Stage 1 Report

The Bid

Pulse Software Inc.

Amr Mohamed – H00414736

Amin Afara – H00421226

Adnan Bootwala – H00422025

Uzair Siddiqui – H00429823

Aadith Rethnan – H00434613

Syeda Fazila Ahmed – H00483589

Raees Sudheer – H00459509

Geona Dsouza – H00476132

Table of Contents

Part 1

1.1 Purpose and Scope

This project's main objective is to create a Virtual Health Companion to support users in their daily lives. By combining and presenting biomarker data from a variety of wearables, medical devices, and health applications, this system aims to help users lead a healthy life. By using clear, responsibly collected, and processed data, it prioritizes user interaction and enables people to make well-informed decisions about their daily activities and health. The system will help both the younger and older generation users by enabling real-time monitoring, management, and customized insights into health parameters, promoting long-lasting behavioral changes that improve general well-being. The project places a high priority on safe handling of health information, platform and device connectivity, and user-friendly design. The system uses data like biomarker readings, environmental factors (weather and air quality), and personalized advice to help users make smart health choices.

1.2 Aim and Objectives

The aim of this project is to create a virtual health companion that helps people make better choices about their health by using real-time health data. The primary goal is to promote healthier habits by making it simple to use and available to all. This report describes the design, challenges encountered, and solutions for the system's development. It also explains the user interface and technical architecture of the system. The main objectives of the application are:

- To build a Virtual Health Companion system that can record, connect, simulate, and display biomarker data from a variety of medical devices, wearables, and health applications while maintaining compatibility via API-based connections.
- To create a user-centered interface that offers unique insights, allows cross-period data comparison, and allows dashboard customization for efficient health management.
- To support a variety of user types, such as healthcare professionals who need access to detailed biomarker insights and patient trends for

clinical evaluation, as well as general users who want to keep an eye on their health and wellness data.
- To build a virtual companion that actively engages users by providing them with proactive health tips, encouragement, and fun game-like features, all within a safe communication platform.
- To deliver a seamless user experience by ensuring the system works and is responsive across all major platforms (web, mobile, and tablet) and operating systems (Windows and macOS) for a great user experience.
- To ensure data privacy, we will build a security framework that gives users full control over their health information through simple consent control, secure storage of data, and authorized sharing with healthcare professionals.
- To improve the system by thoroughly testing its interface and features. This process involves collecting a mix of qualitative and quantitative feedback, which will ensure the system meets users' needs and expectations

1.3 Overview

This report explains the system’s key functions, such as tracking health data, giving personalized advice, and sending alerts for potential health risks. We also cover the system’s functional and non-functional requirements, while identifying the potential technical and management risks. The document also specifies the necessary resources, software development process, and a detailed project plan with tasks and deadlines. Finally, it breaks down the total project cost and explains how we will test the system with users to get feedback for making it better.

1.4 Functional Requirements

1.4.1 User Table

NO	DESCRIPTION	MoSCoW
U-FR-1	USER ACCOUNT MANAGEMENT	
U-FR-1.1	Users must be able to register an account using email/password.	M
U-FR-1.2	Users must be able to log in directly after registering.	M
U-FR-1.3	Users must be able to log out of their accounts at any given time.	M
U-FR-1.4	Users must be able to reset/change their passwords when required, and if any invalid attempts are made, they must be able to recover their account successfully.	M
U-FR-1.5	Users must be able to update their profile upon successfully registering their account (e.g., Age, gender).	M
U-FR-1.6	Users must be able to view their data at any given time.	M
U-FR-1.7	Users must be able to provide consent for health data usage.	M
U-FR-1.8	Users must be able to delete their account and all associated data permanently.	M
U-FR-1.9	Users must be able to customize their dashboard layout and displayed metrics.	M
U-FR-1.10	Users should be able to set up emergency contacts who can be alerted in case of critical health events.	S
U-FR-1.11	Users must be able to define their primary language and region settings.	M
U-FR-1.12	Users should be able to register with third-party providers (Google, Apple ID, Facebook) with the dashboard being updated subsequently.	S
U-FR-1.13	Users must be able to securely share their health data with their healthcare provider.	M
U-FR-1.14	Users must be able to authorize/revoke access for providers.	M
U-FR-1.15	Users must be able to receive personalized recommendations and feedback from their healthcare provider.	M
U-FR-1.16	Users must be able to contact support or a chatbot assistant for issues.	M
U-FR-1.17	Users should be able to access a help/FAQ section from their account.	S
U-FR-1.18	Users should be able to share selected health statistics or achievements on social media platforms.	S
U-FR-2	BIOMARKER DATA	
U-FR-2.1	Users must be able to connect wearables/Apps (e.g., smartwatches, Apple Health)	M
U-FR-2.2	Users should be able to manually input health data (e.g., blood pressure, mood, medication intake).	S
U-FR-2.3	Users must be able to view a list of all connected devices.	M
U-FR-2.4	Users must be able to disconnect or remove connected devices/apps.	M
U-FR-2.5	Users must be able to view their real-time biomarker data (e.g., heart rate, steps, calories)	M
U-FR-2.6	Users must be able to compare health data across different periods (daily, weekly, monthly)	M
U-FR-2.7	Users must be able to export biomarker data to PDF/CSV.	M
U-FR-2.8	Users must be able to receive daily/weekly health summaries based on recorded biomarker data.	M
U-FR-2.9	Users should be able to overlay environmental data (e.g., weather, air quality) on biomarker graphs.	S
U-FR-2.10	Users should be able to set custom alerts for biomarker thresholds (e.g., heart rate too high/low).	S
U-FR-3	Healthcare Providers	
U-FR-3.1	The healthcare provider must be able to register for an account or log in to their registered account.	M
U-FR-3.2	The healthcare provider must be able to log out of their account at any time.	M
U-FR-3.3	The healthcare provider must be able to successfully delete their account from	M

	profile settings.	
U-FR-3.4	The healthcare provider must be able to reset their password if they forget their login details or experience any invalid attempts.	M
U-FR-3.5	The healthcare provider must be able to update their personal details (e.g., name, contact number, email).	M
U-FR-3.6	The healthcare provider must have a separate dashboard with their patients' biometrics and data.	M
U-FR-3.7	Healthcare providers should be able to alert their patients through a notification system.	S
U-FR-3.8	Healthcare providers must be able to export patients' data in PDF format.	M
U-FR-4	Virtual Companion	
U-FR-4.1	Users must be able to interact with the AI companion chatbot via text commands.	M
U-FR-4.2	Users must be able to receive proactive health insights and suggestions from the chatbot.	M
U-FR-4.3	The virtual companion should provide positive reinforcement (e.g., *"Great job, you reached 10,000 steps today!")* when goals are achieved VIA notifications.	S
U-FR-4.4	Users should be able to access a summary of their recent interactions with the AI companion.	S
U-FR-4.5	The virtual companion could help in setting up personalized goals that they aim to achieve.	C
U-FR-4.6	The companion should provide daily/weekly challenges and congratulate the user upon completion.	S
U-FR-5	NOTIFICATIONS	
U-FR-5.1	Users must receive reminders about device usage anomalies (e.g., swimming mode left on).	M
U-FR-5.2	Users must receive alerts when unusual activity is detected (e.g., 1,000,000 steps).	M
U-FR-5.3	Users could be able to receive reminders for upcoming appointments with their healthcare provider.	C
U-FR-5.4	Users must be able to receive alerts for predicted health risks (e.g., fatigue, stress spikes, elevated heart rates).	M
U-FR-5.5	Users must receive personalized recommendations (e.g., hydration, exercise times).	M
U-FR-5.6	Users should be able to customize which notifications they receive.	S
U-FR-5.7	Users should receive recommendations for outdoor activities based on weather and air quality.	S
U-FR-5.8	Users must be able to mute or pause the virtual companion's notifications.	M
U-FR-5.9	Users should receive daily summary notifications (e.g., steps, sleep, calories, and heart rate).	S

1.4.2 System Table

NO	DESCRIPTION	MoSCoW
S-FR-1	Operation System	
S-FR-1.1	The system must be accessible via all major web browsers (e.g., Chrome, Safari, Firefox, Edge) without loss of functionality.	M
S-FR-1.2	The system must be fully responsive on mobile, tablet, and desktop.	M
S-FR-1.3	The system must be able to send push notifications, emails, and SMS messages.	M
S-FR-1.4	The system should allow users to select and filter which data to export.	S
S-FR-1.5	The system must be able to generate and export reports in various formats (e.g., PDF, CSV).	M
S-FR-1.6	The system must have a robust error logging and monitoring mechanism.	M
S-FR-2	API'S CONNECTIVITY	
S-FR-2.1	The system must integrate with major wearable devices	M
S-FR-2.2	The system must connect to health APIs.	M
S-FR-2.3	The system should pull external contextual data from APIs (weather, location, wearable accuracy) to support better health recommendations.	S
S-FR-2.4	The system must be able to gracefully handle API failures and data synchronization errors.	M
S-FR-3	HEALTHCARE PROFESSIONALS' FUNCTIONS	
S-FR-3.1	The system must provide a dedicated, secure portal for healthcare providers to manage their patient data.	M
S-FR-3.2	The system must enable healthcare providers to set custom alerts and thresholds for individual patients' biomarker data.	M
S-FR-3.3	The system must allow healthcare providers to export detailed patient reports and historical data with explicit patient consent.	M
S-FR-4	DATABASE	
S-FR-4.1	The system must store all biomarker, activity, and health data securely in the database.	M
S-FR-4.2	The system must store users' and healthcare providers' passwords in an encrypted format.	M
S-FR-4.3	The system should support real-time data retrieval from the backend for immediate display on the user interface.	S
S-FR-4.4	The system must allow periodic backups of user data to prevent accidental data loss.	M
S-FR-4.5	The system must implement data integrity checks (e.g., validation rules, duplication prevention, and error correction) to ensure accuracy.	M
S-FR-5	DATA DISPLAY	
S-FR-5.1	The system should provide customizable dashboards, allowing users to change themes and mascot outfits.	S
S-FR-5.2	The system must render real-time updates (e.g., steps completed, calories burned).	M
S-FR-5.3	The system must display data with clear units, labels, and timeframes.	M
S-FR-5.4	The system must provide comparative views (e.g., current week vs. previous week, personal bests).	M
S-FR-6	DATA INPUT	
S-FR-6.1	The system must accurately get data from the connected wearables.	M
S-FR-6.2	The system must ensure that the gathered data will be consistent with connected wearables.	M
S-FR-6.3	The system must provide clear feedback on data input success or failure.	M
S-FR-6.4	The system must allow users to edit or delete previously entered manual data.	M
S-FR-7	SECURITY	
S-FR-7.1	The system must implement Multi-Factor Authentication (MFA) for secure user login.	M
S-FR-7.2	The system must allow users to control their data-sharing preferences.	M

S-FR-7.3	The system must securely delete all user data upon account deletion.	M
S-FR-7.4	The system must enforce strong password policies (minimum length, character variety, and expiry).	M
S-FR-7.5	The system should provide an audit log to track login attempts, unusual activity, and data access for accountability.	S

1.5 Non-Functional Requirements

1.5.1 User Table

NO	DESCRIPTION	MoSCoW
U-NFR-1	Interface Usability	
U-NFR-1.1	Users must be able to interact with the system and understand their health statistics, device status, and other visualizations easily.	M
U-NFR-1.2	The interface should be designed in such a way that it is clear and easy to understand for people of all ages.	S
U-NFR-1.3	Users should be able to access help documentation and frequently asked questions to assist them when they encounter difficulties while using the system.	S
U-NFR-1.4	Users must be able to create personalized dashboards to access their preferred features.	M
U-NFR-2	User’s Data Security	
U-NFR-2.1	Users should be able to have access to logs of data access and sharing actions for unauthorized access monitoring.	S
U-NFR-2.2	Users must have the option to determine their data sharing preferences with third-party applications and healthcare providers.	M
U-NFR-3	Device Compatibility	
U-NFR-3.1	Users must be able to log in to the system from multiple devices simultaneously without session conflicts.	M
U-NFR-4	Device Responsiveness	
U-NFR-4.1	The system must load all user dashboards and primary views within 5 seconds under normal operating conditions	M
U-NFR-4.2	Users' devices must be able to display the interface responsively without loss of functionality	M
U-NFR-5	Notifications	
U-NFR-5.1	Users must receive critical health alerts (e.g., predicted health risks, emergency alerts) within a minute of the event being triggered by the system.	M
U-NFR-5.2	Users must experience a notification delivery success rate across all channels (Email, SMS) of at least 99.5%.	M
U-NFR-5.3	Users should be able to customize notification preferences (e.g., type, time of day).	S
U-NFR-6	Scalability	
U-NFR-6.1	Users must be able to add more connected devices and accumulate historical data over multiple days without performance degradation.	M
U-NFR-6.2	Users should not experience long wait times or bad performance when there are more devices connected and greater volumes of data added.	S
U-NFR-6.3	Users must be able to complete data export functions (PDF/CSV) of their health data within 10 seconds.	M

1.5.2 System Table

NO	DESCRIPTION	MoSCoW
S-NFR-1	System Responsiveness and Performance	
S-NFR-1.1	The system should minimize battery consumption on mobile devices when running in the background.	S
S-NFR-2	Database	
S-NFR-2.1	The system should be able to process health data from a large number of smart devices and ensure there is no data lost between transactions.	S
S-NFR-2.2	The system should ensure that connected devices are able to push data into the database in real time.	S
S-NFR-2.3	The system must implement a data retention policy that allows for the secure and permanent deletion of user data upon request.	M
S-NFR-3	Data Security	
S-NFR-3.1	The system must implement robust access control mechanisms to protect users' sensitive health information	M
S-NFR-3.2	The system must successfully prevent and report common web vulnerabilities such as SQL Injection and Cross-Site Scripting.	M
S-NFR-3.3	The system must use industry-standard encryption protocols to secure user login and data transmission.	M
S-NFR-4	Notifications	
S-NFR-4.1	The system should use a reliable third-party service for SMS and email delivery with a high guaranteed delivery rate.	S
S-NFR-4.2	The system must gracefully handle and log failures in notification delivery, with an automatic retry mechanism for critical alerts.	M
S-NFR-5	System Reliability	
S-NFR-5.1	The system must be available 99.9% of the time for users to monitor their health and interact with the platform with minimal downtime.	M
S-NFR-5.2	The system must be able to recover from failure with a minimal loss of data.	M

Part 2

2.1 Use-Case Diagram

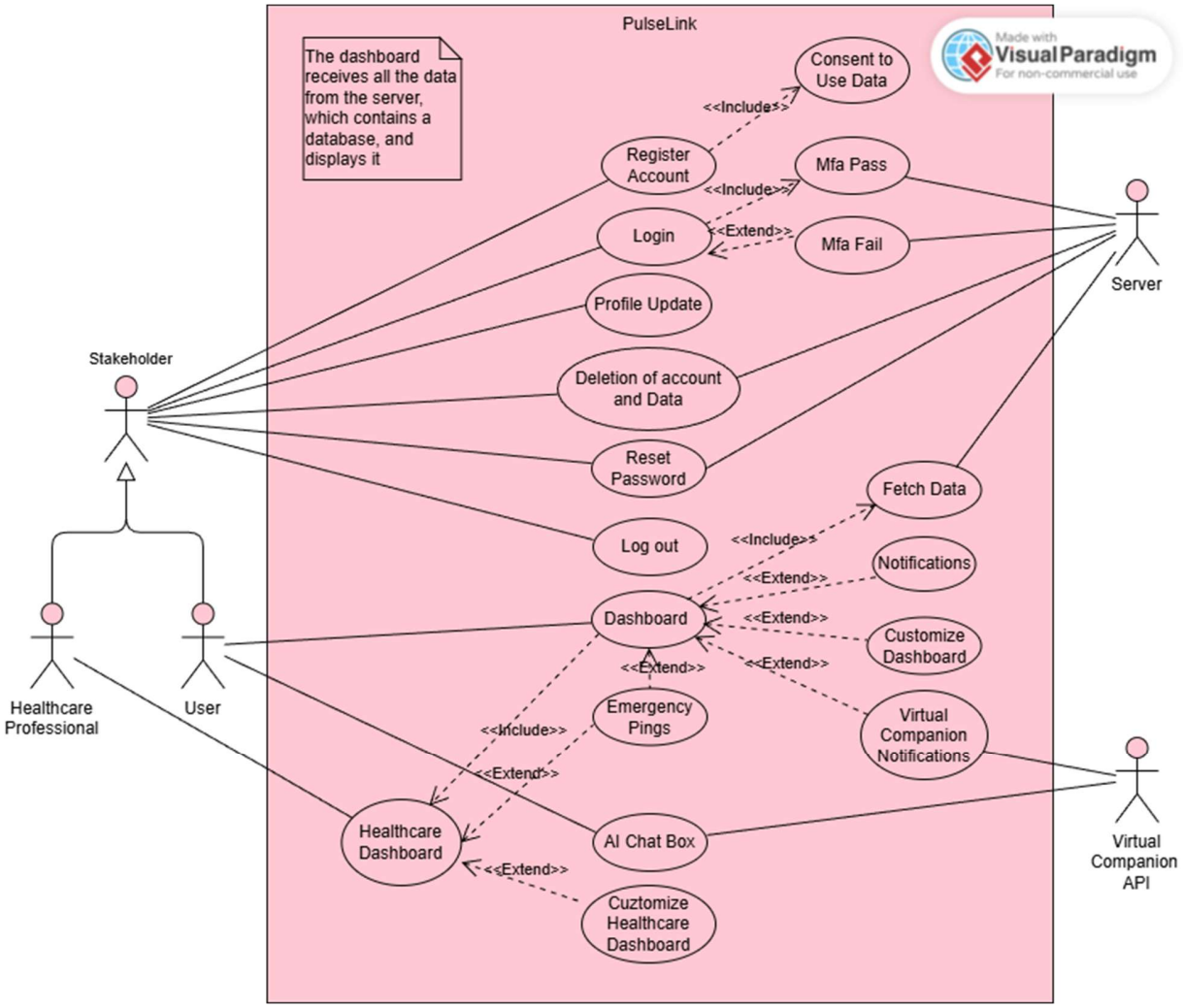


Figure 1 - Use-Case Diagram showing user interactions with the app

Figure 1 demonstrates the different actions the user can take to interact with our web application. There are two main actors on the user end: User and Healthcare provider, with a parent actor for the actions that they can both take. The actors on the represent the backend and APIs as they interact with the user-cases to fetch data or establish connections for the users. For more detailed representations, Use-Case Specifications are included for each case in the Appendices.

