PROGRAMACIÓN DE APLICACIONES MÓVILES NATIVAS

Semana 4: Elección de una arquitectura

Realizado por: Daniel Betancor Zamora

Curso 2023-2024

Contenido

1. Aplicación de E-Commerce para una PYME

A continuación se plantea el primer supuesto, pudiendo visualizar las características principales del proyecto como: el presupuesto, los tiempos de entrega, los recursos humanos y el rendimiento. A partir de estos datos de interés, se propondrá la arquitectura que se considera idónea para llevar a cabo esta aplicación, y se justificará el por qué.

Supuesto 1: *Aplicación de E-commerce para una PYME*

Una pequeña empresa quiere lanzar su tienda online a través de una aplicación móvil nativa.

Presupuesto: Limitado.

Tiempos de entrega: 4 meses.

Recursos humanos: Un desarrollador principal y un diseñador.

Rendimiento: Se espera un tráfico moderado, pero es esencial que la aplicación sea rápida y eficiente.

En este primer caso, se ha propuesto la arquitectura **MVP** para el desarrollo de la aplicación.

En una aplicación de comercio electrónico, la lógica de negocio es crucial, así que gracias al **Modelo** de MVP toda esa lógica puede ser gestionada y representada, como la gestión de productos, carritos de compras y procesamiento de órdenes. Separar esta lógica del resto de la aplicación es fundamental para mantenerla organizada y fácil de mantener. Además, la interfaz de usuario de una aplicación de comercio electrónico es lo que los usuarios ven y con lo que interactúan, por lo que, la **Vista** permite una personalización más sencilla de la interfaz de usuario y la gestión de eventos. Por último, el **Presentador** es el elemento de la arquitectura MVP responsable de comunicarse con el Modelo y actualizar la Vista en consecuencia, por lo que es la pieza intermediaria y clave en todo este proceso.

Por otra parte, en un entorno de comercio electrónico, la calidad y la fiabilidad son esenciales. MVP permite la creación de pruebas unitarias eficientes, lo que significa que se puede probar fácilmente cada componente por separado: el Modelo, la Vista y el Presentador. Esto puede llegar a ser bastante útil para identificar y solucionar errores de manera efectiva.

En resumen, la elección de MVP para el primer supuesto se basa en la necesidad de una separación clara de responsabilidades, la colaboración eficiente entre el desarrollador y el diseñador, la capacidad de realizar pruebas unitarias y la preparación para futuras expansiones. Estos factores hacen que MVP sea una elección excelente para una aplicación de comercio electrónico con un presupuesto limitado y un equipo reducido.

2. Aplicación Social Interactiva para una Startup

De la misma forma, se expondrá el segundo supuesto y sus características principales, con el objetivo de proponer una arquitectura que pueda encajar bien en este caso.

Supuesto 2: *Aplicación Social Interactiva para una Startup*

Una startup quiere crear una aplicación social con características interactivas, como chats en tiempo real y transmisiones en vivo.

Presupuesto: Moderado.

Tiempos de entrega: 6-8 meses.

Recursos humanos: Un equipo de tres desarrolladores, un diseñador y un programador backend.

Rendimiento: Se espera un alto tráfico y es crucial que la aplicación maneje interacciones en tiempo real.

Para este proyecto se ha creído conveniente aplicar la arquitectura **MVVM**.

En una aplicación social interactiva con chats en tiempo real y transmisiones en vivo, la gestión eficiente de eventos y actualizaciones en tiempo real es esencial. Ese es uno de los principales motivos por los que se ha optado por esta arquitectura, ya que MVVM permite una vinculación de datos eficaz, lo que facilita la actualización automática de la interfaz de usuario cuando ocurren eventos en tiempo real.

Otro de los aspectos que se han tenido en cuenta, a la hora de seleccionar la arquitectura, han sido los recursos humanos. Vemos que al ser una *Startup*, se dispone de un equipo multidisciplinar compuesto por varios desarrolladores, un diseñador y un programador de backend. En este tipo de casos, MVVM permite una separación clara de las responsabilidades, lo que facilita la colaboración entre los miembros.

Por último, pero no menos importante, si se espera un alto tráfico y la aplicación tiene el potencial de crecer en el futuro, MVVM proporciona una base escalable y mantenible. Además, la separación de capas y la organización de la lógica en el *ViewModel* hacen que sea más fácil agregar nuevas características y funcionalidades sin afectar el funcionamiento existente.

