Fall 2024 Mobile Application Development (CPRG-303-F)

Assignment: Architectural Decisions

Name: Yaling Wei

No: 000930508

Date: Nov 8th.2024

Scenario 2: Social Networking Mobile App for a University - Architectural Decisions Record (ADR)

1.Status

Proposed

2.Context

We are developing a social networking mobile application for a university that requires:

- Cross-platform support (iOS and Android)
- Student-professor connectivity
- Class information sharing
- Event and club management
- Schedule management
- Assignment and grade posting
- Offline mode capabilities
- Active Directory integration
- Push notifications
- Accessibility features
- Data privacy and security measures

Introduction

This ADR documents the architectural decisions made by the development team responsible for creating a social networking mobile app for a university. The app aims to connect students and professors, share class information, view events and clubs, and manage schedules. This document outlines the chosen architecture, frameworks, technology stacks, and the rationale behind these decisions.

3.Architectural Decisions

3.1 Platform Support

Decision: The app will support both iOS and Android platforms.

Rationale: To ensure widespread adoption and accessibility, it is crucial that the app is available on both major mobile operating systems. This will allow the majority of students and professors to use the app regardless of their device preferences.

3.2 Framework and Technology Stack

Decision: Implement using React Native + TypeScript. The app will be developed using React Native, a framework for building cross-platform mobile apps.

Rationale:

Performance and Native Experience: React Native provides a high-performance experience very similar to native apps, which is essential for real-time features such as GPS tracking and providing directions. React Native allows for efficient development and maintenance across multiple platforms. It uses JavaScript, which is familiar to many developers, and can generate natively rendered mobile UI components. This will ensure a seamless user experience on iOS and Android.

Strong Community Support: A large and active community provides a wide range of resources, libraries, and plugins, which enables faster troubleshooting and access to third-party integrations.

Reusability: Code is reusable across platforms (estimated to be more than 90% shared code)

Maintainability: Strong typing and better maintainability using TypeScript.

3.3 UI Component Architecture

Decision: Implemented using React Native Paper + custom components.

Rationale:

· Material Design implementation for consistent UI

· Built-in accessibility support

· Extensive component library

· Performance-optimized elements

· Customizable theming system

· Strong typing support with TypeScript

· Easy integration with native modules

3.4 Backend Language & Framework

Decision: Node.js with Express.js

Rationale:

· JavaScript/TypeScript consistency across stack

· Excellent async operation handling

· Rich NPM ecosystem

· Strong performance for API operations

· Easy integration with various services

· Scalable architecture support

3.5 Data Storage Solutions

Mobile Storage

Decision: Redux Persist + AsyncStorage

Rationale:

Efficient offline data management

Secure local storage

State persistence across app launches

Performance optimized

Strong community support

Server Storage

Decision: MongoDB with Redis Cache

Rationale:

Flexible schema for varied data types

Horizontal scaling capabilities

Efficient document storage

High-performance caching

Real-time data support

3.6 Authentication and Authorization

Decision: JWT + OAuth2.0 + SSL/TLS.The app will integrate with an existing Active Directory system for secure login. Roles and permissions will be properly enforced to ensure appropriate access control.

Rationale: Securing user data and ensuring appropriate access control is crucial. Integrating with an existing Active Directory system will simplify the authentication process for students and professors, while ensuring that only authorized users can access sensitive information.

The approach has industry-standard security, covering multiple authentication methods. It has secure token management and role-based access control. It also has cross-platform compatibility.

3.7 Push Notification Service

Decision: The app will integrate with a push notification service that supports both iOS and Android platforms.

Rationale: Push notifications are an effective way to keep users informed about new announcements, assignment deadlines, and other relevant updates. By choosing a service that supports both platforms, the team can ensure that all users receive timely notifications.

3.8 Data Privacy and Security

Decision: The app will implement encryption, secure data transmission protocols, and adhere to relevant data protection regulations to prioritize data privacy and security.

Rationale: The app will handle sensitive information such as student grades and personal profiles. Implementing robust security measures is essential to protect this data and maintain user trust.

3.9 Accessibility

Decision: The app will be designed and developed with accessibility features in mind, such as text-to-speech, high contrast modes, and support for assistive technologies.

Rationale: Ensuring inclusivity is important. By implementing accessibility features, the app will be more usable for individuals with disabilities, enhancing overall user satisfaction.

3.10 API Architecture

Decision: RESTful API with GraphQL for complex queries

Rationale:

Standard REST practices for basic operations

GraphQL for optimized data fetching

Reduced network overhead

Strong typing support

Easy version management

4. Additional Frameworks and Technology Stacks

The app will use the following additional frameworks and technology stacks:

- **Redux** for state management
- **Axios** for making HTTP requests
- **React Navigation** for handling navigation between screens
- **Firebase** for real-time database and authentication services (if required)

Rationale: These frameworks and technology stacks will support the development of key app features such as state management, data fetching, navigation, and real-time updates. They are well-documented, widely used, and have a strong community support, making them reliable choices for this project.

5. Conclusion

This ADR documents the architectural decisions made by the development team for the social networking mobile app for a university. By carefully considering platform support, framework and technology stack, offline capabilities, performance optimization, authentication and authorization, push notification service, data privacy and security, and accessibility, the team has ensured that the app will meet the needs of its users while maintaining a high level of quality and security.