2.2 Traceability Matrix

Use Case ID	Description	User Requirements	System Requirements
UC-1	Register Account	U-FR-**1.1**, U-FR-**1.2**, U-FR-**3.1**	S-FR-**7.4**
UC-2	Consent to adding data	U-FR.**1.7**, U-FR-**1.13**, U-FR-**1.14**, U-FR-**2.2**, U-FR-**3.1**, U-FR-**3.5**	S-FR-**3.3**, S-FR-**4.1**, S-FR-**7.2**
UC-3	Login (with MFA)	U-FR-**1.2**, U-FR-**3.1**	S-FR-**7.1**, S-FR-**7.5**
UC-4	Profile Update	U-FR-**1.11**, U-FR-**1.5**, U-FR-**1.12**, U-FR-**1.14**, U-FR-**1.18**, U-FR-**2.1**, U-FR-**2.4**, U-FR-**3.3**, U-FR-**3.4**, U-FR-**3.5**, U-FR-**5.2**, U-FR-**5.6**	S-FR-**6.4**
UC-5	Delete Account and Data	U-FR-**1.8**, U-FR-**2.4**, U-FR-**3.3**	S-FR-**7.3**
UC-6	Reset Password	U-FR-**1.4**, U-FR-**3.4**	S-FR-**4.2**, S-FR-**7.4**
UC-7	Log out	U-FR-**1.3**, U-FR-**3.2**	
UC-8	View Dashboard	U-FR-**1.6**, U-FR-**1.17**, U-FR-**2.3**, U-FR-**2.5**, U-FR-**2.6**, U-FR-**3.8**, U-FR-**4.4**, U-FR-**5.9**	S-FR-**1.1**, S-FR-**1.2**, S-FR-**1.4**, S-FR-**1.5**, S-FR-**1.6**, S-FR-**5.2**, S-FR-**5.3**, S-FR-**5.4**, S-FR-**6.2**
UC-9	Fetch Data	U-FR-**1.6**, U-FR-**2.5**, U-FR-**2.7**, U-FR-**2.8**, U-FR-**2.9**, U-FR-**3.8**	S-FR-2.**1**, S-FR-**2.2**, S-FR-**2.3**, S-FR-**2.4**, S-FR-**3.3**, S-FR-**4.1**, S-FR-**4.3**, S-FR-**6.1**, S-FR-**6.3**

UC-10	Receive Notifications	U-FR-**1.15**, U-FR-**2.7**, U-FR-**2.8**, U-FR-**2.10**, U-FR-**4.1**, U-FR-**4.2**, U-FR-**4.3**, U-FR-**4.6**, U-FR-**5.1**, U-FR-**5.3**, U-FR-**5.4**, U-FR-**5.5**, U-FR-**5.6**, U-FR-**5.8**, U-FR-**5.9**	S-FR-**1.3**, S-FR-**3.2**
UC-11	Customize Dashboard	U-FR-**1.9**, U-FR-**1.12**, U-FR-**1.15**, U-FR-**2.1**, U-FR-**2.2**, U-FR-**2.4**, U-FR-**2.9**, U-FR-**3.7**, U-FR-**4.5**, U-FR-**5.5**	S-FR-**5.1**
UC-12	Receive Virtual Companion Notifications	U-FR-**4.1**, U-FR-**4.2**, U-FR-**4.3**, U-FR-**4.6**	
UC-13	Set Emergency Pings	U-FR-**1.10**, U-FR-**5.4**, U-FR-**5.8**	S-FR-**3.2**
UC-14	View Healthcare Dashboard	U-FR-**3.6**, U-FR-**5.3**	S-FR-**3.1**, S-FR-**3.3**, S-FR-**5.3**, S-FR-**5.4**, S-FR-**6.2**
UC-15	Interact with AI Chat Box	U-FR-**1.16**, U-FR-**4.1**, U-FR-**4.4**, U-FR-**4.5**	
UC-16	Customize Healthcare Dashboard	U-FR-**3.7**	S-FR-**3.1**,

The traceability matrix shows how the use cases in section 2.1 cover the main requirements of the application. Many use cases focus on user requirements, e.g. Customize Dashboard, while others like Fetch Data implement system requirements. This table shows the main features of the web application in a friendly and readable way, while also providing a framework for later sections.

2.3 Sequence Diagrams

A sequence diagram shows the flow of data between different operations when a user performs an action. They allow for a lower level view of the application’s operations during common use cases. Figure 2 shows the flow of data from the user’s end device to Firestore and back. Figures 10 to 12 in the Appendices show other use cases as sequence diagrams.

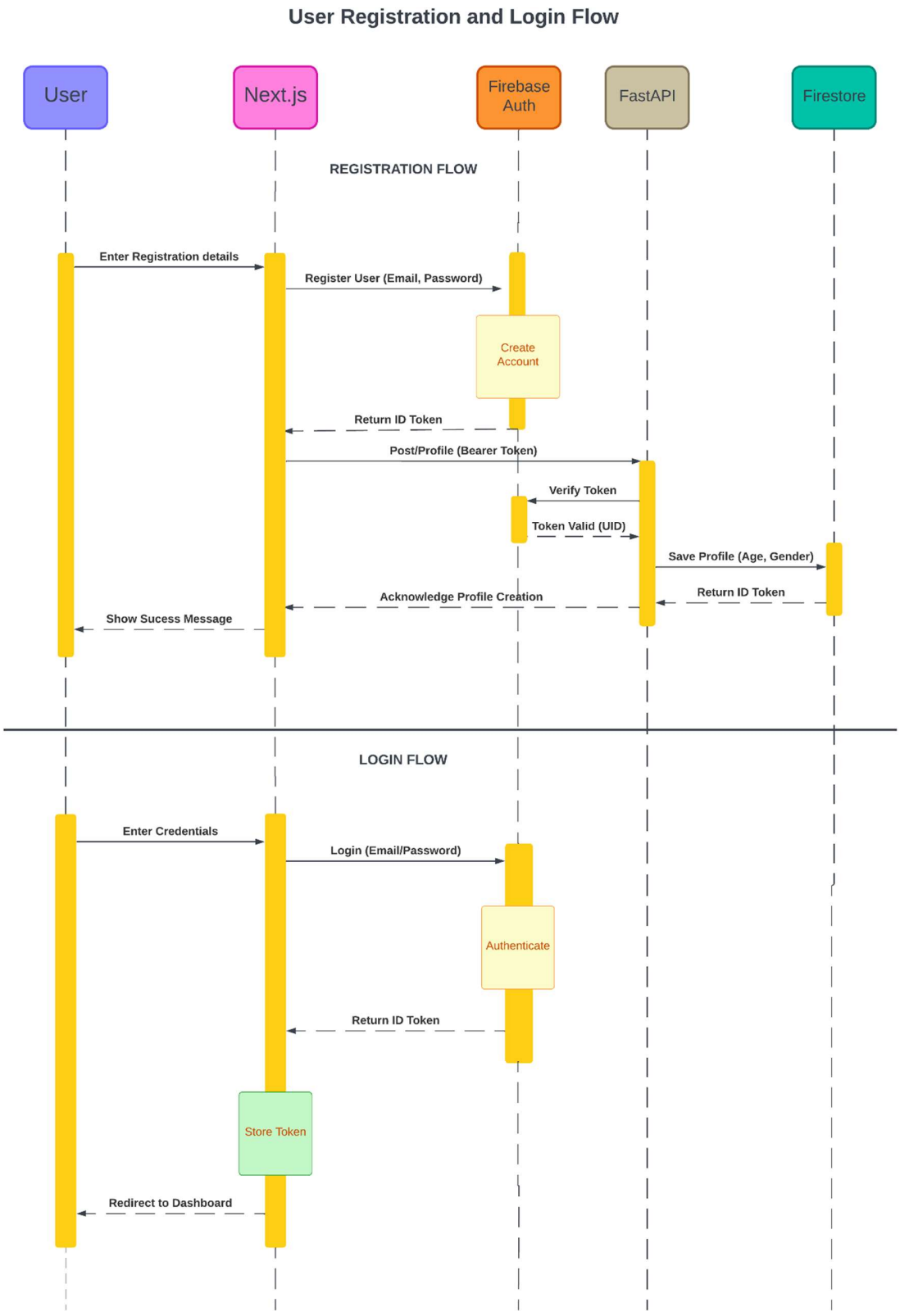


Figure 2 – User Registration and Login Flow Sequence Diagram

Part 3

3.1 Introduction

Risk analysis is a crucial component of PulseLink's planning, which is a Virtual Health Companion that provides an opportunity to combine biomarker information gathered into a single and easily accessible experience. Considering the complexity of the system and the academic context in which the student team is developing the system, the section will identify the risks the team can face and ways of alleviating them. This is aimed at making sure that PulseLink is ready and on time, as well as delivering a quality product that can stand the test of time as it goes through the development lifecycle.

3.2 Risk Analysis

Risk	Type	Likelihood	Impact	Mitigation Strategy
Team member illness or absence	Human	Medium	High	Assign backup roles and maintain shared documentation to ensure continuity.
Missed deadlines due to certain circumstances	Planning	High	High	Use of shared calendars, use a Kanban Board to set internal deadlines before official ones, and hold weekly progress meetings.
Technical issues (e.g., software bugs, device incompatibility)	Technical	Medium	High	Choose stable frameworks, conduct early testing, and maintain version control.
Poor communication within the team	Organizational	Medium	Medium	Use collaboration tools such as Trello, MS Teams, etc., and establish clear communication protocols.

Data privacy and security concerns	Security	Low	High	Use encryption, anonymize user data, and follow GDPR-compliant practices.
Feature creep	Planning	Medium	Medium	Use the MoSCoW prioritization and regularly review the information.
Inadequate usability feedback	Evaluation	Medium	Medium	select different participants early and conduct pilot usability tests.
Device simulation errors (e.g., unrealistic biomarker data)	Technical	Medium	Medium	Validate simulated data ranges and include error-handling logic.
Lack of Managerial Engagement	Stakeholder	Low	Medium	Schedule regular check-ins with the project manager and seek feedback on what can be improved.
Software/tool failure (e.g., GitHub downtime)	Technical	Low	Medium	Maintain local backups and use other platforms when necessary.

3.3 Risk Management

- Monitoring and Review: Risk would be checked daily during sprint meetings. There will be a common risk log that will be updated on a regular basis. (Bannerman, 2008)
- Ownership/Responsibility: Each member of the team oversees reporting encountered problems in his or her area, clear ownership structures are widely identified as essential in software project risk governance (Sarigiannidis et al., 2011).

- Contingency Planning: In the case of high-impact risks, contingencies will be written down. As an example, alternative software tools will be found in case of technical failure, this aligns with best practices in risk mitigation planning (Ahmed, 2022).
- Integration with Project Planning: Risk mitigation plans are synchronized with the project plans and decisions that have been detailed in Section 4. Risk status will be modified, and dependencies will be adjusted accordingly with updated timelines (Kalluri, 2022).
- Communication Protocols: There will be clear lines and expectations of communication in order to reduce the occurrence of misunderstandings and delays, effective communication is consistently highlighted as a key factor in reducing organizational and team-based risks (Bisikirskienė, 2025)

The risks identified in Section 3.2 were based on the well-known categories of software project risks explored in the literature, such as the human and organizational risks, technical uncertainties, planning-related risks, and stakeholder issues (European Union, 2016).

Part 4

4.1 Software Process

For the PulseLink project, we have chosen to adopt Agile development, with the Scrum process allowing us to focus on rapidly and repeatedly releasing testable versions of the software for review and enhancement.

4.1.1 Agile/Scrum Advantages for PulseLink

1. **Collaboration**: Agile, more specifically Scrum, promotes close collaboration between all teams by allowing everyone to work in parallel and simultaneously provide constant feedback and improvements for all aspects of the project.

2. **Constantly Evolving Requirements**: With the constant flow of new features from feedback and user-centered evaluations, Scrum makes room for constant adaptation using Sprints. This ensures that the final product remains relevant and competitive in the rapidly evolving field of health/medical software

3. **User-Centered Development**: Agile emphasizes frequent user feedback and collaboration. Which is crucial for a health assistance app, where user experience and usability is top priority.

4. **Risk Management**: By breaking down a project into many short sprints, we can identify and address potential risks early on, reducing the likelihood of major setbacks, surprises, and morale/motivation drops later in the development process.

4.1.2 Sprints

The sprint is one week of designing, coding, and testing the product. Developers will be split into the team and Scrum Master, with the team being cross-functional and membership only changing between sprints. Before starting the development process, a product backlog will be created, with the requirements and a list of all desired work on the project, and an estimate of how long it would take to implement. It will be prioritized by the product owner and reprioritized at the start of each sprint.

The development process starts by collaboratively deciding on a sprint goal and creating a sprint backlog, selecting tasks from the product backlog (functional requirements) that we can commit to completing, and estimating the number of hours it takes to complete them. There will be a short Daily Scrum meeting for the entire team for tracking progress, identifying roadblocks, and committing to working on a task in front of the team.

After the duration, there will be a Sprint Review to demonstrate what was accomplished during the sprint for the team and the line manager, gathering feedback. There will also be a Sprint Retrospective to reflect

on what is working and what is not, and what we can improve for our next sprint to improve efficiency and productivity.

4.2 Role Breakdown

With a team of 8 people, we assign roles based on individual strengths and interests, and even though each member has a designated role, the tasks to be chosen by each member for sprints are flexible, and no task will be assigned.

Member	Role	Responsibilities
Amr Mohamed	Project Manager, Technical Lead, and Scrum Master	Report Writing; Frontend and Backend Development; DB Development; Managing and Maintaining Software Process
Fazila Ahmed	UI/UX Designer	Report Writing; Prototyping; UI/UX Design
Adnan Bootwala	UI/UX Designer	Report Writing; Prototyping; UI/UX Design
Raees Sudheer	Frontend Engineer	Report Writing; UI/UX Design; Frontend Development
Geona Dsouza	Frontend Engineer	Report Writing; UI/UX Design; Frontend Development
Amin Afara	Back-end Engineer	Report Writing; Backend Development; DB Development
Uzair Siddiqui	Quality Assurance Engineer	Report Writing; Usability Assessment; Testing
Aadith Rethnan	Quality Assurance Engineer	Report Writing; Usability Assessment; Testing

4.3 Gantt Chart

For effective communication, progression, and management of resources, Gantt charts, represented in figures 3 to 6, are used to show all main tasks and their duration, start/end dates, and the team working on them. This allows all members of the team to have a good grasp of the scope and progress, especially when used alongside Trello.