Concluyendo, la elección de MVVM para este proyecto es clave, debido a su capacidad para gestionar eficientemente las interacciones en tiempo real, su separación de responsabilidades clara, su vinculación de datos y reactividad, y su escalabilidad y mantenibilidad. Esta arquitectura es especialmente adecuada para aplicaciones sociales interactivas que requieren una experiencia de usuario fluida y en tiempo real.

3. Aplicación Financiera para una Gran Empresa

Como en los anteriores supuestos, se vuelven a presentar las características claves del proyecto que nos guiarán para seleccionar la arquitectura adecuada.

Supuesto 3: *Aplicación Financiera para una Gran Empresa*

Una gran empresa financiera quiere desarrollar una aplicación para que sus clientes gestionen sus finanzas, con características como visualización de transacciones, transferencias y análisis financiero.

Presupuesto: Alto.

Tiempos de entrega: 10-12 meses.

Recursos humanos: Un equipo grande con múltiples desarrolladores, diseñadores, especialistas en seguridad y analistas.

Rendimiento: Se espera un tráfico muy alto y es esencial que la aplicación sea segura y eficiente.

En una gran empresa financiera, los requisitos empresariales pueden ser altamente complejos y cambiantes. Por eso mismo, la **Clean Architecture** puede ser la arquitectura perfecta para este tipo de proyectos, ya que proporciona una estructura flexible que puede adaptarse a los requisitos cambiantes y acomodar la complejidad de la lógica financiera

Además, dado que se espera un alto tráfico y la aplicación debe gestionar la visualización de transacciones, transferencias y análisis financiero, el empleo de Clean Architecture en un proyecto de esta magnitud es fundamental por su gran eficiencia.

Cabe destacar que esta arquitectura ofrece una separación clara de responsabilidades, lo que facilita la escalabilidad y el mantenimiento a largo plazo, lo cual será imprescindible en una aplicación de este tipo con una gran proyección futura. En concreto, la capa de dominio, permite que la lógica empresarial sea independiente de los detalles de la implementación, lo que simplifica futuras expansiones.

Otro de los aspectos a tener en cuenta es que, en el sector financiero, la seguridad y la privacidad de los datos son factores críticos. En este sentido, Clean Architecture facilita la implementación de medidas de seguridad sólidas al permitir una capa de seguridad dedicada. Además, el aislamiento de la capa de datos garantiza que los datos sensibles se manejen de manera segura.

Por último, es obvio que las grandes empresas financieras tienen una gran variedad de sistemas y tecnologías en uso. Por lo que el uso de este tipo de arquitecturas permite adaptarse a diferentes tecnologías en cada capa, lo que facilita la integración con sistemas heredados y la adopción de nuevas tecnologías según sea necesario.

En resumidas cuentas, Clean Architecture es una elección sólida para este supuesto debido a la complejidad de los requisitos empresariales, la necesidad de escalabilidad y mantenimiento a largo plazo, los requisitos de seguridad y privacidad de los datos, el cumplimiento normativo, la colaboración en equipo y la capacidad de realizar pruebas exhaustivas.

4. Plataforma de Salud y Bienestar para Hospitales

En este cuarto supuesto se plantea una aplicación móvil nativa destinada a proporcionar una plataforma de salud y bienestar para los hospitales. Tal y como se puede observar, se trata de un proyecto de gran envergadura, lo que implicará el uso de una arquitectura bien estructurada, que sea fácil de escalar, y que ofrezca la seguridad necesaria para que este tipo de aplicaciones funcione correctamente.

En la siguiente página se podrán estudiar más en detalle las características básicas del proyecto, en base a las cuales se podrá decidir que arquitectura usar.

Supuesto 4: *Plataforma de Salud y Bienestar para Hospitales*

Un hospital de renombre desea desarrollar una aplicación móvil nativa que permita a los pacientes acceder a sus historiales médicos, programar citas, chatear con especialistas y recibir recomendaciones personalizadas basadas en su historial.

Presupuesto: Muy alto.

Tiempos de entrega: 12-15 meses.

Recursos humanos: Un equipo multidisciplinario compuesto por varios desarrolladores móviles, desarrolladores backend, especialistas en seguridad de la información, diseñadores UX/UI y analistas de sistemas.

Rendimiento: Se espera un tráfico constante y alto debido a la gran cantidad de pacientes. La seguridad y privacidad de los datos es primordial.

Para este supuesto se ha creído idóneo el uso de la arquitectura **Hexagonal**.

