CSC 431

Intellicup

System Architecture Specification (SAS)

Group 13

Sofia Papa	Team Member
Gargi Yadav	Team Member
Gabriel Huang	Team Member

Version History

Version	Date	Author(s)	Change Comments
1.0	4/4/25	Gargi, Gabe, Sofia	Updating document, dividing tasks, and group communicating over Zoom.
1.1	4/5/25	Sofia	Updated System Overview and Architectural Style.
1.2	4/7/25	Gargi	Began updating System Diagram, Actor Identification, and Design Patterns.
1.3	4/8/25	Gargi	Drafted initial versions of sequence diagrams (Functional Design) based on use cases as well as a class diagram (Structural Design).
1.4	4/9/25	Gabe	Updated the Frameworks
1.5	4/22/25	Sofia	Finalized Sequence Diagrams and Table of Contents page numbers and Table of Figures and Table of Tables.

Table of Contents

Table of Tables

Table of Figures

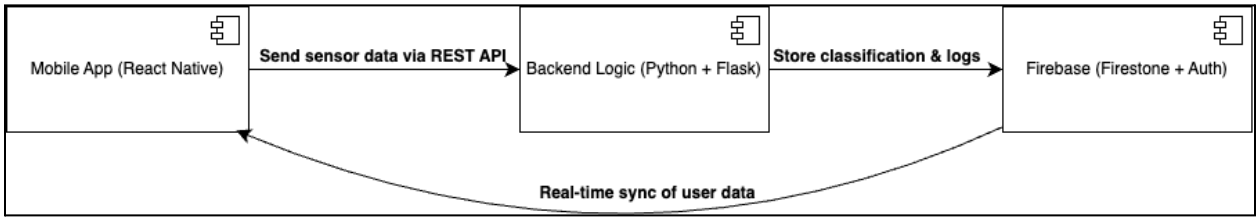
1. System Analysis

1.1 System Overview

IntelliCup is a smart hydration tracking system designed to help users monitor and improve their daily fluid intake through real-time data analysis. It uses simulated sensor data to identify liquids based on measurable properties such as pH, conductivity, and color. This data is used to log consumption history and deliver personalized hydration insights and recommendations. The system is built using a three-tier architecture, which separates the application into the Presentation Tier, Logic Tier, and Data Tier for better scalability, maintainability, and team collaboration.The Presentation Tier is a mobile application developed using React Native. This framework was chosen because it provides a smooth and consistent user experience across both iOS and Android platforms. The mobile app serves as the primary interface for users, allowing them to view hydration data, set personal goals, and receive tailored recommendations. The Logic Tier is responsible for processing and analyzing sensor data. It is built using Python, which was selected for its ease of use, versatility, and strong library support for data processing tasks. This layer handles the classification of liquids, generates insights, and enforces the application's core logic. The Data Tier uses Firebase Firestore to store user information in real time. This ensures that hydration logs and user preferences are always up to date and accessible across devices. Firebase Authentication provides secure user access, and the serverless infrastructure allows the system to scale automatically without the need for complex server management. This architectural approach ensures IntelliCup is responsive, reliable, and capable of evolving as user needs grow.

1.2 System Diagram

The system diagram below illustrates the three-tier architecture of IntelliCup. The Presentation Tier is represented by our mobile application (developed in React Native) that communicates with the Logic Tier (built using Python). The logic Tier processes sensor data, classifies liquids, and then interacts with the Data Tier (Firebase) which handles real-time storage and secure authentication.



