**Assignment: Architectural Decisions**

|  |  |
| --- | --- |
| Team Name: PeachParty (Group-9) | Date: 3/14/2024 |
| Team Members: Chris de Guzman, Matthew Bayona, Ki Fung Yuen, Tao Zhan |  |

**Scenario 3**

**1.**

**Title:** Native, Web, or Hybrid App

**Status:** Accepted

**Context:** The team has been tasked with developing a mobile application for ordering food from local restaurants for delivery or pickup. The app needs to provide users with a seamless and efficient ordering experience, including features such as real-time order tracking and secure payment processing.

**Decision**: Develop a native mobile application for ordering food from local restaurants for delivery or pickup.

**Rationale:** Native apps offer superior performance compared to web or hybrid apps, providing smoother user interactions and faster load times, crucial for ensuring a seamless ordering experience.

**Consequences:** Platform-specific development efforts will be required for each target platform (iOS and Android). Development time may be extended due to separate codebases for each platform. Maintenance costs may increase due to managing multiple codebases and platform-specific updates.

**2.**

**Title:** UI Framework

**Status:** Accepted

**Context:** The team needs to choose a UI framework for developing the user interface of the mobile app. This decision will impact development efficiency, cross-platform compatibility, and user experience.

**Decision:** The team has chosen React Native as the UI framework for its cross-platform compatibility, code reusability, and native-like user experience.

**Rationale:** Utilizing React Native streamlines development efforts by enabling code sharing across platforms, reducing development time and costs. Additionally, its native performance and component-based architecture ensure a consistent and responsive user experience across different devices and operating systems.

**Consequences:** Adopting React Native simplifies cross-platform development but may require additional effort to integrate platform-specific features and optimize performance. However, it facilitates code reuse and ensures a consistent user experience across platforms, enhancing development efficiency and user satisfaction.

**3.**

**Title:** Backend Language

**Status:** Accepted

**Context:** The team must select a backend language for developing the server-side logic and APIs of the food ordering application. This decision will affect scalability, performance, and ease of development.

**Decision:** The team has opted for Node.js as the backend language due to its asynchronous nature, scalability, and suitability for real-time applications.

**Rationale:** Node.js's event-driven, non-blocking I/O model enhances scalability and performance, making it ideal for handling real-time features such as order tracking and payment processing. Its extensive ecosystem and community support also facilitate rapid development and deployment of server-side logic.

**Consequences:** Using Node.js simplifies real-time feature implementation but may require additional effort for handling CPU-intensive tasks. However, its scalability and performance benefits enable efficient handling of concurrent requests, ensuring responsiveness and reliability for real-time features.

**4.**

**Title:** Permissions

**Status:** Accepted

**Context:** The app needs to access the user's location for accurate restaurant recommendations and delivery estimates. The team must decide on the approach for obtaining location permissions.

**Decision:** The team will request location permissions from users to enable accurate restaurant recommendations and delivery estimates, ensuring a seamless user experience.

**Rationale:** Obtaining location permissions enables the app to provide personalized and location-based services, enhancing user engagement and satisfaction. By respecting user privacy and transparently communicating the purpose of location access, the app builds trust and encourages user adoption.

**Consequences:** Requesting location permissions may lead to user concerns about privacy and data usage. However, it enables the app to deliver personalized and location-based services, improving user experience and engagement. Transparent communication about the purpose of location access can mitigate user concerns and foster trust in the app.

**5.**

**Title:** Data Storage

**Status:** Accepted

**Context:** The team needs to determine the data storage solution for storing user accounts, payment information, order history, and restaurant menus.

**Decision:** The team will implement Firebase Firestore as the cloud-based database solution for secure and scalable data storage.

**Rationale:** Firebase Firestore offers real-time syncing, offline support, and scalability, ensuring efficient data management and synchronization across devices. Its flexible data model and serverless architecture simplify development and maintenance, allowing the team to focus on delivering a seamless user experience.

**Consequences:** Implementing Firebase Firestore simplifies data management and synchronization but may require careful consideration of security and privacy measures. However, it offers real-time syncing and offline support, enhancing user experience and ensuring data consistency across devices.

**6.**

**Title:** Additional Frameworks or Technology Stacks

**Status:** Accepted

**Context:** The app requires integration with third-party services such as payment gateways, location-based services, and restaurant inventory management systems.

**Decision:** The team will integrate with third-party APIs to facilitate payment processing, location tracking, and menu synchronization.

**Rationale:** Integrating with third-party APIs streamlines development efforts and reduces the need for building and maintaining complex functionality in-house. Leveraging established services ensures reliability, security, and compliance with industry standards, enhancing the app's functionality and user experience.

**Consequences:** Integrating with third-party APIs simplifies development but may introduce dependencies and potential integration challenges. However, it offers access to specialized services and features, enhancing the app's functionality and user experience.