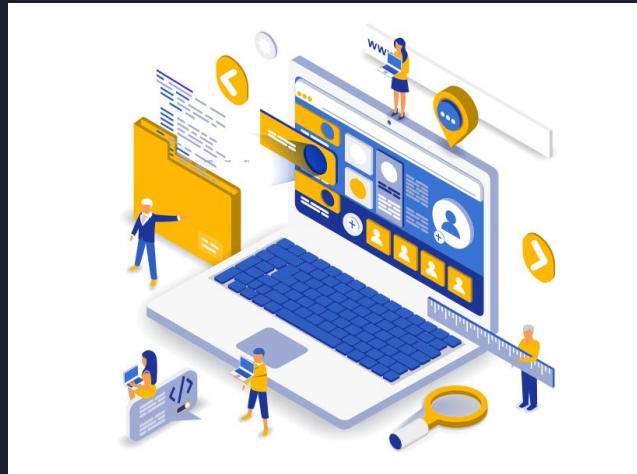



A decorative graphic on the left side of the slide. It consists of a blue parallelogram and a light green parallelogram, both tilted at an angle. The blue shape is in the foreground, and the green shape is partially behind it. They are set against a dark blue background with faint, lighter blue diagonal stripes.

# Diseño del Sistema y Arquitectura

# Introducción al Diseño del Sistema





El diseño del sistema es una etapa clave en el desarrollo de software, donde se define la estructura y los componentes del sistema para garantizar su funcionalidad, escalabilidad y mantenibilidad. Un buen diseño permite reducir costos de desarrollo y facilitar futuras modificaciones.

El diseño del sistema se compone de varios aspectos fundamentales:

- **Base de datos:** Cómo se almacenan y gestionan los datos.
- **Interfaz de usuario:** Cómo los usuarios interactúan con el sistema.
- **Lógica de negocio:** Cómo se implementan las reglas y procesos del negocio



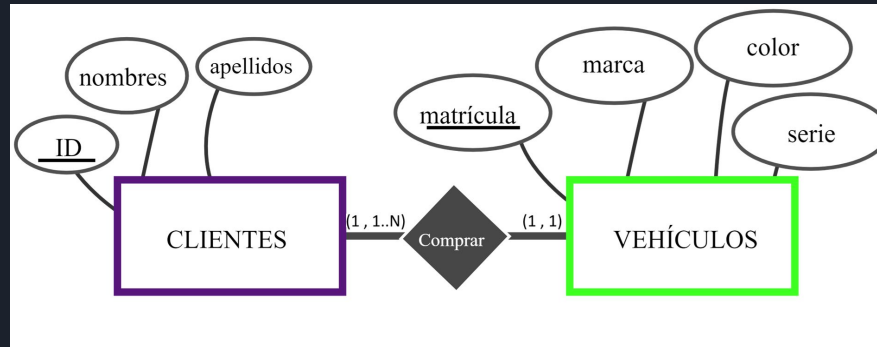
# Componentes Principales del Diseño del Sistema

## Base de Datos

La base de datos es el núcleo del sistema, ya que almacena toda la información necesaria para su funcionamiento.

### Modelado de datos

Antes de implementar una base de datos, se realiza un **modelado de datos**, que define las entidades y sus relaciones. Un enfoque común es el **Modelo Entidad-Relación (E-R)**, que se representa mediante diagramas.



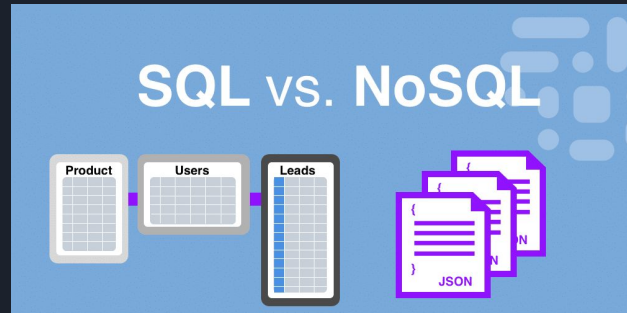
## Normalización

Es una técnica utilizada para reducir la redundancia y mejorar la integridad de los datos. Se basa en dividir grandes tablas en tablas más pequeñas y establecer relaciones entre ellas.

- **Primera forma normal (1FN)**
- **Segunda forma normal (2FN)**
- **Tercera forma normal (3FN)**

## Tipos de bases de datos

- **Relacionales (SQL):** Organizadas en tablas con claves primarias y foráneas (ej: MySQL, PostgreSQL).
- **NoSQL:** Diseñadas para escalabilidad y flexibilidad, como bases de datos de documentos o grafos (ej: MongoDB, Firebase).





## Interfaz de Usuario (UI)

La interfaz de usuario es la parte visible del sistema con la que interactúan los usuarios. Un diseño bien estructurado mejora la experiencia del usuario (UX).