Key Data Flows:

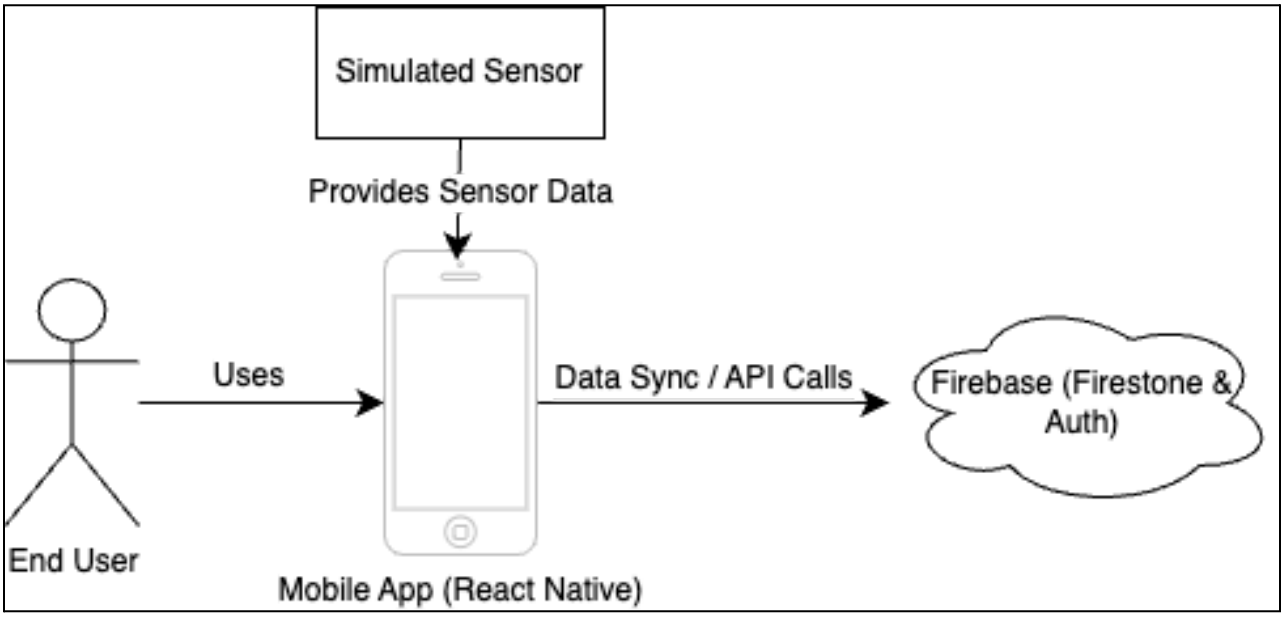
- Mobile App ⇄ Logic Tier: The mobile app sends sensor data via RESTful API calls to the Logic Tier and retrieves processed results.
- Logic Tier ⇄ Data Tier: The Logic Tier stores classification results and consumption logs in Firebase, and data updates are synchronized in real time.

The overall architecture ensures a responsive and scalable system with clear separation of concerns.

1.3 Actor Identification

The following actors interact with the IntelliCup system:

- End User: Uses the mobile application to view hydration data, set personal goals, and track liquid consumption.
- Administrator: Oversees system performance, performs maintenance, and manages updates.
- Simulated Sensor: Acts as a data provider by sending liquid property data (such as pH, conductivity, color, and density) to the Logic Tier.
- Firebase Service: A third-party system that handles authentication and real-time data storage to support system operations.



Each of these actors is integral to the system's workflow, ensuring that data is captured, processed, and displayed correctly.

1.4 Design Rationale

1.4.1 Architectural Style

For the IntelliCup system, we have adopted a Three-Tier Architecture, with the Presentation Tier, Logic Tier, and Data Tier. This architectural style promotes separation of concerns, scalability, and maintainability, which are essential for IntelliCup's long term growth and flexibility.

