*Ellen Dalton-Dawe*

*Keyin College | April 4, 2024*

*The Smart Health Monitoring System*

A Java-Based Project

Table of Contents

1.0 User Documentation...............................................................................................2

1.1 About the Application

1.2 Classes and Functionality

1.3 How to Start/Access the Application

* 1. Class Diagrams

2.0 Development Documentation..................................................................................1

2.1 Javadoc’s

2.2 Source Code Structure

2.3 The Build Process

2.4 Compiler Time Dependencies

2.5 Development Standards

2.6 Setting up a Database for Development

2.7 Getting the Source Code from the Repository

3.0 Deployment Documentation....................................................................................1

1.0 User Documentation

* 1. About the Application

The Smart Health Monitoring System is a Java-based project designed to help users maintain a healthy lifestyle by tracking their daily activities and vital signs. The project uses a database to store user data, allowing for secure login and protected data. Upon logging in, users can then view and update their personal information and can track various health metrics such as weight, height, number of steps taken per day, and heart rate. The system utilizes the collected health data to generate personalized health recommendations for each user. Also, users can set reminders for taking medications at specified times, and can also update or delete these reminders as needed. Healthcare professionals have access to a special portal where they can look up doctors, search for a specific doctor’s list of patients, and can list health data for a particular patient, allowing doctors to monitor their patients health trends over time. Overall, the Smart Health Monitoring System provides users with a convenient platform to monitor their health data and receive personalized recommendations for leading a healthier lifestyle. It also offers valuable tools for healthcare professionals to track and manage their patients health effectively.

* 1. Classes and Functionality

**User** – Represents a user of the Smart Health Monitoring System. This class stores attributes such as user ID, first name, last name, email, password and “isDoctor”. The “isDoctor” attribute distinguishes between regular users and doctors, where doctors have additional attributes stored in the database along with access to a special doctor portal.

**Doctor** – Represents a special type of user that has access to the doctor portal. This class stores attributes such as doctor ID, first name, last name, email, password, “isDoctor”, as well as medical licence number and specialization.

**Health Data** – Represents the health data of a specific user within the Smart Health Management System. It stores various attributes such as the health data ID, the user ID, weight, height, steps, heart rate and date the data was measured.

**Medicine Reminder** – Represents user medication reminders within the Smart Health Management System. It contains attributes such as the reminder ID, user ID, medicine name, dosage, schedule, start date and end date.

**Recommendation System** – This class is responsible for generating personalized health recommendations based on the users health data input. It analyzes the heart rate and steps taken per day in order to provide tailored advice for improving the users well-being.

**User DAO** – This class is responsible for handling user-related database operations within the Smart Health Monitoring System. It includes methods for user creation, user retrieval, updating a user, deleting a user, verifying the password, and getting the user ID by their email.

**Health Data DAO** – This class is responsible for handling health data related database operations within the Smart Health Monitoring System. It provides operations for creating new health data, retrieving stored health data, updating stored health data, deleting stored health data, and getting the health data ID.

**Medicine Reminder Manager** – This class provides CRUD (Create, Read, Update, Delete) operations for managing medicine reminders within the Smart Health Monitoring System. These operations allow for users to effectively manage their medication schedules by adding, retrieving, updating, and deleting reminders as needed. It also fetches the medicine reminder ID’s from the database.

**Doctor Portal DAO** – This class is responsible for managing interactions between the doctor portal and the database. It facilitates operations such as creating a doctor, getting a doctor by doctor ID, getting a doctor by email, verifying the password, updating a doctor, getting patients by doctor ID, getting health data by patient ID, and getting a doctor ID by email.