La arquitectura hexagonal se basa en la idea fundamental de separar las preocupaciones (separación de responsabilidades). En el contexto de la atención médica y la gestión de datos de pacientes, esta separación es esencial, ya que puede haber múltiples sistemas, dispositivos y servicios que interactúan con la aplicación de salud y, en ese sentido, la arquitectura hexagonal ayuda a aislar y a administrar esas interacciones de manera clara y organizada.

En segundo lugar, en una plataforma de salud, es común que la aplicación deba interactuar con una gran variedad de dispositivos médicos y servicios externos, como sistemas de gestión hospitalaria, dispositivos de monitoreo de pacientes, sistemas de laboratorio, etc. En este tipo de casos, la arquitectura hexagonal permite crear puertos para cada uno de estos dispositivos y servicios, lo que facilita la adaptación y la integración.

Por otra parte, y de cara al futuro, la arquitectura hexagonal promueve la mantenibilidad a largo plazo y la escalabilidad, lo que es esencial para una plataforma de salud que debe seguir siendo efectiva y actualizada durante mucho tiempo.

Otra de las motivaciones clave para implementar esta arquitectura, es que, gracias a ella, se pueden implementar medidas de seguridad robustas para garantizar la privacidad y la integridad de los datos de salud, que es una preocupación crítica en el sector. Y por supuesto, en un entorno médico, es importante minimizar el riesgo de errores, siendo ideal la arquitectura hexagonal para ello, ya que ayuda a aislar y gestionar los errores de manera eficiente, garantizando la seguridad del paciente.

En conclusión, la elección de la arquitectura hexagonal en el supuesto 4 está respaldada por su capacidad para gestionar las complejas interacciones en el sector de la salud, su adaptabilidad a dispositivos y servicios externos variados, su enfoque en la mantenibilidad a largo plazo, su capacidad para adaptarse a cambios regulatorios y su enfoque en la seguridad de los datos. Esta arquitectura es altamente adecuada para proyectos en el ámbito de la salud y el bienestar.

5. Aplicación Prototipo para un Hackathon

Por último, se muestran a continuación las características del último supuesto.

Supuesto 5: *Aplicación Prototipo para un Hackathon*

Un grupo de estudiantes decide participar en un hackathon de 48 horas. Su objetivo es crear un prototipo funcional de una aplicación móvil que ayude a las personas a encontrar compañeros de viaje para compartir gastos en carreteras de peaje.

Presupuesto: Mínimo. Los estudiantes usarán herramientas y recursos gratuitos disponibles.

Tiempos de entrega: 48-72 horas.

Recursos humanos: Un equipo de tres estudiantes con habilidades mixtas: un desarrollador, un diseñador y alguien con habilidades de negocio.

Rendimiento: Como es un prototipo, no se espera un tráfico real. La aplicación debe ser lo suficientemente funcional para demostrar la idea.

En un hackathon, el tiempo es un recurso extremadamente limitado. Por eso mismo, creo que la arquitectura **MVC** es la adecuada para llevarlo a cabo. MVC es conocida por su simplicidad y facilidad de implementación, por lo que permitirá que el equipo de estudiantes se enfoque en la creación rápida de un prototipo funcional en lugar de lidiar con una arquitectura compleja.

Es cierto que, en caso de que la aplicación móvil nativa sea desarrollada para Android, puede que esta arquitectura no sea la más adecuada, pero debido al poco tiempo disponible para desarrollar el prototipo considero que es la arquitectura más indicada.

Además, MVC proporciona una separación básica de responsabilidades, lo que permite a los estudiantes dividir las tareas de manera eficiente. Uno puede enfocarse en el Modelo (lógica de negocio), otro en la Vista (interfaz de usuario) y otro en el Controlador (gestión de eventos).

También, cabe destacar que, en un hackathon, es posible que se deban realizar pruebas rápidas y efectivas para asegurar que el prototipo funciona correctamente. En este sentido, MVC facilita la realización de pruebas unitarias en componentes individuales, lo que ayuda a identificar y corregir errores de manera eficiente.

Para concluir, considero que MVC es una elección lógica para esta actividad debido a su simplicidad, capacidad para el desarrollo rápido, comunicación eficiente entre componentes, facilidad de pruebas y aprendizaje. Estas ventajas son especialmente relevantes en el contexto de un hackathon donde el tiempo y los recursos son limitados, y el objetivo principal es crear un prototipo funcional de manera rápida y efectiva.