Practica 3. Diseño arquitectonico de una Aplicación

Marco Hernandez, David Sosa, Ana Jimenez

1 Definición del Dominio

1.1 Elegir un dominio o tema para la aplicación móvil

Hemos elegido nuestro propio trabajo de la asignatura que es la app PioChef, una red social donde los usuarios pueden compartir y descubrir recetas de cocina.

1.2 Definir las principales funcionalidades

- **CRUD recetas**:
  - Los usuarios pueden crear sus propias recetas con ingredientes, instrucciones, fotos y tiempos de elaboración.
  - Los usuarios pueden explorar, buscar y guardar recetas de otros usuarios.
  - Los usuarios pueden editar o eliminar sus propias recetas.
- **CRUD amigos**:
  - Los usuarios pueden enviar y aceptar solicitudes de amistad.
  - Los usuarios pueden seguir o dejar de seguir a otros chefs.
  - Los usuarios podrán ver el perfil y las recetas de amigos.
- **Visualización de recetas**:
  - Feed personalizado donde los usuarios pueden ver, darle a "like" y comentar las recetas que se han compartido.
- **Descubrir recetas**:
  - Sección en la cual los usuarios podrán explorar recetas mediante la búsqueda de palabras clave.
  - Se podrá filtrar según su tiempo de elaboración, categoría o cantidad de "likes".

2 Identificación de los principales elementos de la arquitectura elegida

El patrón de diseño elegido es el de Modelo-Vista-Presentador que deriva del patrón arquitectónico Modelo-Vista-Controlador. Este patrón está diseñado para estructurar la aplicación en tres capas lógicas interconectadas que en nuestro caso serán las siguientes:

1. Modelo: podría usarse Firebase para almacenar y gestionar los datos de las recetas. También define clases o estructuras para representar las recetas o usuarios por ejemplo.

2. Vista: aquí es donde está la interfaz de usuario gráfica que es la encargada de gestionar las interacciones del usuario así como define las páginas, textos y componentes que conformarán nuestra interfaz de usuario.

3. Presentador: son los paquetes de clase que se encargan de la gestión de la lógica de negocio, la coordinación entre modelo y vista y el manejo de errores.

3 Diseño de la Arquitectura

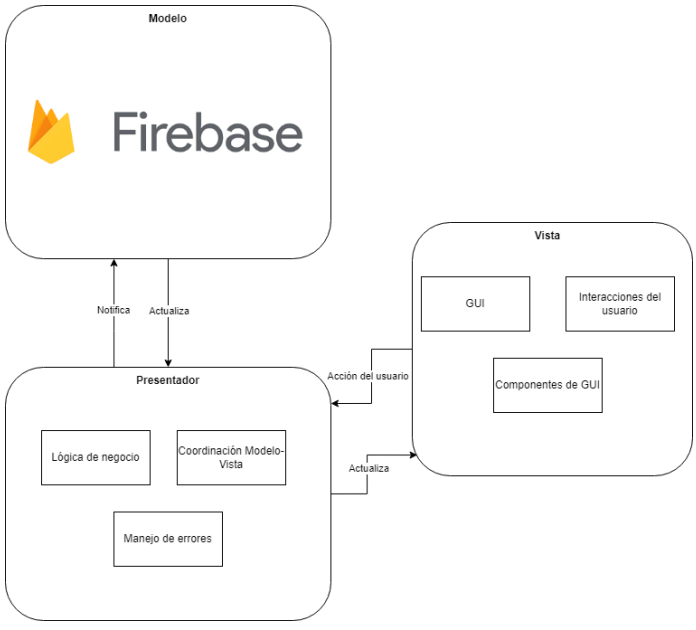

Figure 1: Diagrama que representa el Modelo-Vista-Presentador

4 Casos de Uso

Historia: Añadir receta	**ID: 1**
Como usuario registrado **quiero** añadir mis propias recetas **para poder** compartirlas y tener un recetario en linea	
Validación: - El usuario tiene que estar registrado - Se debe completar el formulario de creación de receta: título, descripción, un formulario de pasos e ingredientes - Se pueden añadir pasos y ingredientes como el usuario desee - La receta se añade a la base de datos, será visible para todos si el usuario lo especifica - Habrá una ventana de confirmación una vez finalizado el formulario y la receta añadida a la base de datos	**Valor: 100** **Prioridad: Alta** **Estimación: 5h**