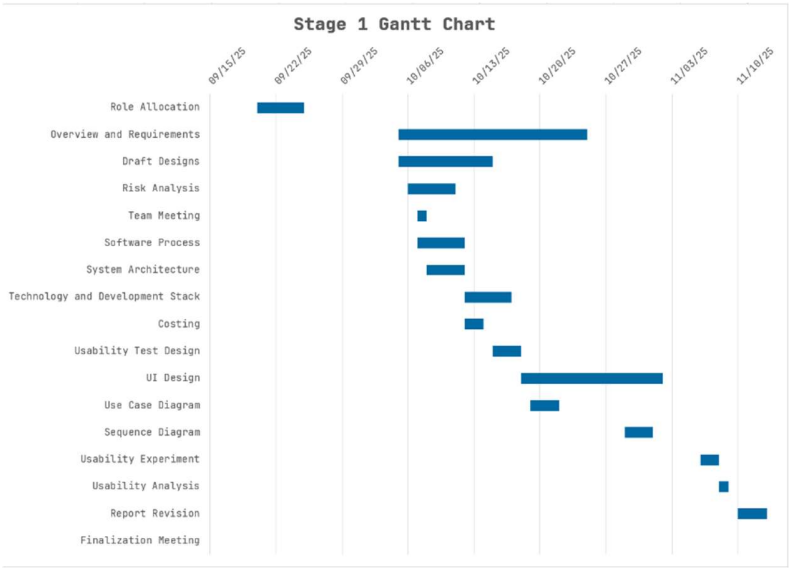


Figure 3 – Gantt Chart Stage 1

Stage 1 Task	Begin Date	Duration	End Date	Team
Role Allocation	20/09/25	5	25/09/2025	All
Line Manager Meeting 1	29/09/25	0	29/09/2025	All
Overview and Requirements	05/10/25	20	25/10/25	Usability
Draft Designs	05/10/25	10	05/15/25	Design
Risk Analysis	06/10/25	5	13/10/25	Design
Team Meeting	07/10/25	1	8/10/25	All
Software Process	07/10/25	5	14/10/25	Software
Line Manager Meeting 2	08/10/25	0	8/10/25	All
System Architecture	08/10/25	4	14/10/25	Software
Technology and Development Stack	12/10/25	5	17/10/25	Software
Costing	15/10/25	2	19/10/25	Software
Usability Test Design	15/10/25	3	20/10/25	Usability
Line Manager Meeting 3	18/10/25	0	18/10/25	All
UI Design	18/10/25	15	7/11/25	Design
Use Case Diagram	19/10/25	3	22/10/25	Design
Line Manager Meeting 4	28/10/25	0	28/10/25	All
Sequence Diagram	29/10/25	3	2/11/25	Design
Line Manager Meeting 5	03/11/25	0	3/11/25	All
Usability Experiment	06/11/25	2	10/11/25	Usability
Usability Analysis	08/11/25	1	10/11/25	Usability
Report Revision	10/11/25	6	18/11/25	All
Finalization Meeting	20/11/25	1	21/11/25	All
Stage 1 Submission	21/11/25	0	21/11/25	All

Figure 4 – Gantt Chart Table Stage 1

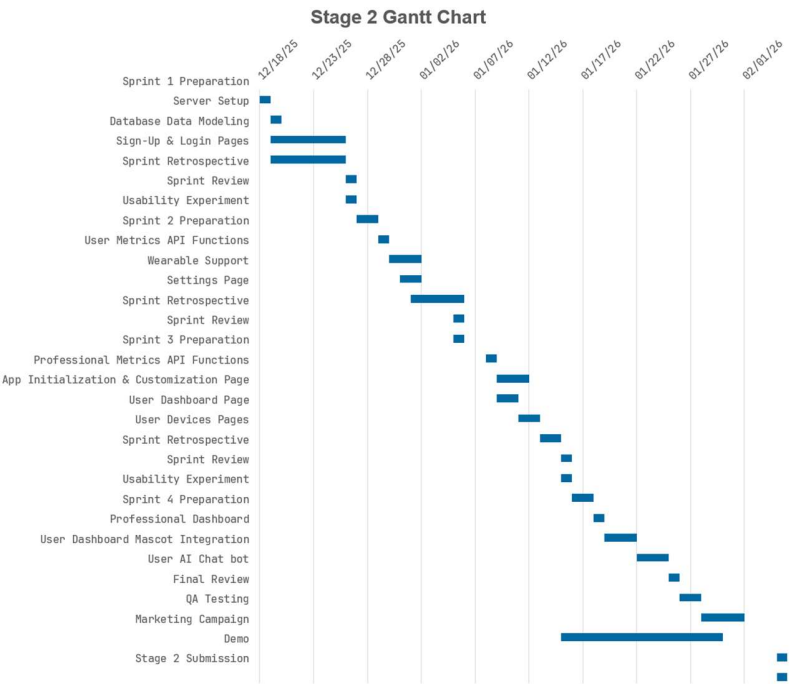


Figure 5 – Gantt Chart Stage 2

Stage 2 Task	Begin Date	Days	End Date	Team
Sprint 1 Preparation	18/12/25	1	19/12/25	All
Server Setup	19/12/25	1	22/12/25	Backend
Database Data Modeling	19/12/25	7	30/12/25	Backend
Sign-Up & Login Pages	19/12/25	7	30/12/25	Frontend, Backend
Sprint Retrospective	26/12/25	1	29/12/25	All
Sprint Review	26/12/25	1	27/12/25	All
Usability Experiment	27/12/25	2	30/12/25	Usability
Sprint 2 Preparation	29/12/25	1	30/12/25	All
User Metrics API Functions	30/12/25	3	02/01/26	Backend
Wearable Support	31/12/25	2	02/01/26	Backend
Settings Page	01/01/26	5	06/01/26	Design, Frontend
Sprint Retrospective	05/01/26	1	06/01/26	All
Sprint Review	05/01/26	1	06/01/26	All
Sprint 3 Preparation	08/01/26	1	09/01/26	All
Professional Metrics API Functions	09/01/26	3	13/01/26	Backend
App Initialization & Customization	09/01/26	2	11/01/26	Design, Frontend
User Dashboard Page	11/01/26	2	13/01/26	Design Frontend
User Devices Pages	13/01/26	2	15/01/26	Design Frontend
Sprint Retrospective	15/01/26	1	16/01/26	All
Sprint Review	15/01/26	1	16/01/26	All
Usability Experiment	16/01/26	2	18/01/26	Usability
Sprint 4 Preparation	18/01/26	1	19/01/26	All
Professional Dashboard	19/01/26	3	22/01/26	Design, Frontend
User Dashboard Mascot Integration	22/01/26	3	25/01/26	Design, Frontend
User AI Chat bot	25/01/26	1	26/01/26	Frontend
Final Review	26/01/26	2	28/01/26	All
QA Testing	28/01/26	4	02/02/26	Design, Usability
Marketing Campaign	15/01/26	15	05/02/26	Design
Demo	04/02/26	1	05/02/26	All
Stage 2 Submission	04/02/26	1	05/02/26	All

Figure 6 – Gantt Chart Table Stage 2

4.4 System Architecture

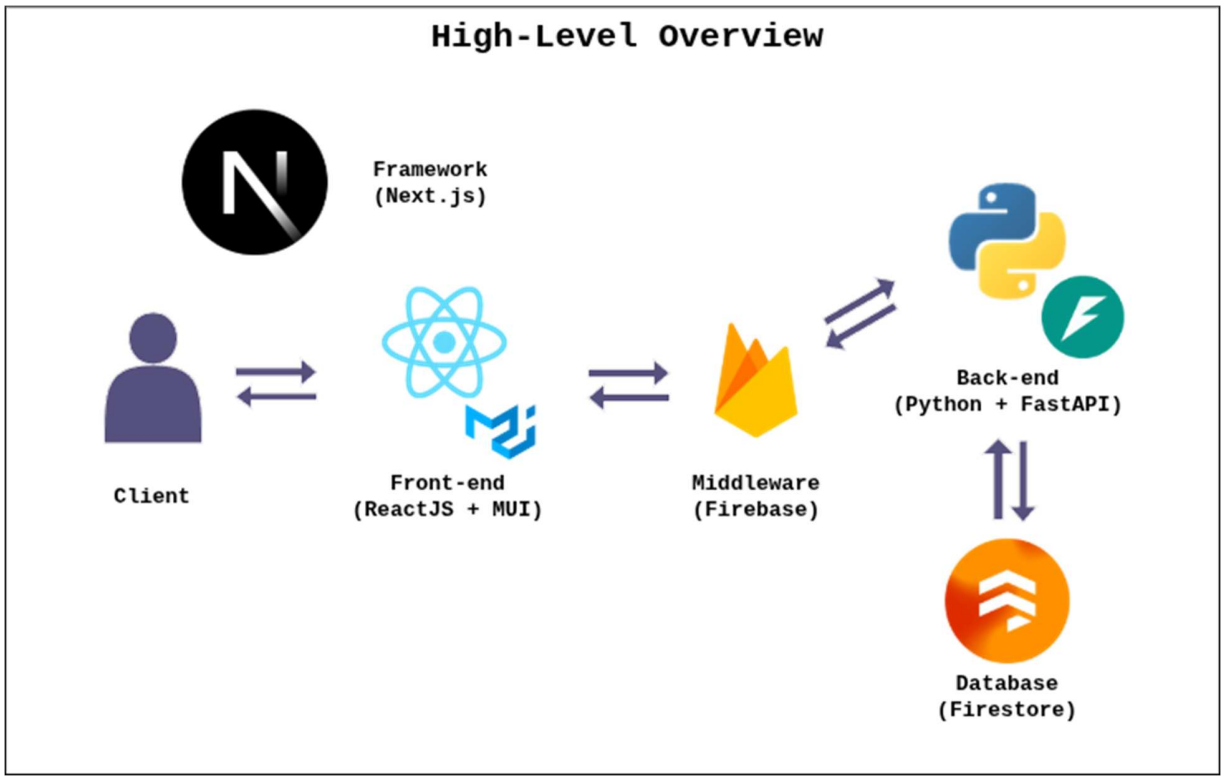


Figure 7 - High-Level System Architecture Diagram

Figure 7 shows the workflow of our system architecture from user interaction to server processing. When a user logs in or changes settings, the request is handled by the Frontend (React in a Next.js framework). The frontend authenticates with Firebase Auth and includes the ID token with each API call to the Python + FastAPI backend (Authorization: Bearer <ID_TOKEN>). FastAPI verifies the token server-side, applies role (user/professional) and consent checks, runs the use-case operation, e.g. Fetching data, and then reads/writes JSON documents in Firestore. The backend returns structured JSON to the frontend, which updates the UI. We chose a web application (Next.js) over a native app for faster development, platform independence, and a single codebase to maintain.

For security, sensitive reads/writes do not occur directly from the frontend to Firestore; they always go through FastAPI to keep validation, authorization, and auditing consistent. This separation of concerns

keeps layers loosely coupled for testability and maintainability, while efficiently handling user requests.

4.5 Technology Stack

4.5.1 Framework

For the web application, we chose to use established frameworks to ease the development process & ensure responsiveness and compatibility with a range of devices. The chosen framework is Next.js, a Full-stack framework that simplifies routing and optimizes performance across devices using Server-Side Rendering and Server-Side Generation, and supports our chosen middleware, Firebase, with its many synchronization and scalability features. This will allow us to be compatible with all required browsers, such as Chrome, Opera, Firefox, Safari, and Operating Systems such as Windows, macOS, and Linux.

Next.js also uses the same language as the Frontend library ReactJS: JavaScript, which is a language that is familiar to the team, allowing smooth and streamlined development for the web application. Next.js is also compatible with the middleware Firebase, which is 4.5.2 Frontend Development

4.5.2 Frontend

React

ReactJS was chosen as the main front-end library due to its component-based architecture that allows the development of efficient, reusable components. Components allow complex web applications to update data without requiring a page reload, which is beneficial for our project due to the need to update the dashboard's metrics from various devices in real time. React's ability to update only the chosen components makes it ideal for building the main dynamic dashboard page of the project.

ReactJS has a large community and ecosystem, with it being developed originally by Facebook (now Meta), and the community provides an extensive range of resources, libraries, and solutions. It is also

compatible with many of the libraries and APIs used for requesting and managing data.

Overall, with its modularity, efficiency, and integration with the rest of the stack, ReactJS serves as the main library for developing the user interface

Next.JS

Next.js is a ReactJS Framework allowing us to develop our web application fast and optimized. The Server-Side Rendering improves performance, and the Routing splits the code automatically for the best user experience.

Although there is a learning curve for advanced features and a need for some custom configuration, the framework will be used optimally to achieve the best results

Material UI (MUI)

MUI is a React Library with common stylized responsive components, allowing us to build at an accelerated pace without sacrificing flexibility or control. It provides customizable components with community-driven documentation and accessibility features.

By leveraging MUI alongside ReactJS, there will be a smooth transition from prototype to modern web application.

4.5.3 Backend and Database

Python + FastAPI

Python, specifically version 3.11, was chosen to be the language for the backend due to its readability, simplicity, and extensive library ecosystem. Its support for asynchronous programming also makes it a strong choice for web application backends due to their need to handle real-time data and HTTP requests. FastAPI is a modern, fast web framework for building APIs in Python. Known for its high-performance capabilities, it supports the creation of RESTful APIs with minimal code and provides asynchronous support, making it ideal for handling real-time data and managing multiple device connections. FastAPI gives typed validation

(Pydantic), clean routing, and auto docs, fast to implement and easy for the Backend engineers.

Firebase + Firestore

Firebase provides managed backend services (Auth, Hosting, Audits), and Firestore is its managed NoSQL document database. We use Firestore to store health and consent data as JSON documents, like users, profiles, links, and measurements. It requires zero server maintenance, auto-scales for reads/writes, and supports real-time listeners when we need live updates. Firestore's composite indexes enable fast queries by owner and time window, while batched writes/transactions ensure atomic updates for sensitive flows such as link/unlink, and audit write. Authentication is handled by Firebase Auth, the frontend sends an ID token that our FastAPI backend verifies server-side before performing any read/write, so role/consent checks and validation remain centralized in the API. The backend also performs server-side aggregation for daily/weekly summaries, writes audit entries for sensitive actions, and can export data to CSV or Cloud Storage to support reporting and **right-to-erasure** workflows. This choice gives us fast implementation, secure integration with the frontend, and scalable, JSON-first storage aligned with our scope.

4.5.4 API Infrastructure

API Overview

Our team plans to use RESTful API infrastructure to manage communication between the frontend and backend of our health application. FastAPI will serve as the main backend, providing endpoints for data management and user interaction. Firebase's authentication is used to handle sign-in and authorization, ensuring security and privacy. Firestore API will manage database interactions, storing and retrieving user data, device logs, and historical usage statistics. Axios will be used at the front end to manage HTTP requests between the front end and FastAPI backend. It provides an intuitive interface for sending requests and handling responses, with automatic JSON parsing, advanced error handling capabilities, and wide browser support. Finally, to allow for real-time data handling, WebSocket will be implemented where it is necessary, giving users better feedback on important health metrics.

Example Interaction of user logging into the web application:

1) Login (frontend): Next.js uses Firebase Auth → Obtains an ID token
2) Call API: send Authorization: Bearer <ID_TOKEN> to FastAPI
3) Verify (backend): FastAPI verifies the ID token, loads the user's role and consent from Firestore, executes the use-case, then reads/writes Firestore documents
4) Respond: return JSON to the frontend.

4.6 Assessing Software

Code

Alongside unit tests, a CI/CD pipeline will be implemented to ensure code is maintained and critical errors are addressed before merging branches or staging any changes. During a sprint, QA Engineers will also rigorously test any feature implemented to catch bugs and ensure development continues without disruptions.

Usability

As user acceptance and experience are top priorities, during development, usability and user experience tests will be conducted, using scenarios that map to relevant user stories. This allows us to get crucial feedback early and make sure development stays user driven.