Presentation Tier:

The Presentation Tier, or Client Layer, of the IntelliCup system is represented by a mobile application developed using React Native. React Native is chosen for its ability to build cross platform apps efficiently and support both iOS and Android. This approach ensures that the application remains consistent across different devices, providing a seamless experience for all users. The mobile app is particularly well suited for on the go users because it allows them to track their hydration in real time through an intuitive and user friendly interface. It also enables users to interact with the system by viewing consumption history, receiving personalized hydration recommendations, and setting hydration goals. Additionally, the app integrates seamlessly with Firebase, our backend solution, to provide real time data synchronization. The frontend communicates securely with the backend via RESTful APIs. RESTful APIs are chosen because they are easy to implement, can scale well as the app grows, and allow for efficient communication. Communication with the backend is handled through RESTful APIs exposed via Firebase Cloud Functions. This helps ensure smooth performance and a responsive user experience. This architecture allows for a responsive, scalable, and easily maintainable application that can evolve alongside future updates to the IntelliCup system.

Logic Tier:

The Logic Tier is responsible for the core processing of the IntelliCup system. It handles tasks such as analyzing sensor data, classifying liquids, and tracking consumption. This tier is built using FastAPI because it is a modern Python framework that is known for its high performance and ease of integration with Python's data processing libraries, such as Pandas and NumPy. Pandas are ideal for handling and analyzing the data generated by the system. Python is also highly scalable, ensuring that the system can grow and evolve as new features or enhancements are added, such as integrating machine learning for more accurate liquid classification. Using FastAPI also enables rapid development and strong API documentation. The logic tier communicates with the data tier to manage real time data storage, user authentication, and security policies. By isolating the application logic from the other tiers, this architecture allows for easier maintenance and ensures that new features can be implemented without disrupting the user interface or data management.

Data Tier:

The Data Tier utilizes Firebase Firestore. We chose this cloud based real time database because we want something that stores user data such as hydration logs, liquid classifications, and preferences in real time to tailor the experience to the user. Firebase's real time synchronization ensures that updates made on one device are instantly reflected on another, without manual refresh or delay. We also decided to use Firebase Authentication to manage and secure user access to ensure data privacy. The choice of Firebase also eliminates the need for manual server management, leveraging its serverless architecture to scale automatically as the user base grows. This makes the Data Tier flexible, cost effective, and scalable without the overhead of managing additional infrastructure.

1.4.2 Design Pattern(s)

For IntelliCup's architecture, the following design patterns have been applied:

- Model-View-ViewModel (MVVM) / Redux Pattern: The mobile application is structured using the MVVM pattern (with Redux for state management). This pattern separates the UI (View) from business logic (ViewModel) and enables effective state management and testing.

- Facade Pattern: Within the Logic Tier, a Facade pattern is implemented to encapsulate the complexity of sensor data processing and potential AI-based liquid classification. This provides a simple interface for the mobile app to interact with diverse system components.

- Singleton Pattern: Key components such as the database connection manager and API client are designed as singletons. This ensures that a single instance is maintained throughout the system's operation, reducing resource overhead and ensuring data consistency.

These design patterns collectively contribute to a modular, maintainable, and scalable codebase that is easy to extend as new requirements emerge.

1.4.3 Framework

For the implementation of the IntelliCup system, we have selected several key frameworks that support our three-tier architecture, each chosen for specific advantages and capabilities:

Presentation Tier Framework: React Native

- React Native is our primary framework for mobile application development, enabling us to create a cross-platform solution that works on both iOS and Android devices
- React Native's component-based architecture aligns perfectly with our MVVM/Redux design pattern, allowing for reusable UI elements and efficient state management.