**Health Monitoring App** – This is the main class of the Smart Health Monitoring System in which all the other methods and classes are called. Within this class a main menu is displayed, from which the user can choose what operation he/she would like to undertake. At the top level, a user can register or login to the portal. After login, he/she can choose from a variety of options, which are dependent on whether he/she is a doctor accessing the doctor portal, or a user accessing the user portal. From the doctor portal a doctor can choose from the following options: 1. View and Update Information 2. Look-Up Doctor by Doctor ID 3. Look-up Patients by Doctor ID 4. List health data for a particular patient. From the patient portal a user can: 1. View and Update Information 2. Input Health Data 3. View and Update Health Data 4. Delete Health Data 5. View Personalized Health Recommendations 6. Set a Medicine Reminder 7. View and Update Medicine Reminders 8. View OVERDUE Medicine Reminders 9. Delete a Medicine Reminder.

* 1. How to Start/Access the App
* Ensure you have Visual Studio Code (VS Code) installed on your computer.
* Download the Java Development Kit (version 17.0.9 or later).
* Download the Java Extension Pack in your VS Code.
* Download source code Java files to your local disk. For further instruction on how to do this, see section 2.7 “Getting the Source Code from the Repository”.
* After downloading, open the directory containing the source files with VS Code. To do this, open VS Code and click on “Open Folder”. Navigate to the source code directory and click “Open”.
* Once opened with VS Code, double click on the HealthMonitoringApp class to open the main class.
* To run the program, click on the right-pointing triangle in the upper right corner of VS Code. This should open a terminal window where you will see the output and where you can interact with the program.
  1. Class Diagrams

2.0 Development Documentation

2.1 Javadoc’s

* Javadoc comments are provided for all classes and methods to document their functionality and usage.
* To compile the Javadoc documentation in a HTML format, you can use the VS Code integrated terminal.
* Open the terminal within VS Code by selecting “Terminal” > “New Terminal” from the menu.
* In the terminal, navigate to the directory containing the downloaded Java files and use the following Javadoc command: javadoc -d docs \*.java
* This will generate documentation for all Java files in the current directory, and its subdirectories, and will output the HTML files to a directory named “docs”.
* Ensure you have the Live Server extension installed in your VS Code. This will enable you to run the Javadoc documentation from a browser.
* Right click on the help-doc.html file and select “Open with Live Server”. This should open the Javadoc Help menu in a browser environment.
* From here, you can navigate to all Javadoc documentation within the browser environment (the help menu is listed as a starting point so that it can help guide you through the usage of the Javadoc documentation).

2.2 Source Code Structure

* All Java source files are organized in a single directory. Each class is defined in its own .java file.

2.3 The Build Process

* Compile the project using a Java compiler (ex: javac \*.java).

2.4 Compiler Time Dependencies

* There are no external dependencies required at compile time.

2.5 Development Standards

* Followed standard Java coding conventions and best practices for naming, formatting, and structuring code.

2.6 Setting up a Database for Development

* This application does utilize a database for development.
* Ensure that MySQL server and pgAdmin is installed on the development environment.
* Ensure that access credentials (username and password) for MySQL server are available.
* Launch pgAdmin and connect to the MySQL server using the appropriate credentials.
* Create a new database.
* Create all required tables according to the CREATE statements saved in the project directory.
* Update the database connection settings in the DatabaseConnection.java file.
* Optionally, populate the database with initial data for development/testing purposes.
* Restart the application to apply the changes.
* To verify that the application can connect to the MySQL database, perform basic CRUD (create, read, update, delete) operations to ensure the database interaction is functioning as expected.

2.7 Getting the Source Code from the Repository

* Visit <https://github.com/ejd500/Term3-Sprint2-Java-SmartHealthMonitoringSystem> to download source code from the repository.
* To download, click on the green button called “Code”. From here you will see an option to “Download ZIP” – click it!
* Double click on the downloaded file to unzip it and proceed to open the folder with your VS Code.

3.0 Deployment Documentation

* Ensure Java Runtime Environment (JRE) is installed on the target system.
* Copy the compiled .class files to the deployment directory.
* Run the HealthMonitoringApp class to start the application.
* See section 1.3 “How to Start/Access the App” of this document for further instruction.