4.7 Design and Version Control

Design Tools

The prototype and initial designs/mockups will all be developed using Figma Pro to allow for full creativity and flexibility after receiving feedback from the usability evaluations. At each stage, interactive and usable prototypes of the web application will be developed for testing purposes, allowing us to visualize the application's design and experience without needing to implement any code, optimizing the feedback and iteration loop.

Version Control

To allow for efficient management of code, documentation, and collaboration for the team, we chose GitHub as our version control

platform, allowing us to maintain an organized repository and allowing team members to work simultaneously without the risk of overwrites. Using Git branches, we can also implement features from the sprint backlog and automate CI/CD pipelines.

4.8 Project Management and Communication Tools

To effectively manage the project and task allocation while staying organized, we are using a Trello Kanban board, which allows us to assign and monitor tasks and manage timelines effectively. We are also using a Teams group to schedule meetings, post summaries and takeaways, while also displaying crucial deadlines from the Gantt chart.

As for communication and collaboration among team members, we are using WhatsApp as our primary communication tool. This allows for quick messaging, updates, and seamless coordination in real-time. Among the many uses listed above, Microsoft Teams also allows us to communicate with the line manager effectively and professionally, reducing the risk of miscommunication and mismanagement.

Part 5

This section presents the estimated budget for developing our web application. The budget will cover all the stages of this project, which include development, project testing, evaluation, and deployment. These costs will primarily be divided into Software, Hardware, Payroll, and Office Expenses, with each of these categories being explained based on the realistic market estimates. The figures are intended to reflect a practical cost structure if the system were to be developed and implemented (Appello, 2024).

5.1 Software Breakdown

Category	Service	Cost (AED)	Notes
Backend Services (Cloud Hosting)	AWS EC2 (Amazon Web Services)	50-100/month	Used for hosting backend and databases

Software Tools	Visual Studio Code	Free	Open-source development
	Adobe Creative Cloud	262.50/month	Graphic Designing
	Figma Professional (full seat)	60/month	Used for UI/UX design and prototyping
Version control (CI tools)	GitHub Pro	12/month	Source control and integration
Testing and Deployment Tools	Google Play Developer	100	Android App Development
	Apple App Store Developer Program	370/year	IOS app Development
Estimated **Software costs** for a project of this nature come up to: **~AED 3100**			

5.2 Hardware Breakdown

Item	Quantity	Unit Price (AED)	Total (AED)
Developer Laptops (mid-range)	8	~4000	~32000
Smartwatch (testing)	2	~1200	~2500
Heart Monitors (testing)	2	~450	~900
Smartphone (Watch compatible)	2	~2000	~4500
Local Server	1	~20000	~20000
Wireless Headsets	8	~200	~2000
Miscellaneous	-	~2000	~2000

(Adapters, Chargers and Cables, Setting up Costs)			
Estimated **Hardware costs** for a small team of 8 come up to: **~AED 64,000**			

5.3 Payroll and Office Breakdown

5.3.1 Payroll Table

Employee Name	Role	Hourly Rate (AED)	Hours Assigned/Monthly	Monthly Salary (AED)
Amr Mohamed	Project Manager, Technical Lead	85	160	13,600
Fazila	UI/UX Designer	70	160	11,200
Adnan	UI/UX Designer	70	160	11,200
Raees Sudheer	Frontend Engineer	75	160	12,000
Geona	Frontend Engineer	75	160	12,000
Amin	Back-end Engineer	80	160	12,800
Uzair	Quality Assurance Engineer	65	160	10,400
Aadhi	Quality Assurance Engineer	65	160	10,400
The **Subtotal Monthly for payroll** will be **AED 93,600**				

5.3.2 Overhead Office Costs Breakdown

Category	Description	Approximate Cost/Monthly (AED)
Internet and Office Utilities	Internet costs, electricity bills,	1500
Workspace	Shared access workspace	5000
Administrative and Miscellaneous	Documentation, Printing	500
The Total for **Overhead Office costs** will come up to **~AED 7000**		

5.4 Total Costs

- **Software breakdown** enlists all the software services we will be using throughout this project: **~ AED 3100**
- **Hardware breakdown** shows all the one-time hardware costs the company will have during its Development and Testing: **~ AED 64,000**
- **Payroll Costs** shows how much staff costs add up to over the course of this six-month project: **~ AED 562,000**
- **Overhead Office Costs** breaks down estimates for office and utility costs: **~ AED 42,000**

The above estimated costs for developing and testing this WebApp over a **six-month period** amount to approximately **AED 672,000**. This amount includes all the expenses involved in software, hardware, payroll, and office operations (Sholiq et al., 2016).

Part 6

6.1 Overview and Objectives

The purpose of this study is to evaluate the usability of the early prototype of the PulseLink health application. The findings from this study will help us further improve the health application before the final product

The usability evaluation will be conducted by using step-by-step tasks to be completed by the participants and noting down the outcome of

whether they could complete the tasks with assistance or could not because of bugs. After this, they will be presented with a post-test questionnaire using Likert scale questions, Quantitative data such as task completion rate, assistance required, and the post-test questionnaire Likert scale questions, will be collected to measure the clarity of the prototype

6.2 Experiment design

6.2.1 Test plan

The test was conducted with a small group of participants to gather feedback and identify user experience and usability. We evaluated the participant's ability to complete the task scenarios step by step without any assistance, identifying any layout or navigation issues while also gathering quantitative data for further improvements to the application.

Participants were observed while they completed the tasks and noted down their completion time, assistance required, and their overall satisfaction with the prototype. The data collected is mostly quantitative, focusing on task performance and post-test ratings. This study followed a within-subjects design with a total of 10 participants completing the same tasks for both the user view and professional view of the application

6.2.2 Design

Both the User View and the Professional View of the PulseLink prototype were carefully designed to support their intended audiences. Both views have the same overall design and style, but they differ in navigation and features.

In figure 8, the dashboard, settings, devices, and companion are some of the main screens that make up the user’s view. Users can freely navigate between pages in a nonlinear way thanks to the sidebar navigation. This layout is consistent across screens and supports exploration.

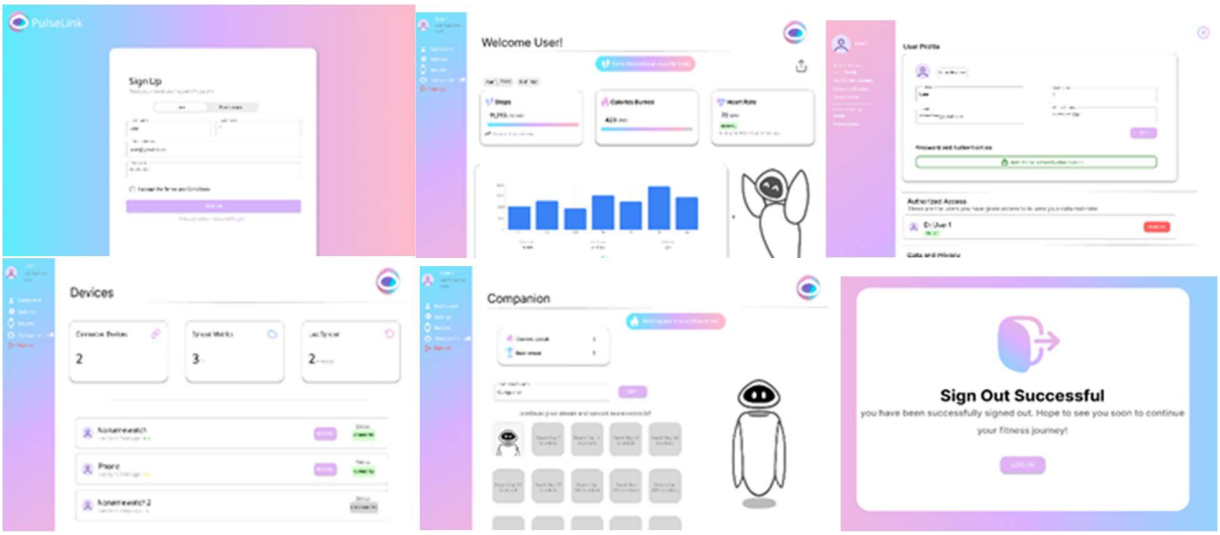


Figure 8 - User Prototype Design

The professional view on the other hand, represented in figure 9, concentrates on data review and client monitoring. Its structure is more simplified, presenting important information like client details, alert and synced health information, while maintaining a consistent look as the user view. The doctors can easily navigate the client data and go back to the main dashboard, resulting in a smooth process that is ideal for fast data checks.

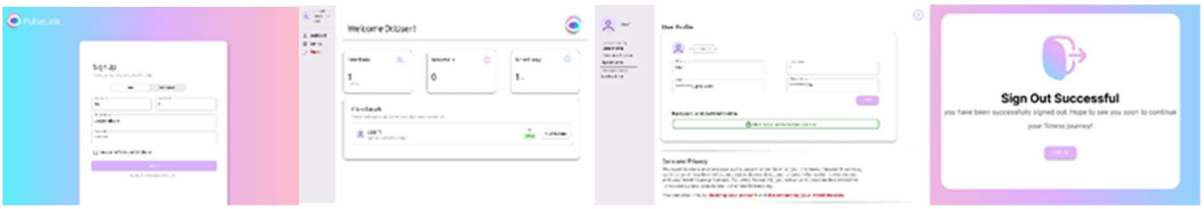


Figure 9 - Healthcare Professional Prototype Design

Mixed initiative interaction is supported by both views; the user can freely navigate the interface while the system automatically updates health information and redirects to the appropriate screen when needed. This makes sure that both the regular users and medical professionals can achieve their objectives with the least amount of work. Note that higher quality versions of both prototypes are included in the Appendices.

6.2.3 Task scenarios

Participants were asked to complete a structured set of tasks for each interface. These tasks were common actions which a real user would

typically interact with daily. Both user and professional scenarios are included in the Appendices.

Each prototype used forward navigation which allows participants to proceed at their own pace. Observers noted the task performance and recorded the bugs or parts where assistance was required.

6.3 Findings and Outcomes

There were pre-test and post-test questionnaires provided during the evaluation. Pre-test helped us gather background information on the participants, while post-test questionnaires allowed us to gather the quantitative data used during the analysis. Note that all questionnaires used for the evaluation will be included in the Appendices.

6.3.1 Pre-test Questionnaire Findings

80% of the participants were aged between 18-23, with a small number of older participants as well. The group had 6 male participants, two female participants and two who preferred not to say. Our participants had a wide range of experience, with 50% having used a health app before. Only 30% of participants used a virtual companion before, which reflects the larger population, as most companies are yet to implement the technology in their applications.

All the participants knew how to use a laptop or a touchpad. This ensured that basic device skills would not affect their ability to complete the usability tasks.

6.3.2 Task Scenario Results

All ten participants were able to complete the required task step by step for both the user view and professional view. Half of the participants started from the user view while the other half from the professional view. 90% of the participants completed every step without assistance showing that the navigation and layout were easy to understand and complemented the color palette. 10% of participants needed minor assistance, usually trying to go back to the dashboard or clicking some buttons that were not implemented. Despite this, the completion rate of approximately 90% without any assistance showcasing that the navigation flow of the prototype was concise and functional. After this, we conducted a post-questionnaire to get their feedback on the prototype.

6.3.3 Post-Test Questionnaire Findings

After the participants completed the task scenarios, we conducted a post-questionnaire to get feedback on the prototype using Likert scale questions. Results are shown in the table below:

Questions	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
The layout was clear and easy to understand	5	5	0	0	0
Completing the tasks on the website felt easy	6	4	0	0	0
It was easy to navigate between different screens	4	5	1	0	0
The app colors were not pleasing	0	0	1	3	6
It was easy to find and open various menus	4	5	1	0	0
The health information provided was clear and easy to understand	7	2	1	0	0
The information felt relevant and helpful	5	4	1	0	0
It took me long to complete the task	0	1	4	2	3
The text and visual were easy to understand	5	4	1	0	0
Overall, I am satisfied with the prototype	6	4	0	0	0

The results show that the participants responded very positively to the prototype. 95% of the users think that the layout was clear, the tasks were easy to complete and navigation between different screens was smooth. 95% of the participants also stated that the text and visuals

representing health information such as calories, steps, heart rate, etc., were appealing and organized. Overall satisfaction was high, but 40% of participants did take a considerable amount of time to complete the task, which can negatively affect their experience.

6.3.4 Improvements and Next Steps

For this evaluation, we will address the results regarding task completion time by simplifying the interface design and reducing clutter. Another concern was the color scheme which did not impact the usability, but more themes need to be introduced to improve overall satisfaction with the product. To continuously improve the interface, task flow, and visual design, we will routinely conduct usability testing between major sprint sessions to ensure the development remains user centered.

Conclusion

PulseLink is a Virtual Health Companion to support users in their daily lives with many features. For example, there will be a dashboard for users to view their metrics, with a built-in Mascot you can chat with, as well as a dashboard for professionals to view patient data, in app or exported, with an alert system for monitoring patients long-distance. The team will develop the web application using the Agile Scrum software process, to manage risks and remain focused and user driven. Usability evaluations are also necessary and were conducted both for the prototype, and future versions with new major features. Overall, the work completed in this stage sets a clear framework of requirements, development process, and design, ensuring the project is technically feasible and aligned with user feedback. This provides a solid foundation for the next stage, where implementation and further testing will move us closer to delivering the PulseLink web application.

References

- Appello. (2024) Healthcare App Development Costs: Key Features & Pricing Insights. Found at:https://appello.com.au/articles/healthcare-app-development-cost
- Sholiq, S., Wahyuni, T. and Rochman, F.(2016)A model to determine cost estimation for software development projects of small and medium scale. Journal of Theoretical and Applied Information Technology (JATIT), Vol.88 No.2, pp.330-337. Found at:https://repository.dinamika.ac.id/id/eprint/6773/1/Artikel%20JATIT%2012.pdf
- Bannerman, P. L., 2008. 'Risk and risk management in software projects,' Information and Software Technology, 50(11-12), pp. 1197-1216. Available at: https://www.sciencedirect.com/science/article/abs/pii/S0164121208000897
- Sarigiannidis, L. and Chatzoglou, P. D., 2011. 'Software development project risk management: a new conceptual framework', Journal of Software Engineering and Applications, 4, pp. 293-305. Available at: https://www.researchgate.net/publication/220204290_Software_Development_Project_Risk_Management_A_New_Conceptual_Framework
- European Union, 2016. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation). Available at: https://gdpr-info.eu/
- Ahmed, R., 2022. 'Exploring the relationship between scope creep, project success and project complexity,' International Journal of Project Management, Article S2666-7215(22)00024-2. Available at: https://www.sciencedirect.com/science/article/pii/S2666721522000242
- Kalluri, R., 2022. 'A Human Factors Study of Risk Management of Complex Agile Scrum Projects in Large Enterprises,' EMSE Publications, Old Dominion University Digital Commons. Available at:https://digitalcommons.odu.edu/cgi/viewcontent.cgi?article=1144&context=emse_fac_pubs
- Bisikirskienė, L., 2025. 'Software project risk identification based on Scrum artefacts', Applied Sciences, 15(2), 824. Available at: https://www.mdpi.com/2076-3417/15/2/824