Logic Tier Framework: Flask/FastAPI

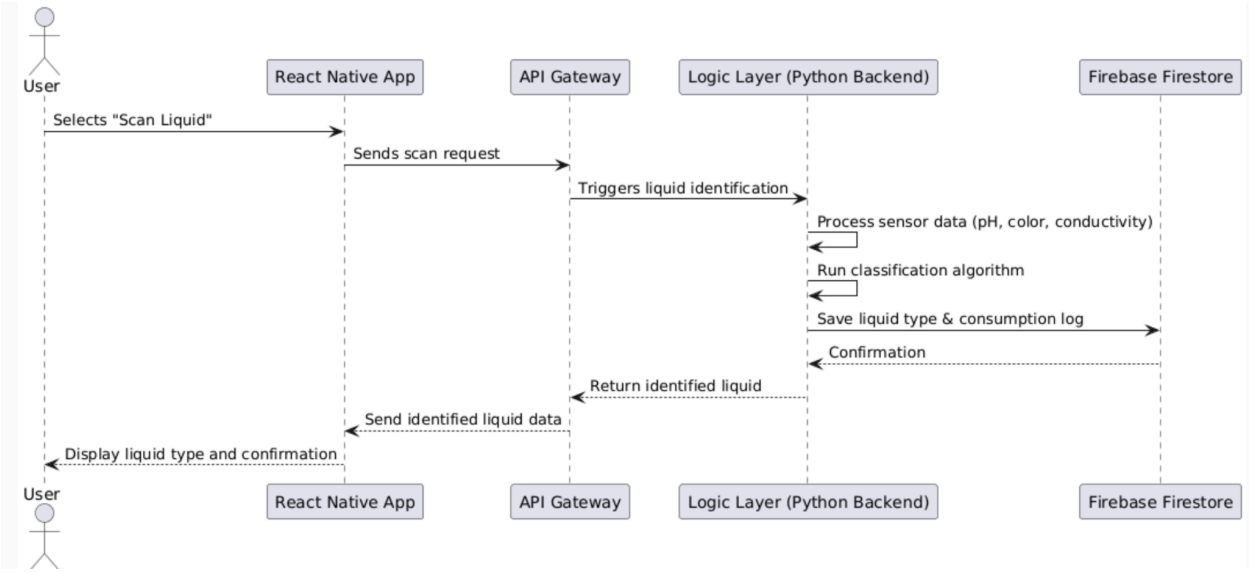
- For our backend API development, we utilize Flask/FastAPI, lightweight Python web frameworks
- Their integration with Python's data processing libraries (Pandas, NumPy) allows for efficient implementation of our liquid classification algorithms.

Data Tier Framework: Firebase SDK

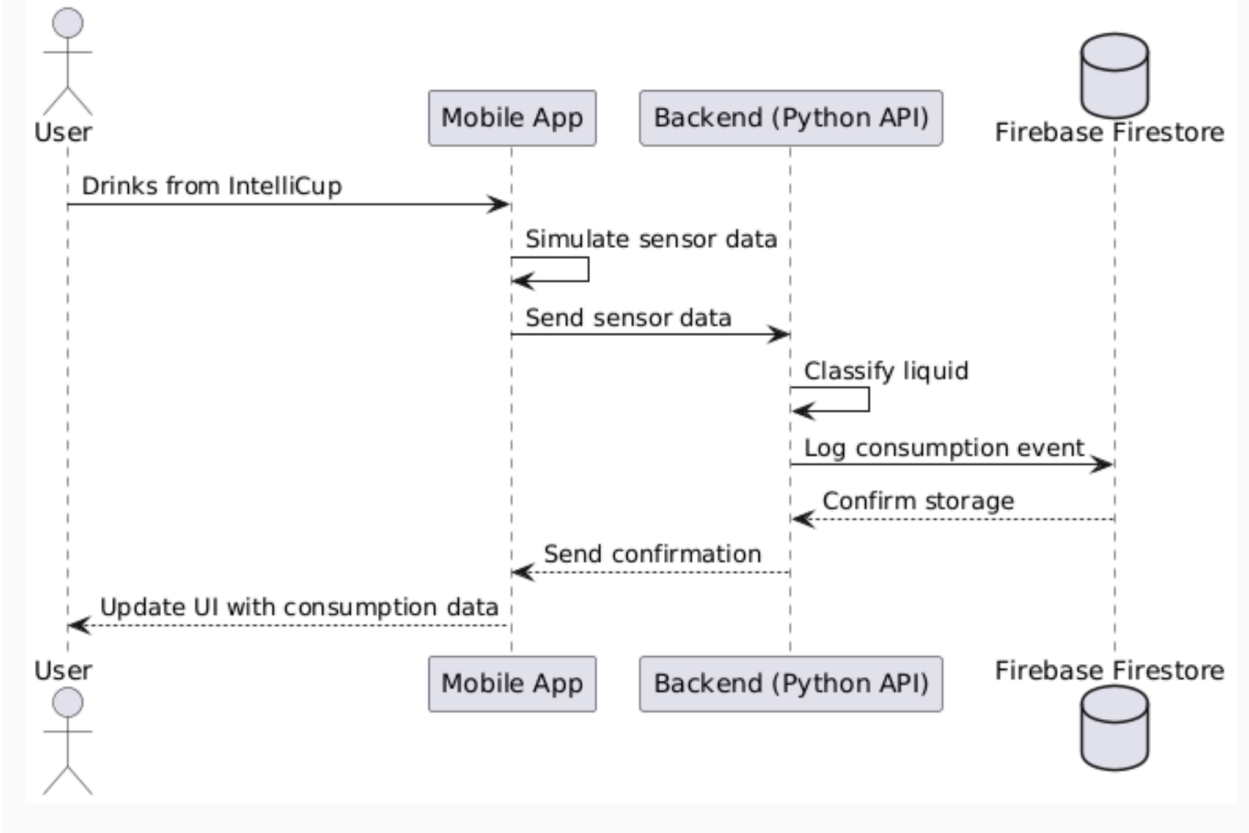
- Firebase provides a comprehensive set of tools for our data management needs, including Firestore for database operations and Firebase Authentication for user management.
- This framework was chosen for its real-time synchronization capabilities, which are essential for delivering instant hydration updates across user devices.
- The Firebase SDK offers robust client libraries for both our mobile application (React Native) and Logic Tier (Python), ensuring seamless integration throughout our system.