### Principios de diseño de interfaces

- **Simplicidad:** Diseños limpios y fáciles de entender.
- **Consistencia:** Mantener elementos visuales y patrones similares en toda la aplicación.
- **Feedback inmediato:** Informar al usuario cuando una acción se ha realizado correctamente o ha fallado.
- **Accesibilidad:** Asegurar que la interfaz sea usable para todos los usuarios, incluidos aquellos con discapacidades.

### Wireframes y prototipos

Antes de desarrollar la interfaz, es común crear **wireframes** (bocetos de la estructura de la UI) y **prototipos interactivos** para evaluar la experiencia de usuario.

### Diseño responsivo

El **diseño responsivo** asegura que la interfaz de usuario se adapte a diferentes dispositivos y tamaños de pantalla. Esto es crucial en aplicaciones web y móviles.







## Lógica de Negocio

La lógica de negocio define cómo el sistema procesa los datos y toma decisiones basadas en reglas establecidas.

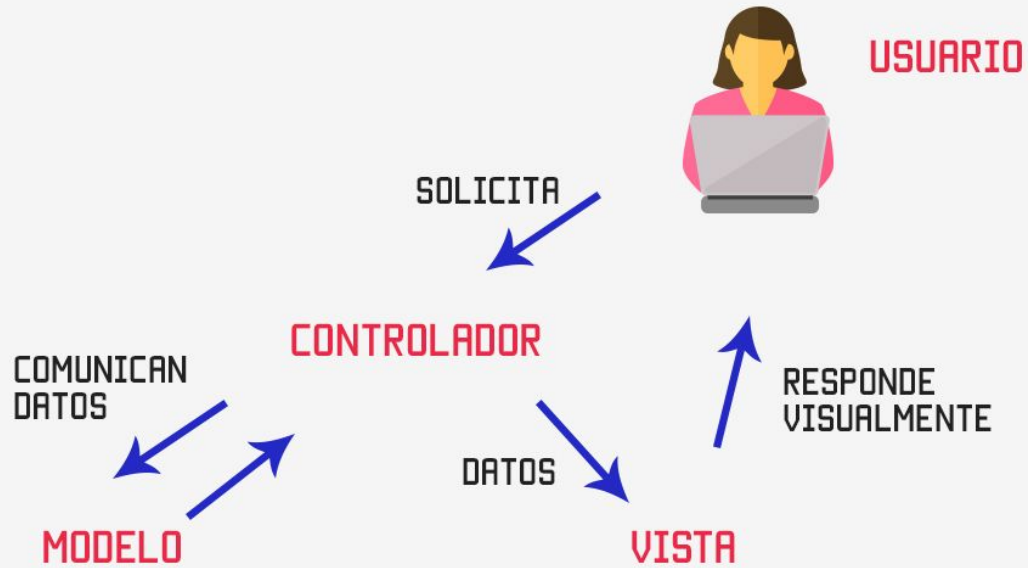
### Patrones de diseño en la lógica de negocio

Para organizar el código de manera eficiente, se utilizan patrones de diseño como:

- **MVC (Modelo-Vista-Controlador):** Divide la aplicación en tres capas:
  - **Modelo:** Maneja los datos y la lógica de negocio.
  - **Vista:** Muestra la interfaz de usuario.
  - **Controlador:** Maneja las interacciones del usuario y actualiza el modelo y la vista.
- **MVVM (Modelo-Vista-ViewModel):** Similar a MVC, pero con mayor separación entre la lógica y la interfaz.


### Buenas prácticas en el diseño de la lógica del negocio

- Separación de preocupaciones
- Reutilización de código
- Pruebas unitarias





# Introducción a la Arquitectura de Software



La arquitectura del software define la estructura general del sistema, incluyendo la organización de sus componentes y la forma en que interactúan entre sí.

## Modelos arquitectónicos comunes

Existen diferentes modelos arquitectónicos, cada uno con ventajas y desventajas:

### Arquitectura Monolítica

- Toda la aplicación se desarrolla como un solo bloque de código.
- Ventajas: Sencilla de desarrollar y desplegar.
- Desventajas: Difícil de escalar y mantener a largo plazo.

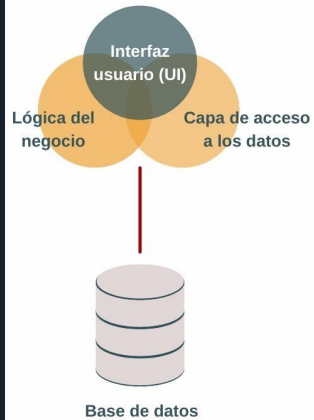
### Arquitectura de Microservicios

- La aplicación se divide en servicios independientes que se comunican entre sí.
- Ventajas: Escalabilidad, facilidad para actualizar partes del sistema sin afectar al resto.
- Desventajas: Mayor complejidad en la gestión de servicios y comunicación entre ellos.

### Patrón Cliente-Servidor

- Separación entre un **cliente** (interfaz de usuario) y un **servidor** (lógica de negocio y base de datos).
- Es el modelo común en aplicaciones web y móviles.

## Arquitectura monolítica



## Arquitectura microservicios