Appendices

Sequence Diagrams

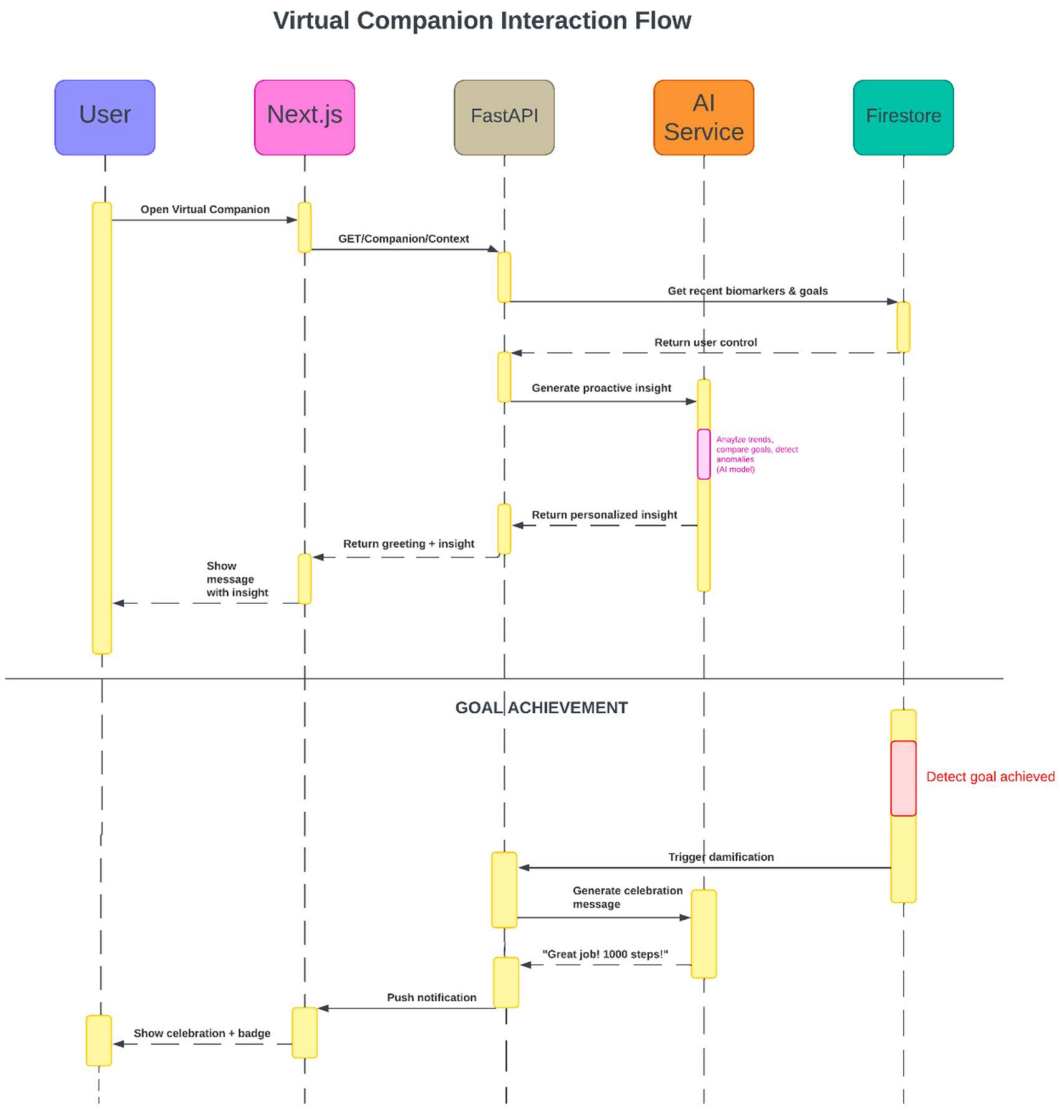


Figure 10 – Virtual Companion Sequence Diagram

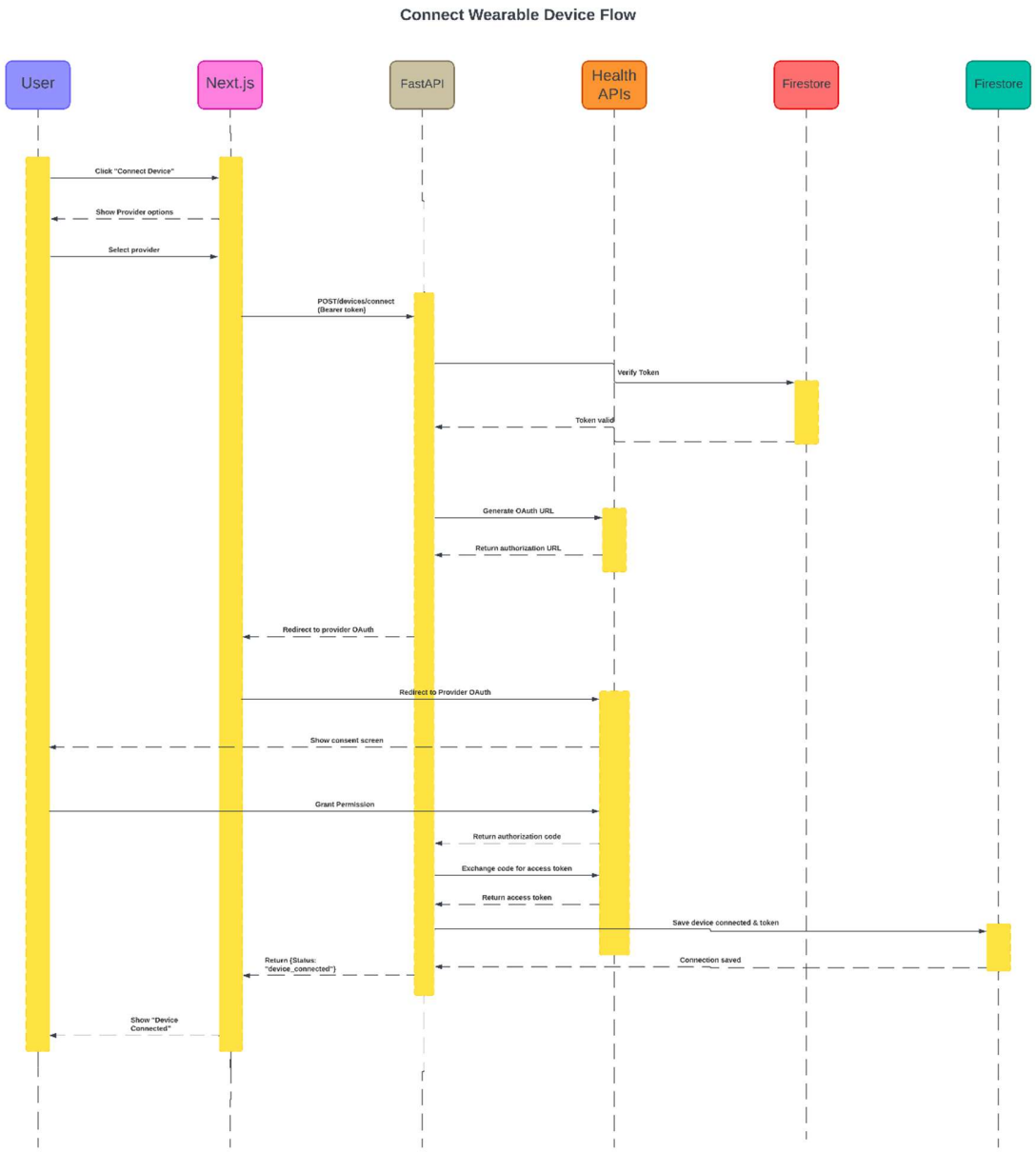


Figure 11 – Connecting Device Sequence Diagram

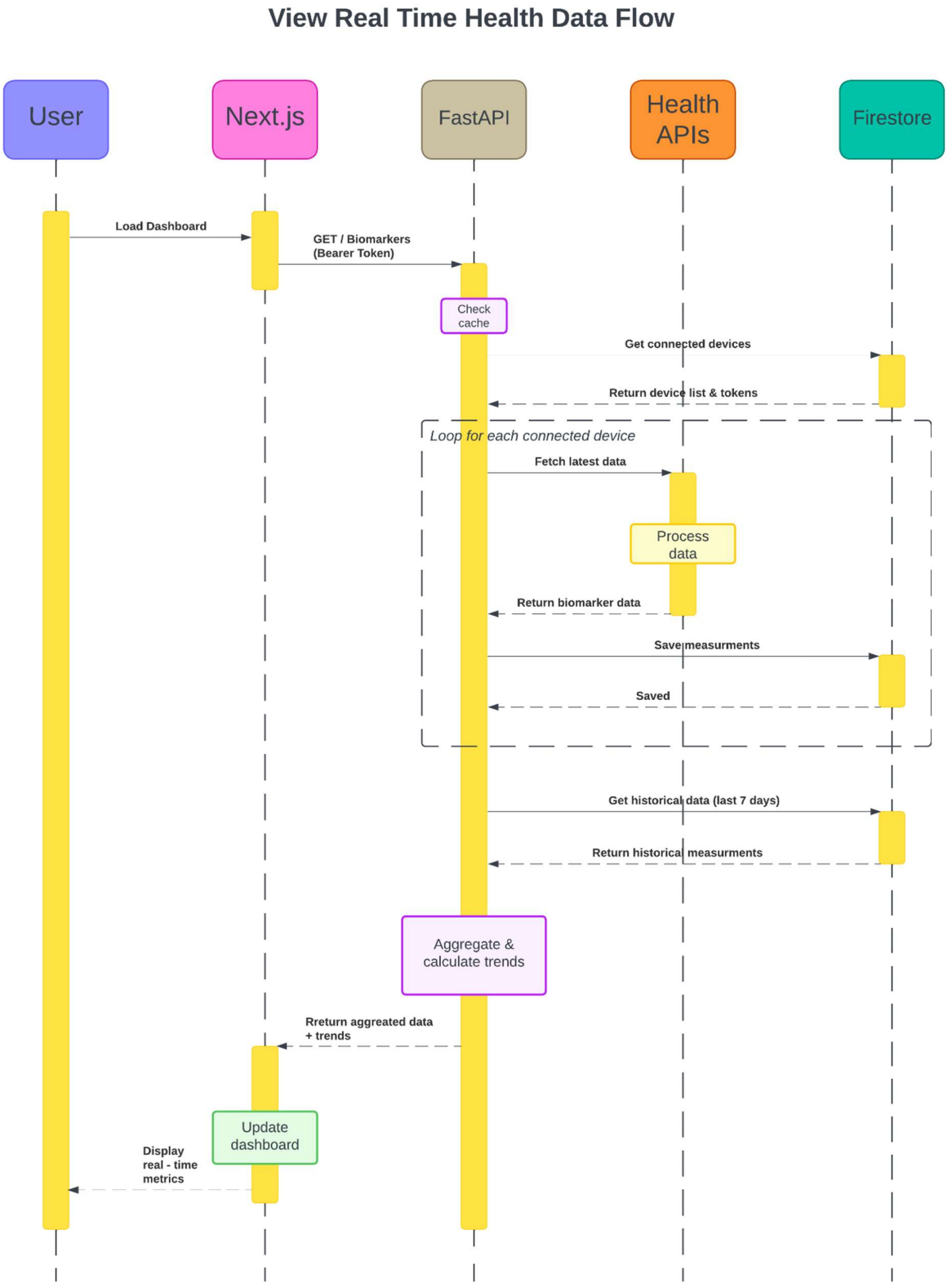


Figure 12 – Real Time Data Sequence Diagram

Use-Case Specifications

Use Case: SIGN UP

ID: *UC-01*

Goal: *To create an account*

Primary actor: Stakeholder

Secondary actor(s): *System*

Preconditions:

1. *The user does not have an existing account.*
2. The user has an email address that they can use to register.
3. The user has a stable internet connection.

Postconditions:

1. *New account is created with appropriate role.*
2. *The account credentials are stored securely.*

Main flow:

1. *Stakeholder is on the registration page.*
2. *The system displays the required input fields.*
3. *The stakeholder selects an account type (user/healthcare professional).*
4. *The stakeholder inputs the name, email, and password.*
5. *The stakeholder accepts the terms of service and privacy policy check box.*
6. *The Stakeholder clicks the sign-up button.*
7. *The system validates and verifies all the stakeholder's input.*
8. *The system validates that the user has accepted the terms of service and privacy policy check box.*
9. *The system creates an account for the stakeholder.*
10. *System displays a success message.*
11. *System Redirects the stakeholder to the log in page.*

Alternative flows:

7a. Invalid or Incomplete Data

1. *During validation, the system detects missing or invalid fields.*
2. *The system displays an error message indicating which fields need to be corrected.*
3. *The flow returns to step 3 for the user to edit their input.*

7b. Email Already Registered

1. *During validation, the system detects the provided email is already associated with an existing account.*
2. *The system displays an error message that the account already exists.*

3. *The system provides an option to log in or to use a different email address.*
4. *The flow returns to step 4 for the user to make a new selection.*
5. *8a. Terms of Service Not Accepted*
6. *During validation, the system detects the terms of service checkbox is not checked.*
7. *The system displays an error message stating that the terms must be accepted.*
8. *The flow returns to step 5 for the user to accept the terms.*

Use Case: LOGIN

ID: UC-02

Goal: *Login to PulseLink*

Primary actor: *Stakeholder*

Secondary actor(s): *System, MFA pass , MFA fail*

Preconditions:

1. *Stakeholder is not currently logged in.*

Postconditions:

1. *Stakeholder's consent preferences are updated*

Main flow:

1. *The stakeholder goes to the login page.*
2. *The System displays the login interface with input fields.*
3. *The Stakeholder enters the email address and password.*
4. *The Stakeholder clicks the login button.*
5. *The System validates if the fields are filled.*
6. *The system sends the credentials to the server for authentication.*
7. *The system verifies the credentials and the account exists.*
8. *The System Redirects to MFA verification*
   - *System sends a verification code to the stakeholder's MFA method*
   - *Stakeholder enters the verification code*
   - *System validates the code*
   - *MFA pass*
9. *System retrieves the stakeholder profile data.*
10. *The system redirects stakeholder to the dashboard.*
11. *The system displays personalized welcome message.*
12. *The systems load the dashboard data.*

Alternative flows:

5a. Empty Fields

1. *During validation, the system detects the email or password field is empty.*
2. *The system displays an error message: "Email and password are required".*
3. *The flow returns to step 3 for the user to enter their credentials.*

7a. Invalid Credentials

1. *The system detects that the email and password combination is incorrect.*
2. *The system displays an error message: "Invalid email or password".*

7b. Account Not Found

1. *The system cannot find any account associated with the provided email address.*
2. *The system displays a dialog stating, "Account not found".*
3. *The system offers the choice to "Try again" or "Sign up".*

3a. Stakeholder chooses "Try again".

1. *The flow returns to step 3 for the user to re-enter their credentials.*

3b. Stakeholder chooses "Sign up".

1. *The user is redirected to the Sign-Up page (UC-01).*

8a. MFA Fails

1. *During MFA verification, the user enters an invalid code.*
2. *The system displays an "Invalid code" message.*
3. *The user is prompted to enter the code again.*

Use Case: PROFILE UPDATE

ID: *UC-03*

Goal: To update user profile information

Primary actor: Stakeholder

Secondary actor(s): *System*

Preconditions:

1. The stakeholder is logged into the System.
2. The User is in the settings page

Postconditions:

1. *The user profile information is updated in the database.*
2. The changes are reflected across the system

Main flow:

1. *Stakeholder navigates to the profile settings section.*
2. *System displays the current profile information.*
3. *The stakeholder modifies the desired fields (name, email, phone)*
4. *The stakeholder submits updated information.*
5. *The system updates profile in database*
6. *The system refreshes the profile display with latest information.*

Alternative flows:

4a. Invalid Data Format

1. *Upon submission, the system detects that one or more fields have an invalid data format.*
2. *The system displays an error message: "Invalid data format."*
3. *The flow returns to step 3 for the user to correct the information.*

4b. Email Already In Use

1. *Upon submission, the system detects the new email is already registered to a different account.*
2. *The system displays an error message: "Email already in use."*
3. *The flow returns to step 3 for the user to enter a different email.*

Use Case: DELETION OF ACCOUNT AND DATA

ID: UC-04

Goal: *To permanently delete all user account and personal data*

Primary actor: *Stakeholder*

Secondary actor(s): *System*

Preconditions:

1. *Stakeholder is logged into the system.*

Postconditions:

1. Stakeholder's account is permanently deleted from the database.
2. All personal data has been erased.
3. User session is terminated.

Main flow:

1. *the stakeholder navigates to settings page.*
2. *The stakeholder clicks on "Delete account" option.*
3. *The system displays warning about permanent deletion of data and account.*
4. *The Stakeholder confirms deletion.*
5. *The system initiates deletion of data and account details*
6. *System logs out the stakeholder.*
7. *The system displays account deletion confirmation.*

Alternative flows:

4a. Stakeholder Cancels Deletion

1. *Instead of confirming, the stakeholder selects the "Cancel" option.*
2. *The system returns the user to the settings page (step 1).*

5a. Error During Deletion

1. *The system encounters an error while trying to delete the account data from the database.*
2. *The system displays an error message: "An error occurred while deleting your account. Please try again."*
3. *The user remains on the settings page.*

Use Case: CONSENT TO DATA

ID: *UC-05*

Goal: *To allow the user to manage permissions for collecting, using, and sharing their personal and health data.*

Primary actor: *Stakeholders*

Secondary actor(s): *Server, System*

Preconditions:

1. *Stakeholder is logged into the system.*

Postconditions:

1. *Stakeholder's account is permanently deleted from the database.*
2. *All personal data has been erased*
3. *User session is terminated.*

Main flow:

1. The user navigates to the "Consent Management" or "Data Sharing" section in the account settings.
2. The system displays a list of data categories.
3. The system also displays a list of parties that can be granted access.
4. The user selects a data category or a party to manage consent.
5. The user grants or revokes access for specific data categories to specific parties using clear toggle switches or checkboxes.
6. The user saves their consent preferences.
7. The system updates the user's consent settings in the database.
8. The system displays a confirmation message that the settings have been saved.