2. Functional Design

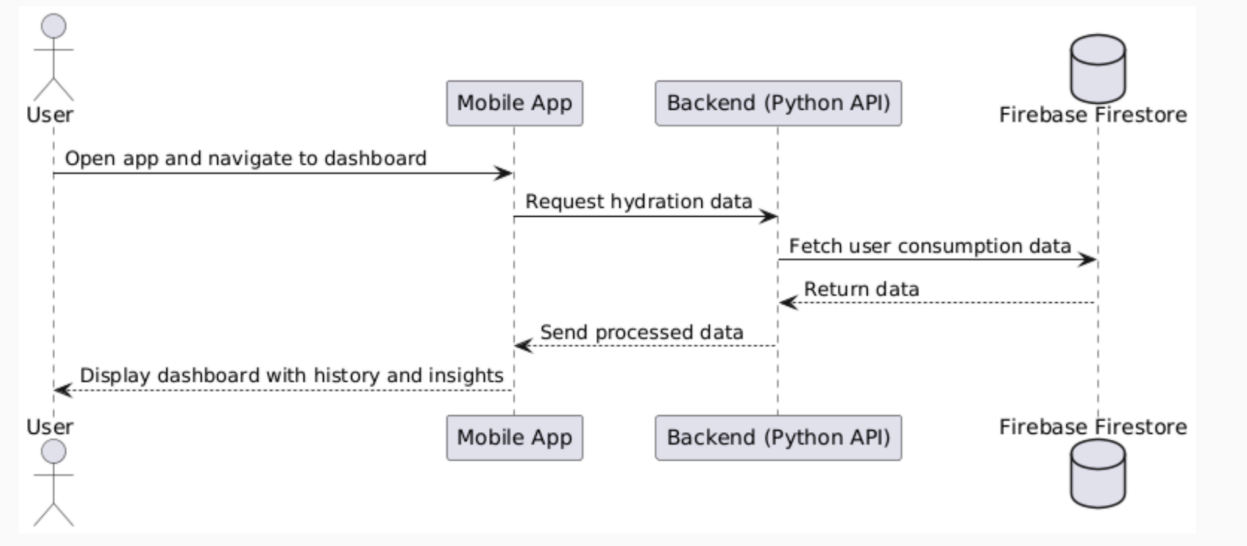
2.1 Sequence Diagram: Liquid Identification



