

MVC – Explicación Completa con Ejemplos (Markdown)

MVC: Model - View - Controller

MVC es un patrón arquitectónico que separa una aplicación en **tres capas** para mejorar la organización, mantenibilidad y escalabilidad del sistema.

1 MODEL (El cerebro - Datos y Lógica de Negocio)

El **Model** contiene:

- Reglas de negocio
- Entidades (clases)
- Validaciones
- Acceso a datos (Firestore, MySQL, APIs, archivos)

Ejemplos reales:

- Clase `Usuario`, `Pelicula`, `Producto`
- Conexión a la BD
- Validación: "el stock no puede ser negativo"

Ejemplo en tu app WatchMe:

```
```java
public class Pelicula {
 String id;
 String titulo;
 String poster;
}
```

---

## 2 VIEW (La interfaz – Lo que ve el usuario)

---

La **View** muestra la información al usuario.

NO contiene lógica de negocio.

Puede ser:

- XML (Android)
- HTML / Blade / JSP
- Plantillas

- Interfaces gráficas

Ejemplo en tu app:

```
<TextView android:id="@+id/tvTitulo" />
<ImageView android:id="@+id/imgPoster" />
```

---

## 3 CONTROLLER (El intermediario – Controla el flujo)

---

El **Controller** recibe las acciones del usuario:

- llama al model
- procesa la respuesta
- decide qué vista mostrar

Ejemplo en WatchMe (Android):

```
public class DetallePeliculaActivity extends AppCompatActivity {

 @Override
 protected void onCreate(Bundle b) {
 super.onCreate(b);
 setContentView(R.layout.activity_detalle);

 tvTitulo.setText(pelicula.getTitulo());
 Glide.with(this).load(pelicula.getPoster()).into(imgPoster);
 }
}
```

---

## EJEMPLO REAL N°1: E-commerce (Pregunta clásica de examen)

---

### MODEL

- Producto
- Carrito
- Pedido
- BD MySQL/Firestore

### VIEW

- HTML/plantilla del carrito
- Detalle de producto
- Confirmación de compra

## CONTROLLER

- `/productos/listar`
- `/carrito/agregar`
- `/checkout/pagar`



## EJEMPLO REAL N°2: Instagram usa MVC

---

### MODEL

- Usuario
- Publicación
- Story

### VIEW

- Pantalla principal
- Perfil
- Stories

### CONTROLLER

- LikeController
- PostController
- LoginController



## EJEMPLO REAL N°3: API con MVC (PHP / Node / Java)

---

### Controller:

- `GET /api/clientes` → ClienteController
- `POST /api/pedidos` → PedidoController

### Model:

- `ClienteModel`
- `PedidoModel`

### View:

- JSON de respuesta



## BENEFICIOS DE MVC

---

- Separación de responsabilidades
- Código más ordenado
- Facilita testeo
- Facilita mantenimiento
- Puedes cambiar la vista sin tocar la lógica



## RESUMEN SUPER CORTO PARA ESTUDIAR

---

**MODEL = datos + lógica**

**VIEW = pantalla**

**CONTROLLER = coordina y ejecuta**

El usuario interactúa → Controller → Model → Vista → usuario.



## FIN DEL DOCUMENTO

---

Si quieres, te genero también:

- ☒ **Modelo de Capas en .md**
- ☒ **SOA en .md con ejemplos reales**
- ☒ **Diagramas de despliegue en .md listos para la prueba**

¿Quieres que siga con **Modelo de Capas**?