Alternative flows:

6a. User revokes consent required for a core feature.

1. When the user attempts to save, the system detects that revoking consent will disable a feature (revoking biometric data access will prevent the dashboard from functioning).
2. The system displays a warning message explaining the consequences of this change and asks for confirmation.
3. If the user confirms, the changes are saved. If not, the flow returns to step 5.

6b. User cancels changes.

1. Instead of saving, the user selects the "Cancel" option.
2. The system discards all changes and restores the consent settings to their last saved state.

Use Case: RESET PASSWORD

ID: *UC-6*

Goal: To allow stakeholders to create a new password when they cannot access their account

Primary actor: *Stakeholder*

Secondary actor(s): *System, Email*

Preconditions:

1. *The stakeholders have a registered account in the server.*
2. *The stakeholder has access to the registered email*

Postconditions:

1. The stakeholder password is updated in the server.

Main flow:

1. The stakeholder goes to the login page.
2. The stakeholder selects forgot password link.
3. The system displays the reset password page with the email input field.
4. The stakeholder inputs the email of their registered account.
5. The stakeholder submits the request.
6. System validates the email exists in the server.
7. The system generates and sends a reset link to the registered email.
8. The system displays new password page with the valid input fields.
9. The stakeholder enters and confirms new password.
10. System validates the password to meet the requirements.
11. The system updates the password in the server.
12. The system displays a success message

Alternative flows:

6a. Email Not Found

1. *During validation, the system detects that the provided email does not exist in the database.*
2. *The system displays an error message: "Email not found. Please try a different email."*
3. *The flow returns to step 4 for the user to re-enter their email.*

10a. Password Requirements Not Met

1. *During validation, the system detects the new password does not meet the security requirements.*
2. *The system displays a warning message listing the password requirements that were not met.*
3. *The flow returns to step 9 for the user to enter a new password.*

Use Case: LOGOUT

ID: *UC-7*

Goal: *The user can securely log out of their account.*

Primary actor: *Stakeholder*

Secondary actor(s): *Server , System*

Preconditions:

1. *The user is authenticated and logged in.*
2. *The system is connected securely.*

Postconditions:

1. *The user is logged out.*
2. *The user is redirected to the login page.*
3. *Logout details are logged.*

Main flow:

1. *The user selects “Log out” from the dashboard.*
2. *System sends a logout request to the server.*
3. *Server terminates the session.*
4. *System confirms that the logout is successful.*
5. *System displays the login screen.*

Alternative flows:

3a. Session termination fails

1. *The server fails to terminate the session and returns an error.*
2. *The system displays a "Logout failed, please try again" message.*
3. *The user remains logged in and on their current screen.*

Use Case: VIEW DASHBOARD

ID: *UC-8*

Goal: *To display real-time data, health graphs, notifications, and the system features*

Primary actor: *Stakeholder*

Secondary actor(s): *Server, System, Virtual companion, wearable APIs*

Preconditions:

1. *The user is logged in*
2. *The user has granted consent to use the data*
3. *System is connected to the required APIs*

Postconditions:

1. *Dashboard data is retrieved and displayed*
2. *Real-time biomarker data is displayed*
3. *The user can access all the dashboard features*

Main flow:

1. *The user opens the dashboard.*
2. *System requests the biomarker data from the server.*
3. *The server retrieves all the required data and shows it on the dashboard.*

4. *System retrieves the virtual companions' insights and wearable APIs data and displays them.*
5. *The system displays the dashboard with all the collected data.*
6. *The user can navigate to other functions of the dashboard.*

Alternative flows:

3a. No data exists

1. *The server finds that no biomarker or device data is available for the user.*
2. *The system displays a default or empty dashboard with a message indicating no data is present.*

4a. Failure to connect to companion or wearable APIs

1. *One or more of the virtual companion or wearable APIs fails to return data.*
2. *The system displays the dashboard with partial data and shows a warning for the missing information.*

Use Case: FETCH DATA
ID: *UC-9*
Goal: *Allows the system to retrieve real-time health data from connected devices and APIs.*
Primary actor: *System*
Secondary actor(s): *Stakeholder, Wearable APIs, Server*
Preconditions: 1. *The database is working.* 2. *The user has connected at least one device.* 3. *User has granted consent to use data.*
Postconditions: 1. *The data is processed and stored in the database.* 2. *Dashboard shows the updated data.* 3. *Any error is logged.*
Main flow: 1. *The system sends a request to fetch data from the connected devices.* 2. *The API receives the request and returns the biomarker data* 3. *System verifies the received data.* 4. *The system ensures the data received matches with biomarker data.* 5. *System stores the data securely in the database.* 6. *System updates in real-time and displays the data in the dashboard.*
Alternative flows: *4a. Actor cancels customization* 1. *Instead of saving, the actor exits the customization view.* 2. *The dashboard layout remains unchanged from its previous state.* *6a. Server fails to save the settings* 1. *The server is unable to store the new preferences and returns an error.*

2. *The system displays an "Unable to save changes" message.*
3. *The user is returned to the customization screen (step 3) to try again.*

Use Case: NOTIFICATIONS
ID: *UC-10*
Goal: *The system sends actors alerts, summaries, and personalized health recommendations.*
Primary actor: *Stakeholder*
Secondary actor(s): *Server, Virtual companion API*
Preconditions: 1. *The user is logged in.* 2. *The system is connected to the device and other wearables.*
Postconditions: 1. *All notifications are shown to the actor.* 2. *Notification status is updated with read/unread status.*
Main flow: 1. *System receives the notification data from wearables, Virtual companion, and their healthcare professionals.* 2. *The system categorizes the notifications.* 3. *The user opens the notification panel.* 4. *The system displays all the notifications chronologically* 5. *The user can read, dismiss, or customize notification preferences*
Alternative flows: *1a. API failure while receiving data* 1. *The system fails to receive notification data from one or more sources.* 2. *The system displays a warning message in the notification panel indicating that data could not be fetched.*

Use Case: CUSTOMIZE DASHBOARD

ID: *UC-11*

Goal: *To allow the user to personalize their dashboard view.*

Primary actor: *Stakeholder*

Secondary actor(s): *System*

Preconditions:

1. *The user is logged into the system.*
2. *The user is on the dashboard page.*

Postconditions:

1. *The user's dashboard layout and displayed metrics are updated and saved.*

Main flow:

1. *The user selects the option to customize the dashboard.*
2. *The system presents the available customization options (e.g., layout, metrics, themes).*
3. *The user modifies the dashboard by adding, removing, or rearranging elements.*
4. *The user saves the changes.*
5. *The system updates the user's dashboard to reflect the changes.*
6. *The system confirms that the changes have been saved.*

Alternative flows:

4A: User cancels customization.

1. *At any point during the customization, the user chooses to cancel.*
2. *The system discards all changes and restores the dashboard to its last saved state.*

Use Case: RECEIVE VIRTUAL COMPANION NOTIFICATIONS
ID: *UC-12*
Goal: *To provide the user with proactive health insights, suggestions, and positive reinforcement from the AI companion.*
Primary actor: *Stakeholder*
Secondary actor(s): *System, Virtual Companion*
Preconditions: 1. *The user is logged into the system.* 2. *The user has enabled notifications from the Virtual Companion.*
Postconditions: *1. The user has received a notification from the Virtual Companion.*
Main flow: 1. *The Virtual Companion identifies a trigger for a notification (e.g., goal achievement, proactive health tip).* 2. *The system sends a notification to the user's device.* 3. *The user views the notification.*
Alternative flows: *2a: User has disabled notifications.* 1. *The system does not send the notification.*

Use Case: *SET EMERGENCY PINGS*

ID: *UC-13*

Goal: *To allow the user to set up emergency contacts who can be alerted in case of critical health events.*

Primary actor: *Stakeholder*

Secondary actor(s): *System*

Preconditions:

1. *The user is logged into the system.*

Postconditions:

1. *The user's emergency contacts are saved in the system.*

Main flow:

1. *The user navigates to the emergency contacts settings.*
2. *The user adds a new emergency contact by providing their details (e.g., name, phone number, email).*
3. *The user saves the new contact.*
4. *The system confirms that the emergency contact has been added.*

Alternative flows:

1a: User enters invalid contact details.

1. *The system detects that the provided contact details are invalid.*
2. *The system displays an error message and prompts the user to correct the details.*

3a: User removes an emergency contact.

1. *The user selects an existing emergency contact to remove.*
2. *The system asks for confirmation.*
3. *The user confirms the removal.*
4. *The system removes the contact and confirms the action.*

Use Case: *VIEW HEALTH CARE DASHBOARD*
ID: *UC-14*
Goal: *To allow a healthcare provider to view a patient's biometrics and data.*
Primary actor: *Healthcare Professional*
Secondary actor(s): *System*
Preconditions: 1. *The healthcare provider is logged into the system.* 2. *The healthcare provider has been granted access to the patient's data.*
Postconditions: 1. *The healthcare provider has viewed the patient's dashboard.*
Main flow: 1. *The healthcare provider selects a patient from their list.* 2. *The system retrieves and displays the selected patient's healthcare dashboard, including their biometrics and data.*
Alternative flows: *2a: Healthcare provider does not have access to the patient's data.* 1. *The system displays a message indicating that access is denied.*

Use Case: INTERACT WITH AI CHATBOX

ID: *UC-15*

Goal: *To allow the user to interact with the AI companion chatbot via text commands.*

Primary actor: *Stakeholder*

Secondary actor(s): *System, Virtual Companion*

Preconditions:

1. *The user is logged into the system.*

Postconditions:

1. *The user has received a response from the AI Chat Box.*

Main flow:

1. *The user opens the AI Chat Box.*
2. *The user types a message or command and sends it.*
3. *The system sends the user's message to the Virtual Companion.*
4. *The Virtual Companion processes the message and generates a response.*
5. *The system displays the Virtual Companion's response to the user in the chat interface.*

Alternative flows:

4A: The AI Chat Box does not understand the user's command.

1. *The Virtual Companion returns a message indicating that it does not understand the request.*
2. *The system displays this message to the user and may provide suggestions for valid commands.*

Use Case: CUSTOMIZE HEALTHCARE DASHBOARD

ID: *UC-16*

Goal: *To allow a healthcare provider to customize their dashboard view for a patient.*

Primary actor: *Healthcare Professional*

Secondary actor(s): *System*

Preconditions:

1. *The healthcare provider is logged into the system.*
2. *The healthcare provider is viewing a patient's dashboard.*

Postconditions:

1. *The healthcare provider's dashboard layout for the patient is updated and saved.*

Main flow:

1. *The healthcare provider selects the option to customize the dashboard.*
2. *The system presents the available customization options.*
3. *The healthcare provider modifies the dashboard layout.*
4. *The healthcare provider saves the changes.*
5. *The system updates the dashboard and confirms that the changes have been saved.*

Alternative flows:

4A: Healthcare provider cancels customization.

1. *The healthcare provider chooses to cancel the customization.*
2. *The system discards all changes and restores the dashboard to its last saved state.*