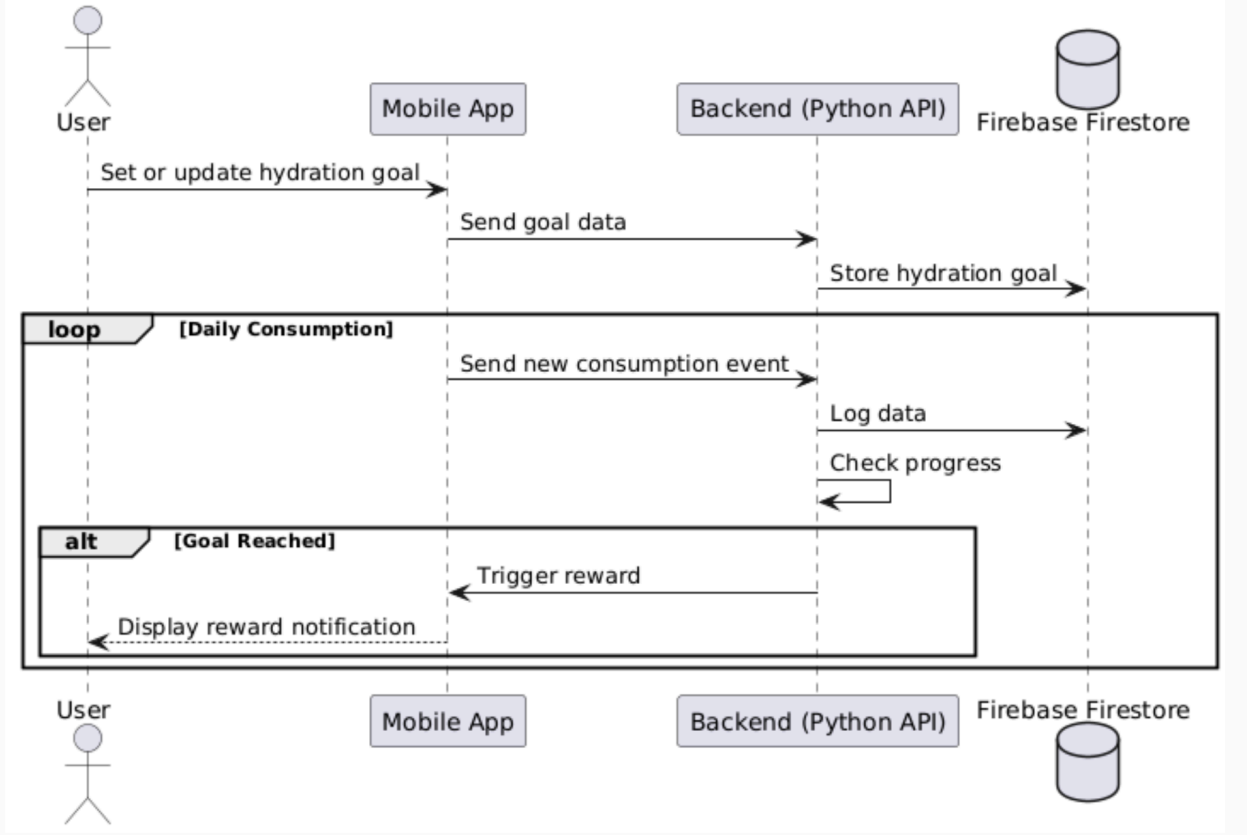
2.2 Sequence Diagram: Consumption Tracking



2.3 Sequence Diagram: User Dashboard



2.4 Sequence Diagram: Hydration Goal & Reward System



3. Structural Design

3.1 Class Diagram