Figure 2: Añadir receta

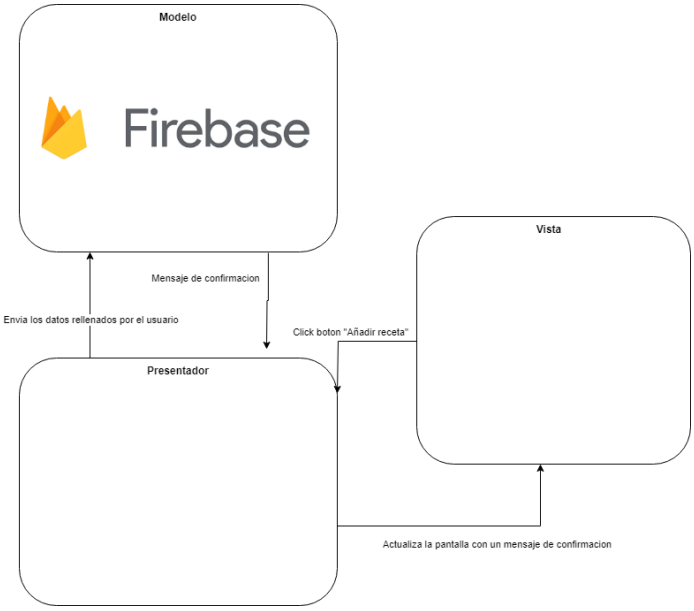

Figure 3: Diagrama que seguira el caso de uso "Añadir receta"

Historia: Descubrir recetas	**ID: 2**
Como usuario registrado **quiero** ver nuevas recetas **para poder** visualizarlas y guardarlas en mi recetario	
Validación: - El usuario tiene que estar registrado - Habrá una ventana que muestre la imagen y título de la receta. - Al hacer clic, se podrá acceder a información de la receta	**Valor: 90** **Prioridad: Baja** **Estimación: 4h**

Figure 4: Descubrir receta

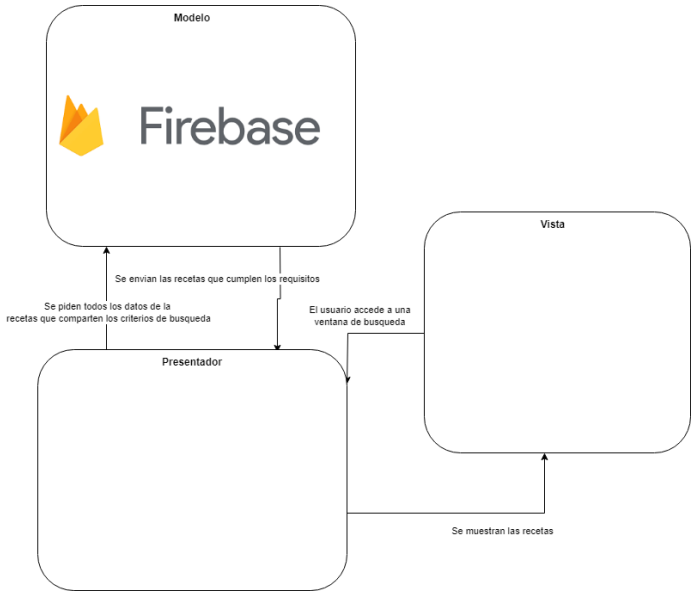

Figure 5: Diagrama que seguira el caso de uso "Descubrir recetas"

5 Conclusión

MVP usa un presentador, responsable de manejar las acciones de usuario y actualizar la vista creando independencia del modelo, por lo que se facilita la reutilización de código y creación de pruebas unitarias, si lo comparamos con MVC, este tiene un controlador que es menos efectivo que un presentador porque es responsable de demasiadas cosas, impidiendo un código limpio y organizado

además de poder acoplarse con la vista y modelo dificultando la reutilización de código.

Siguiendo con las ventajas de MVP, esta separa lógica de negocio con lógica de presentación permitiendo un codigo facil de mantener y modificar.