Prototype UI Design

User Interface Design

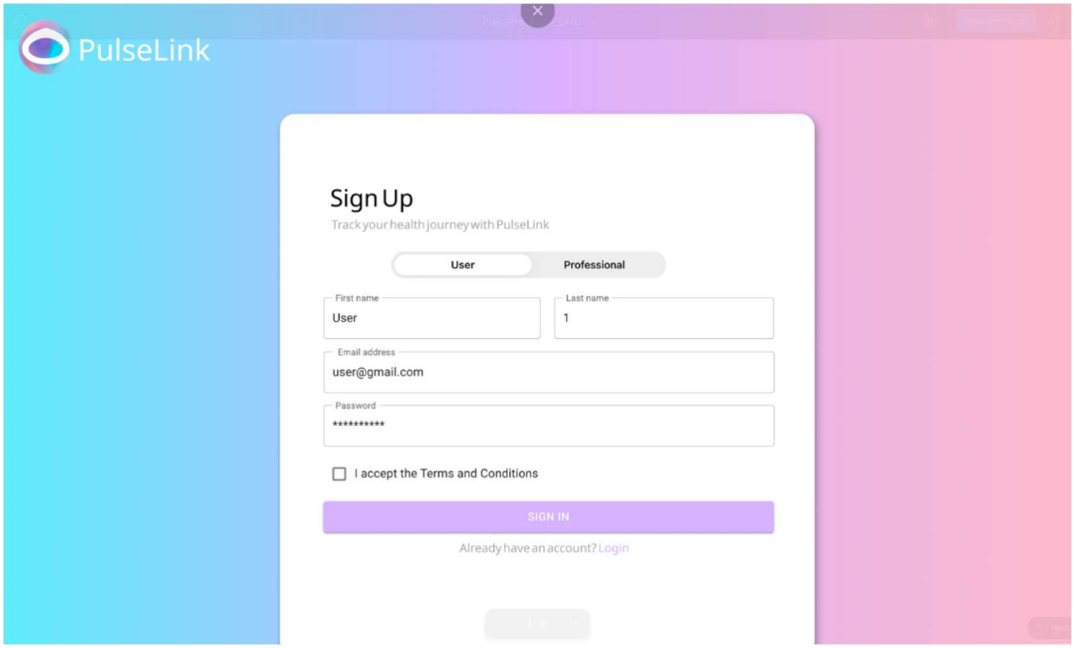


Figure 13 - Sign up page user

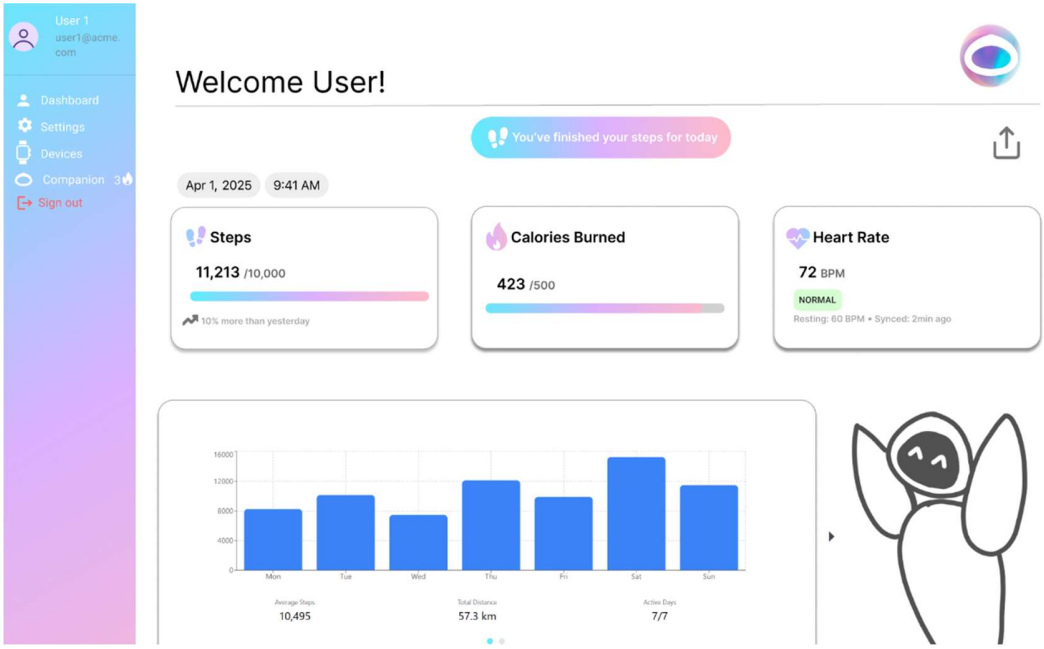


Figure 14 - Dashboard page

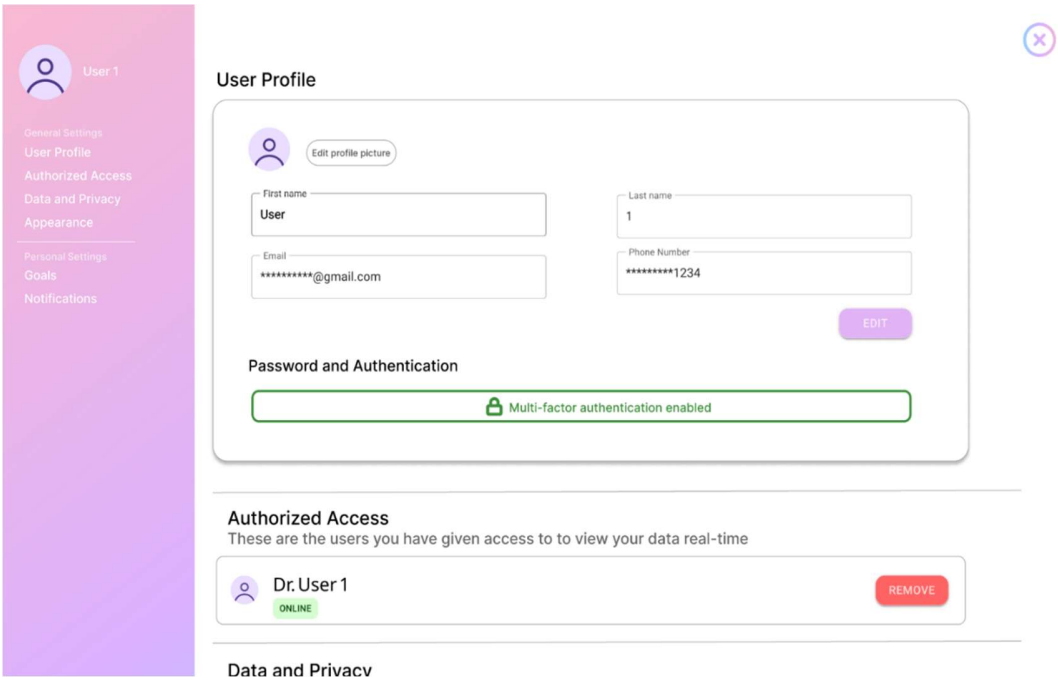


Figure 15 - Settings page

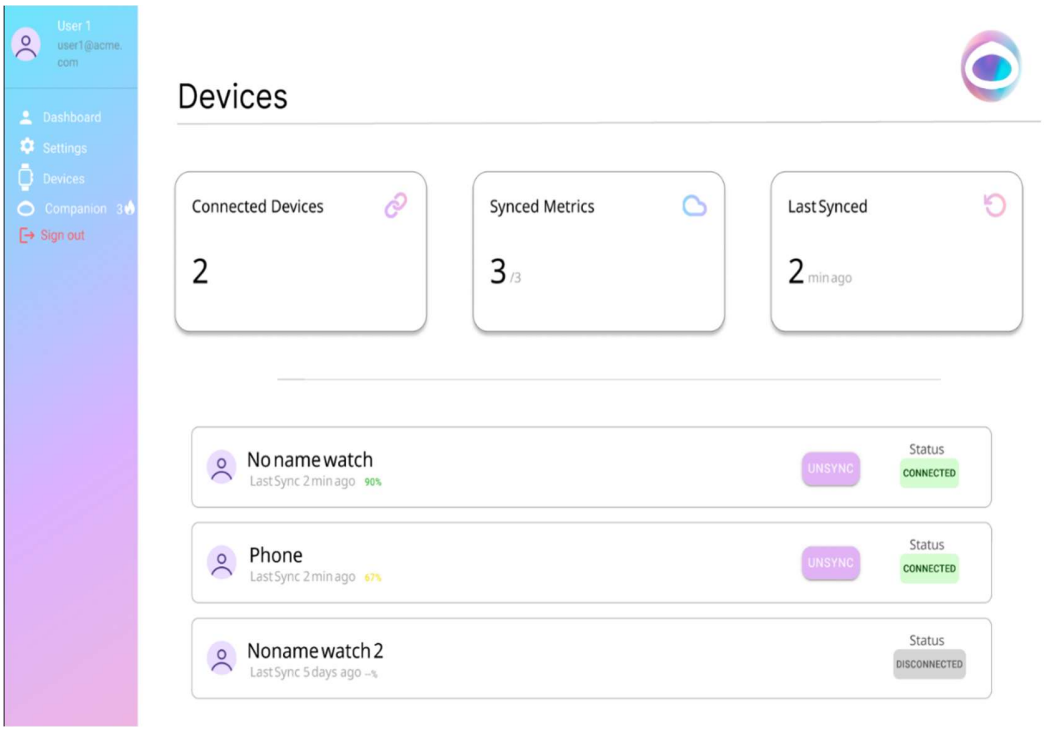


Figure 16 - Devices page

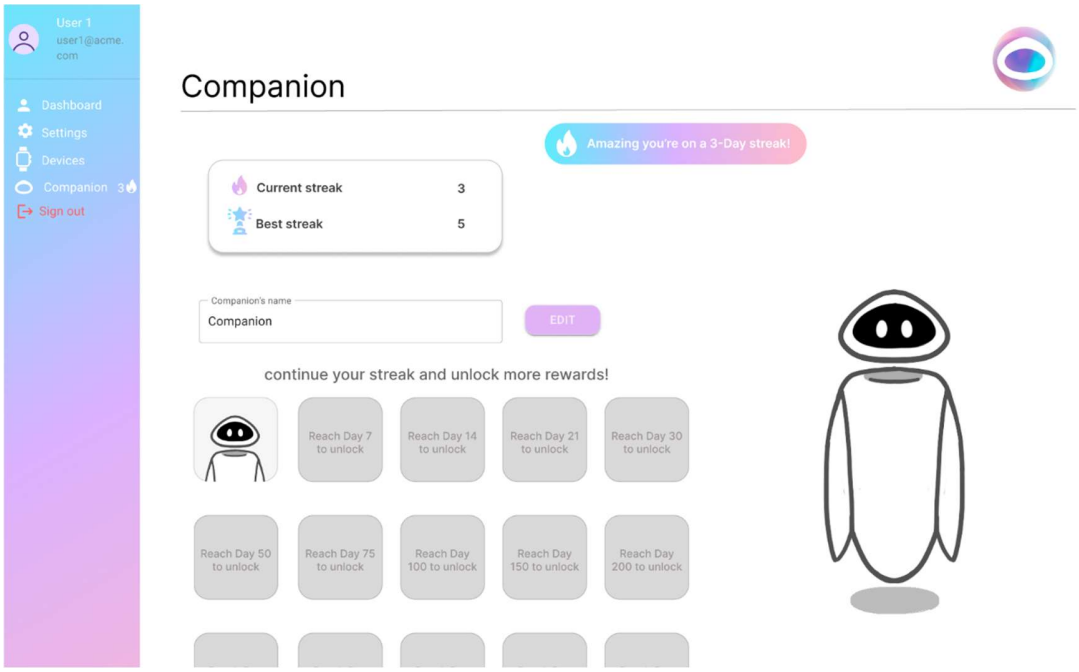


Figure 17 - Companion page

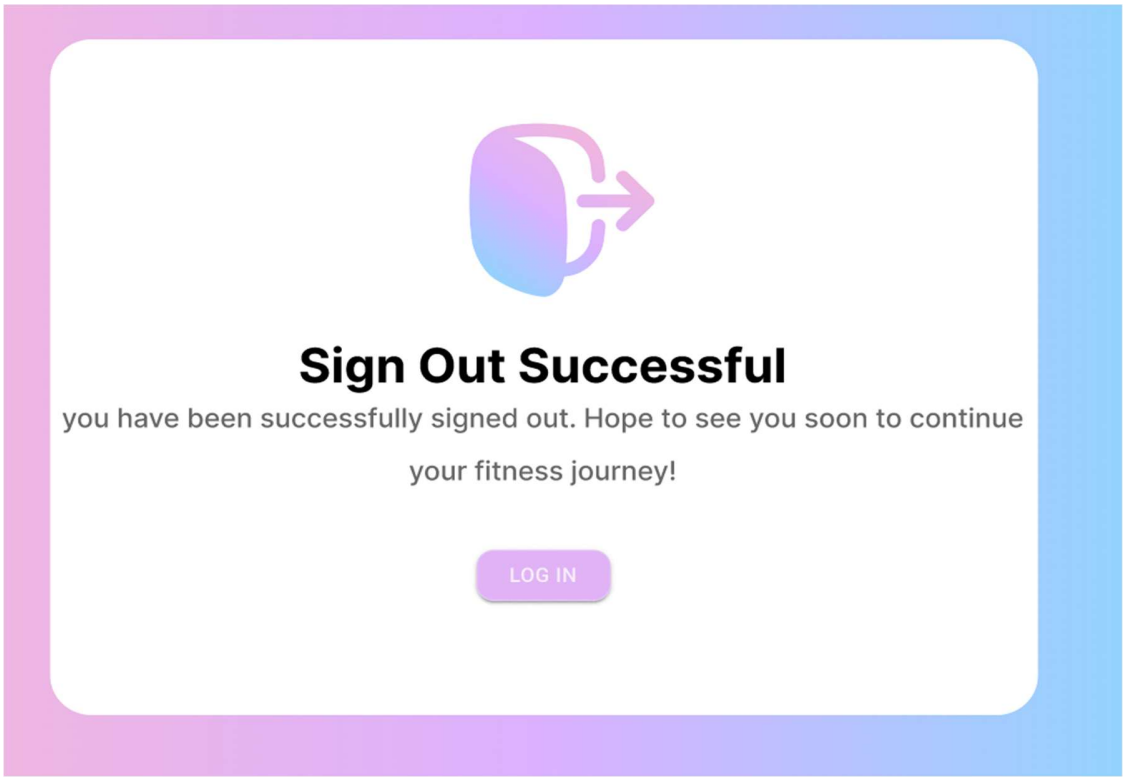


Figure 18 - Sign out page

Professional Interface Design

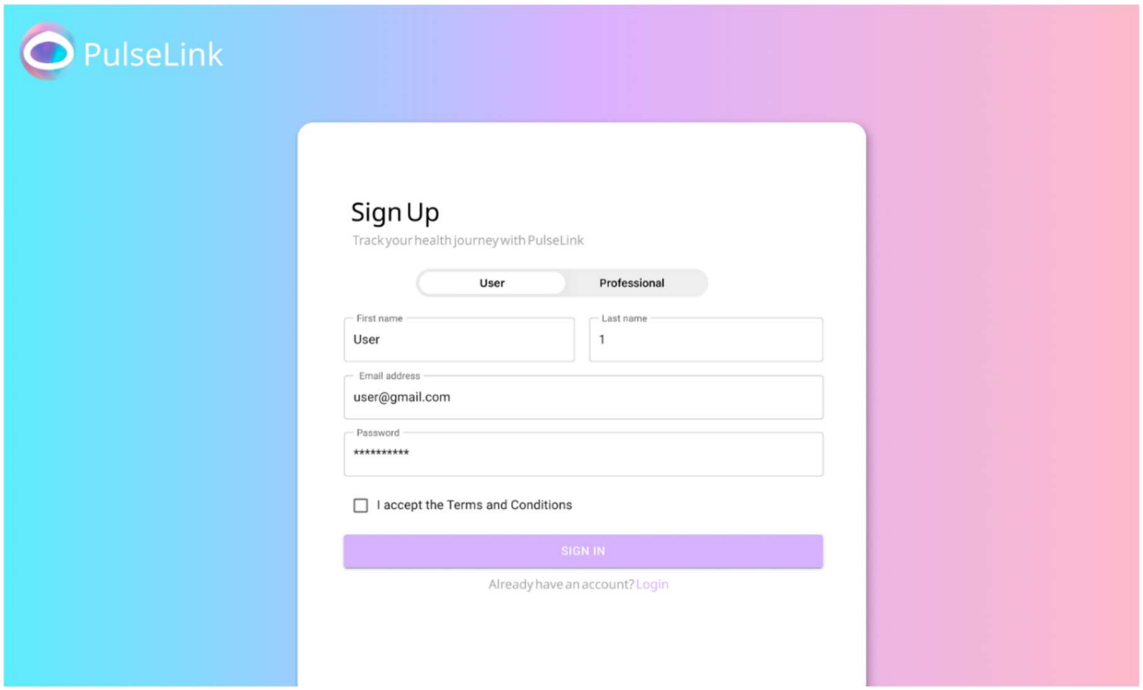


Figure 19 - Sign up page professional

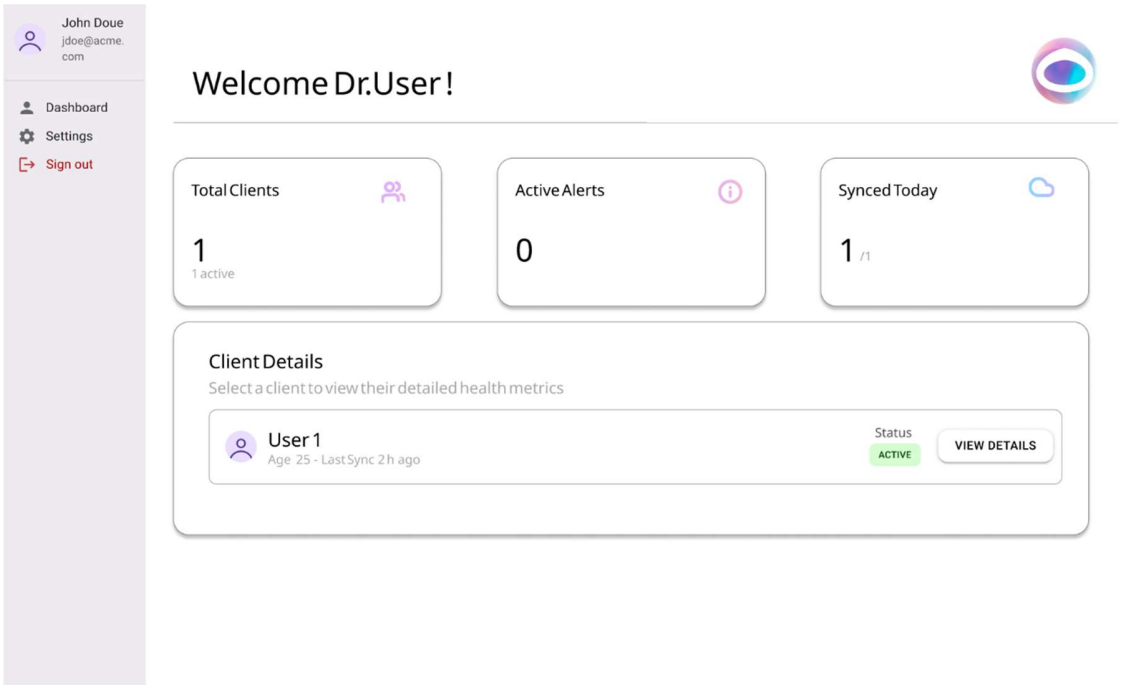


Figure 20 - Dashboard page

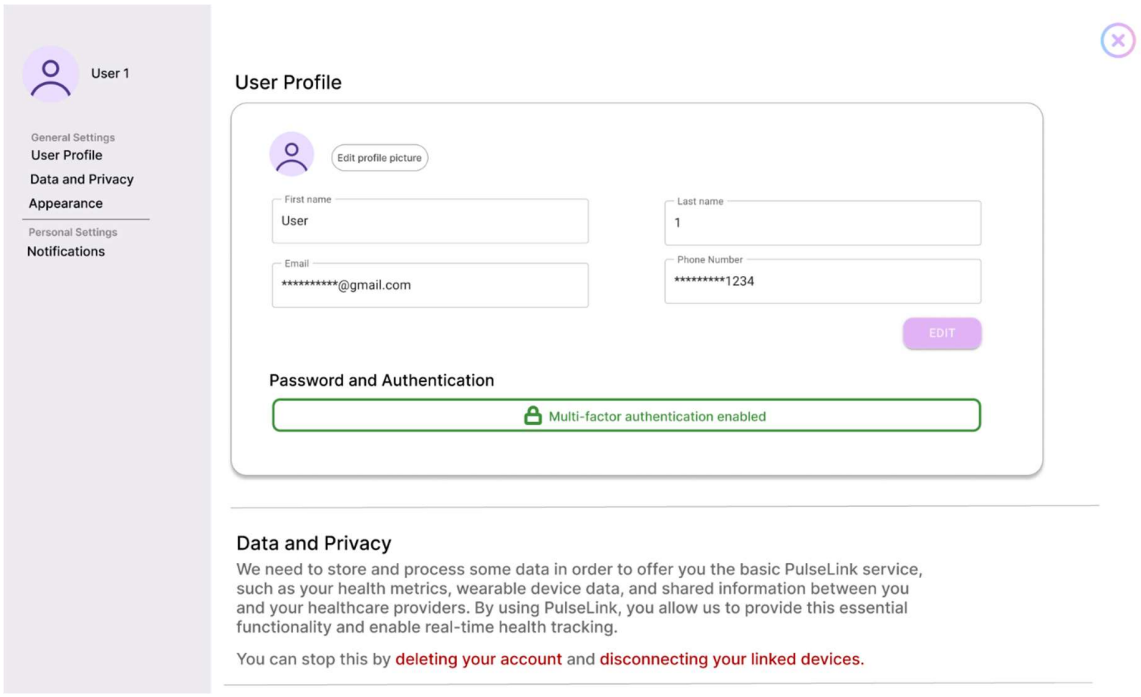


Figure 21 - Settings page

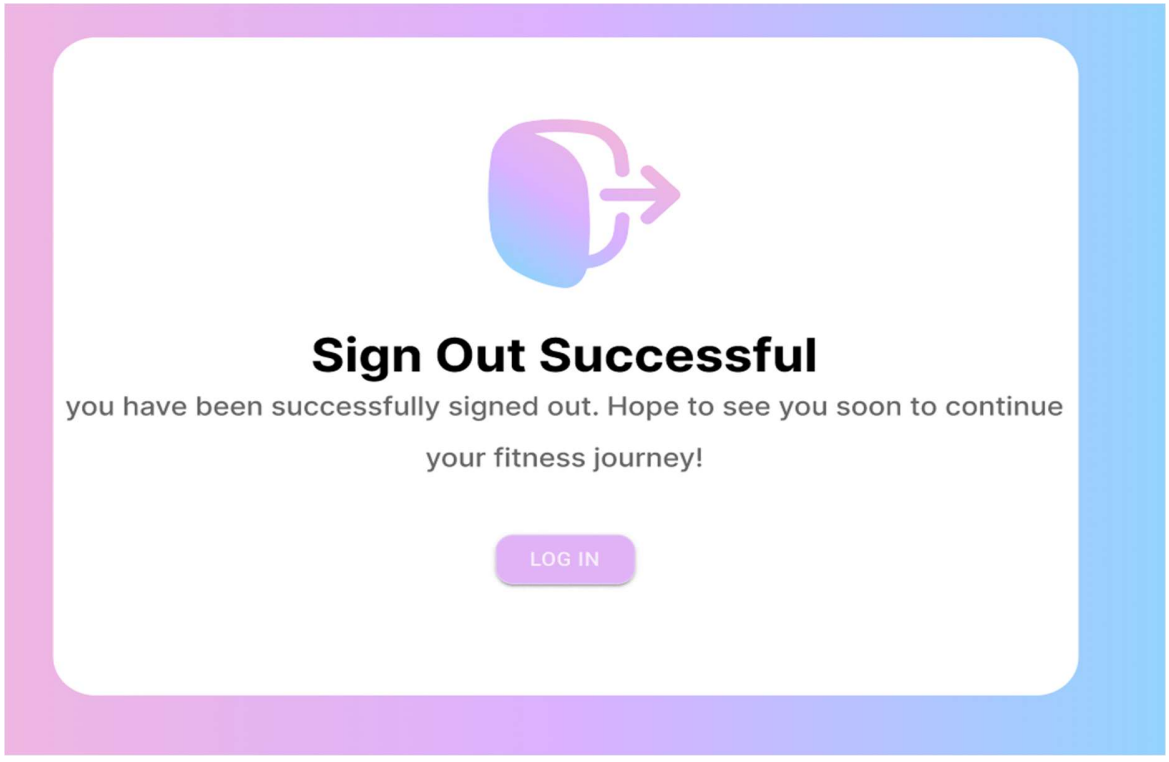


Figure 22 - Sign out page

Task Scenario for users

Step 1: Click the user tab on the top

Step 2: Navigate through the dashboard

Step 3: Click on settings menu to view its content

Step 4: Click on devices to see which devices are connected

Step 5: Click on companion to view your streak and progress

Step 6: Sign out of the user

Task Scenario for healthcare professionals

Step 1: Click on professional tab on the top

Step 2: Navigate through the dashboard

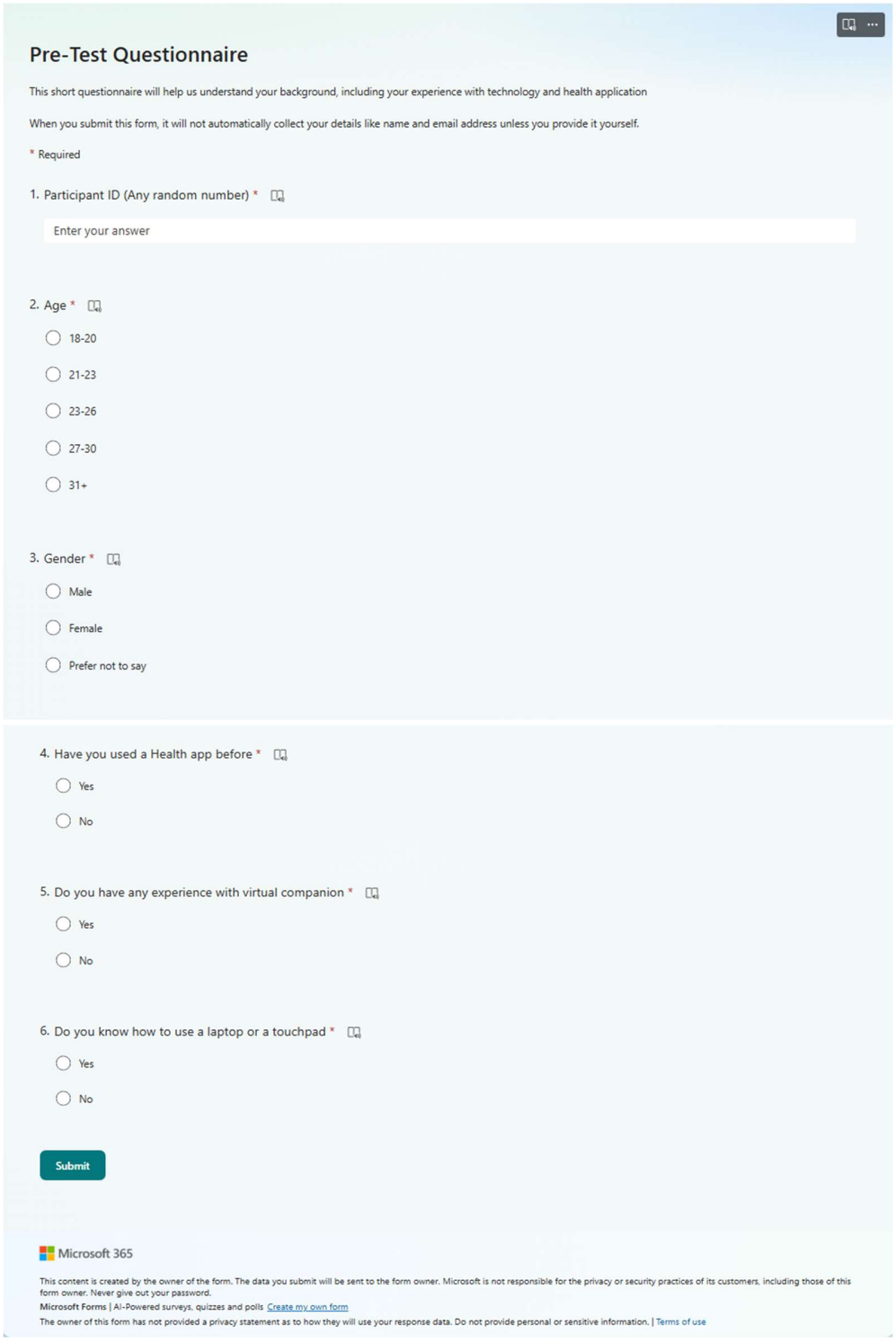
Step 3: click on view details for user 1 and check their calories, steps, and heart rate etc.

Step 4: Go back and click on settings to view the content

Step 5: Sign out of healthcare professional

Figure 23 - Both task scenarios used for evaluation

Pre-test Questionnaire



Pre-Test Questionnaire

This short questionnaire will help us understand your background, including your experience with technology and health application

When you submit this form, it will not automatically collect your details like name and email address unless you provide it yourself.

* Required

1. Participant ID (Any random number) *

Enter your answer

2. Age *

- 18-20
- 21-23
- 23-26
- 27-30
- 31+

3. Gender *

- Male
- Female
- Prefer not to say

4. Have you used a Health app before *

- Yes
- No

5. Do you have any experience with virtual companion *

- Yes
- No

6. Do you know how to use a laptop or a touchpad *

- Yes
- No

Submit

Microsoft 365

This content is created by the owner of the form. The data you submit will be sent to the form owner. Microsoft is not responsible for the privacy or security practices of its customers, including those of this form owner. Never give out your password.

Microsoft Forms | AI-Powered surveys, quizzes and polls Create my own form

The owner of this form has not provided a privacy statement as to how they will use your response data. Do not provide personal or sensitive information. | Terms of use

Figure 24 – Pre-test questionnaire

Post-test questionnaire

Post-test Questionnaire

After testing the prototype, please answer the questions below about your experience. Your feedback will help us improve the application in the future

When you submit this form, it will not automatically collect your details like name and email address unless you provide it yourself.

* Required

1. Participant ID (any random number) *

Enter your answer

2. The layout was clear and easy to understand *

- ◯ Strongly agree
- ◯ Agree
- ◯ Neutral
- ◯ Disagree
