**Deliverable 3: Project Phase 1**

**CSCE 5430 (Fall 2024)**

**A. Requirements**

**1. Overview**

Phase 1 of the Health Data Analytics Platform is designed to establish the foundational functionalities necessary for effective user interaction, including user registration, authentication, and health data collection. In this phase, we aim to create a secure environment where users can register their accounts, log in, and enter various health metrics essential for tracking their well-being. This structured approach ensures that all functionalities work seamlessly together, providing a smooth user experience from the outset.

**Changes (Phase-1):**

In addition to the core functionalities, we have incorporated the capability to export health input data in CSV and JSON formats. This feature was specifically implemented to facilitate easy data export for users, allowing them to manage and share their health information conveniently. Moreover, the inclusion of data export functionality serves a dual purpose; it not only assists users in tracking their health metrics more efficiently but also provides a structured dataset that can be utilized in machine learning (ML) models in later phases of the project. This consideration for future enhancements ensures that our platform remains flexible and scalable, capable of integrating advanced analytics as the project evolves.

To implement the export functionality, we utilized the Pandas library, which provides a straightforward mechanism for converting user health data into both CSV and JSON formats. By allowing users to download their health metrics, we aim to enhance user engagement and encourage more active participation in managing their health. This capability not only fosters transparency but also empowers users with the control they need over their health information, making it easier for them to consult with healthcare providers or utilize the data for personal health tracking.  
  
**Functional Requirements**

* **User Registration and Authentication**
  + Users can create an account by providing a username, password, and email.
  + Users can log in using their credentials.
  + Google authentication is provided as an alternative login method.
* **Database Setup**
  + A SQL database is established to store user credentials and health data.
  + The database schema accommodates various health metrics.
* **Health Data Collection**
  + Users can input various health metrics after successful authentication:
    - Weight
    - Height
    - Blood Pressure (Systolic/Diastolic)
    - Heart Rate
    - Body Temperature
    - Body Mass Index (BMI)
    - Blood Glucose Level
    - Cholesterol Levels (Total, HDL, LDL, Triglycerides)
    - Oxygen Saturation
    - Activity Level
    - Dietary Intake (Calories, Macronutrients)
    - Sleep Patterns (Average Hours, Quality)
    - Medications (Name, Dosage, Frequency)
    - Symptoms or Concerns
  + Data input fields are optional.
* **Data Export**
  + Users can export their health data in CSV or JSON formats.

**Non-functional Requirements**

**1. Usability and User Experience**

The Health Data Analytics Platform must be designed with a user-friendly interface that allows users of all technical skill levels to navigate the application with ease. The application will adopt a clean and intuitive design using Streamlit, ensuring that users can quickly access key functionalities such as registration, authentication, and health data entry.

* **User-Friendly Interface**: The application will employ clear labeling, logical flow, and organized layouts, reducing cognitive load and making it easier for users to understand and use the platform. Components such as buttons, text inputs, and dropdowns will be easily distinguishable, with consistent styling applied across the application.

**2. Performance and Scalability**

While not directly related to security and usability, performance and scalability are crucial non-functional requirements. The application must be able to handle an increasing number of users and their data efficiently.

* **Database Optimization**: The SQLite database will be optimized with appropriate indexing strategies to ensure quick read and write operations. Additionally, database queries will be written efficiently to minimize response times.
* **Load Handling**: The application should be designed to handle multiple concurrent users without degrading performance. This might involve setting up a backend server with load balancing to manage user requests effectively.

**B. UML Design for Phase 1**

**1. Class Diagram**

+-------------------+

| User |

+-------------------+

| - id: int |

| - username: str |

| - password: str |

| - email: str |

+-------------------+

| + sign\_up() |

| + sign\_in() |

+-------------------+

+-------------------+

| HealthData |

+-------------------+

| - user\_id: int |

| - weight: float |

| - height: float |

| - blood\_pressure: str |

| - heart\_rate: float |

| - body\_temp: float |

| - bmi: float |

| - glucose\_level: float |

| - cholesterol: str |

| - oxygen\_saturation: float |

| - activity\_level: str |

| - dietary\_intake: str |

| - sleep\_patterns: str |

| - medications: str |

| - symptoms: str |

+-------------------+

| + collect\_health\_data() |

| + export\_data() |

A screenshot of a computer

Description automatically generated

**2. Sequence Diagram**

User -> Streamlit App: Sign Up/Sign In

Streamlit App -> Database: Verify Credentials

Database -> Streamlit App: Return User Info

Streamlit App -> User: Welcome Message / Error Message

User -> Streamlit App: Input Health Data

Streamlit App -> Database: Save Health Data

Database -> Streamlit App: Confirm Data Saved

Streamlit App -> User: Success Message

A screenshot of a computer

Description automatically generated

**3. Use Case Diagram**

* **Normal Case**: User registers, logs in, collects health data, and exports data.
* **Error Case**: User attempts to log in with incorrect credentials.

**C. Test Cases**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Input** | **Expected Output** |
| TC-01 | User Registration Success | username, password, email | "Account created successfully!" |
| TC-02 | User Registration Failure | empty username | "Username cannot be empty." |
| TC-03 | User Sign In Success | valid username, password | "Welcome back, [username]!" |
| TC-04 | User Sign In Failure | invalid username, password | "Invalid username or password." |
| TC-05 | Health Data Collection | weight, height | "Health data saved successfully!" |

**D. User Manual**

**Installation Instructions**

1. Clone the repository: git clone <repo-url >
2. Navigate to the project directory
3. Install required libraries:

pip install streamlit pandas authlib

**Usage Instructions**

1. Run the application:

streamlit run app.py

1. Select either "Sign Up" or "Sign In" from the sidebar.
2. Fill in the required fields and click the corresponding button to proceed.
3. After signing in, enter health metrics in the provided input fields.
4. Click "Save Data" to store health data, or use the download buttons to export the data.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**E. Compile/Run Instructions**

1. Ensure all dependencies are installed as per the installation instructions.
2. Run the application with the command:

streamlit run app.py

**F. Peer Review Feedback**

During the peer review session, feedback was received regarding the user interface design, suggesting it be made more intuitive. Based on this feedback, adjustments were made to the layout, and additional tooltips were added for user guidance.

**G. Reflection**

In Phase 1, we successfully implemented the core functionalities for user registration and health data collection. The process of integrating authentication methods was smooth, though additional testing revealed areas for improvement in user feedback upon successful data entry. Future phases will focus on enhancing user experience and implementing advanced features.

**H. Member Contribution Table**

**Member Contribution Table**

|  |  |  |
| --- | --- | --- |
| **Member Name** | **Contribution Description** | **Overall Contribution (%)** |
| Jaideep Tripurani | Authored the introduction and requirements section | 15% |
| Ajay Eedara | Developed the backend processing logic | 20% |
| Tagore Hari Prasad Chintamaneni | Designed the database schema and queries | 15% |
| Satish Velaga | Implemented user authentication and security features | 20% |
| Siddhartha Alapati | Contributed to health data analysis algorithms | 10% |
| Sai Shruthik Errammagari | Contrite the ui Phase | 10% |
| Devendra Kumar Gaddipati | Engaged in iterative testing of algorithms for health data analysis | 10% |
| Sai Venkata Manish Lingamallu | Compiled the report, edited final files | 10% |
| Ajay Kumar Aitha | Assisted with testing and debugging | 10% |