- ◯ Strongly disagree

3. Completing the tasks on the website felt easy *

- ◯ Strongly agree
- ◯ Agree
- ◯ Neutral
- ◯ Disagree
- ◯ Strongly disagree

4. It was easy to navigate between different screens *

- ◯ Strongly agree
- ◯ Agree
- ◯ Neutral
- ◯ Disagree
- ◯ Strongly disagree

5. The app colors were not pleasing *

- ◯ Strongly agree
- ◯ Agree
- ◯ Neutral
- ◯ Disagree
- ◯ Strongly disagree

6. It was easy to find and open various different menus *

- ◯ Strongly agree
- ◯ Agree
- ◯ Neutral
- ◯ Disagree
- ◯ Strongly disagree

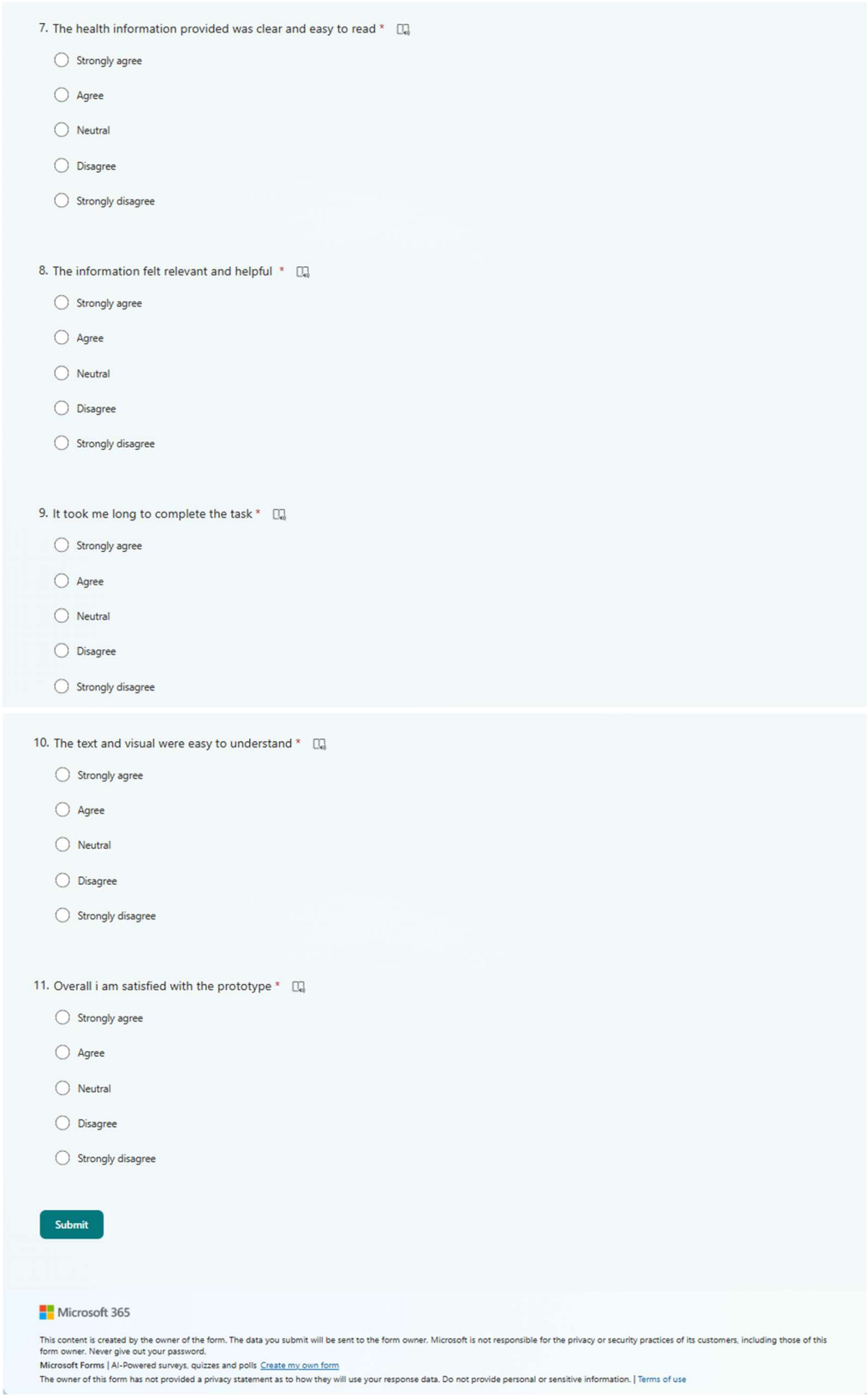


Figure 25 – Post-test questionnaires

Testing Protocol

Testing protocol

Facilitator initials:

Session number:

PulseLink

App testing protocol

Date:

Heriot-Watt University, Dubai

Aim: The purpose of this study is to assess the usability of the early app design for the PulseLink health application. Any input we receive before the project moves forward will be utilized to improve the interface and user experience.

The purpose of this test is to check how easily the participants can complete the required step-by-step tasks such as registering a new account, logging in, browsing through the dashboard etc. Since the app is in the prototype phase, not every function will be usable, but your feedback will help us improve the app in the future.

Introduction: This session is designed to evaluate the usability of an early prototype of PulseLink. The project aims to develop a Virtual Health Companion for Daily Living prototype that integrates real-time biomarker data from diverse sources, fostering user engagement and promoting informed health decisions.

During this session, I will ask you to complete some short tasks step by step. I will observe how easily you can complete them and take notes if any error occurs or you are stuck somewhere

After you finish the task, I will be presenting you with a short questionnaire regarding how your experience was.

Your participation is anonymous, and you can stop the session at any time.

Consent Form

Health App / Group 5 / Heriot watt university, Dubai / School of MACS

Consent to act as a participant in an experimental study

Aim:

The purpose of this study is to evaluate the usability of the early app design for the PulseLink health application. The feedback we receive will help improve the interface and user experience before development progresses.

Voluntary Participation and Withdrawal:

Your participation is voluntary. You may withdraw at any time without giving a reason. Participation or withdrawal will not affect your grades, course status, or relationship with university in any way.

Data Handling:

All data collected will be anonymous and used only for academic and evaluation purposes within the software engineering group project. No personal information such as name or student id will be included in the report.

Participant Voluntary Consent:

I confirm that I have read and understood the information above. I understand that participation is voluntary and that I can withdraw at any time. I confirm that I am at least 18 years old and have no known impediments that might affect my ability to provide consent or participate in this study.

Date _____ Participant signature _____ Individual signature _____