

**Health Guard Pro**

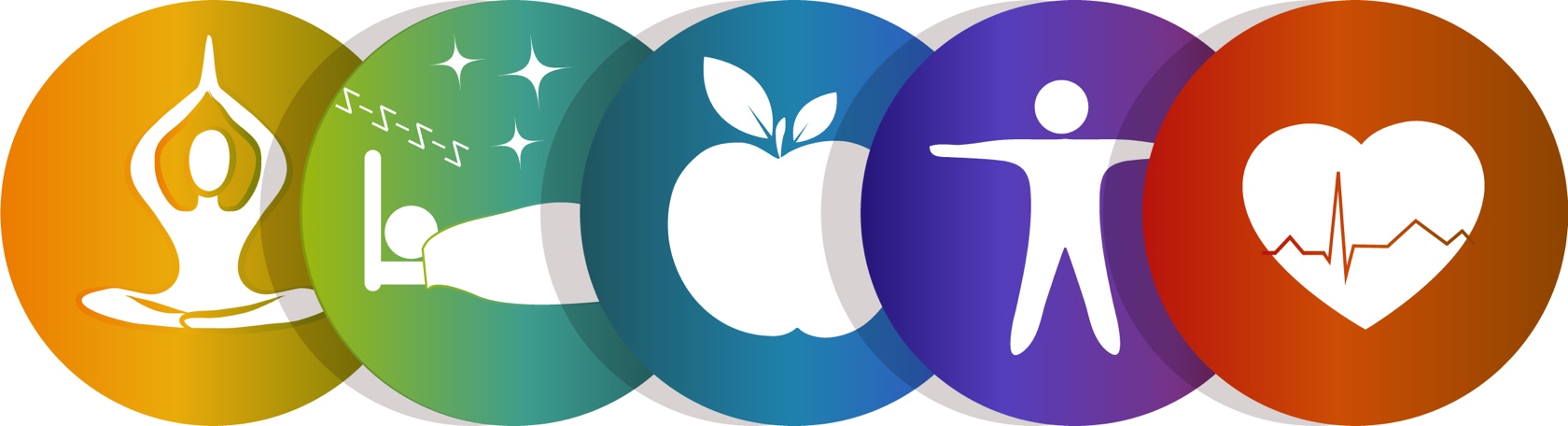
Smart Health Monitoring System

User Manual

Version: 1.0.0

**Prepared by:** Malerie Earle

**Date:** April 14, 2024

**Welcome to Health Guard Pro**

**Smart Health Monitoring System!**

**Introduction**

Welcome to Health Guard Pro, your comprehensive smart health monitoring system! This guide offers insights into the features and best practices for utilizing our software, tailored to seamlessly organize user lifestyles. With monitoring capabilities spanning vital signs, medical records, and daily activities, our platform ensures thorough health management. Additionally, doctors are granted accounts to oversee patient information and medications, including personalized medication reminders for users, detailing dosage, and frequency.

**User Documentation**

**Benefits for Users:**

The Smart Health Monitoring System offers several benefits:

* Efficient management of user information including medical records, vital signs, medication list, daily activities, and immunization records.
* Doctors have their own accounts to manage and view patient data, post medical records/complete charting, and access medication lists.

# **Purpose of the Manual:**

# This manual serves as Smart Health Monitoring *System*. It provides instructions on installing, configuring, and using the software. Additionally, it includes documentation on the code structure, development process, and deployment procedures.

1. **Install Java Development Kit (JDK):** Ensure you have the Java Development Kit (JDK) installed on your system. You can download and install the JDK from the official Oracle website or adopt OpenJDK, which is an open-source implementation of the JDK.
2. **Set up Git:** If you haven't already, download and install Git from the official website. Git is a version control system that allows you to manage and track changes to your code.
3. **Access the GitHub Repository:** Navigate to the GitHub repository containing the Java application you want to install.
4. **Clone the Repository:** Once you're on the GitHub repository page, locate the green "Code" button and click on it. This will reveal a URL. Copy the URL to your clipboard. Open your terminal or command prompt and navigate to the directory where you want to clone the repository. Use the **cd** command to navigate directories. Once in the desired directory, use the **git clone** command followed by the URL you copied earlier. For example:

git clone https://github.com/malerie-earle/FinalJavaSprint.git

Replace “**https://github.com/username/repository.git”** with the actual URL of the GitHub repository you want to clone.

1. **Compile the Java Application:** After cloning the repository, navigate into the cloned directory using the **cd** command in your terminal or command prompt. Once inside the directory containing the Java application's source code, you may need to compile the code if it's not already compiled. Use the **javac** command to compile the Java files. For example:

javac YourMainClass.java

Replace **YourMainClass.java** with the main class file of the Java application.

1. **Run the Java Application:** After compiling the code (if necessary), you can run the Java application using the **java** command followed by the name of the main class. For example:

java YourMainClass

Replace **YourMainClass** with the name of the main class of the Java application.

1. **Explore and Use the Software:** Once the application is running, you can explore its functionalities based on its documentation or user interface. Interact with the software as needed to achieve your desired tasks.

These instructions should guide you through the process of installing a Java application from a GitHub repository and using it on your system.

Bottom of Form

**Purpose & Key Features**

# The Smart Health Monitoring System “Health Guard Pro” is a comprehensive software solution that allows users to manage many aspects of their medical information, including personal details, daily activities, vital sign tracking, step count log, medication lists and more. The key functionalities include:

**User Management:**

* The system allows the adding of new users, editing of user information, and deleting of users from the database. Each user has an ID number, first and last name, email, password and whether they are a doctor.
* Users can be assigned to a doctor who can access the information they share with the system.

**Doctor Management:**

* Doctors can be users as well. Their accounts have their medical license numbers, and specialization. With their account they can access their patient lists, their patient’s information, add data to the patients record, and add medications to their medication lists.

**Medication Reminders:**

* Incorporated into this app is a reminder system that will remind you to take any prescribed medication when you are supposed to take it. It even has a start and end date for medications such as antibiotics that have a specific duration.

**Health Data Storage:**

* This application allows you to store various personal and health information. This includes your height, weight, age, resting heart rate, blood pressure, blood sugar, along with keeping track of your current medications, any allergies you may have, and any pertinent medical history your doctor should be aware of.

**Daily Activity Tracker:**

* Allows you to add your daily activities such as your steps taken, minutes of cardio exercise in a week, and your active heart rate. You can even keep track on your sleep schedule and how many hours you get each night.

**Health Recommendations:**

* Based on your actual health information, this app is able to give various recommendations based on the input data of resting heart rate, active heart rate, blood pressure, blood sugar, height, weight, hours slept, calculates your BMI and gives recommendations based on it. Also takes into consideration your weekly minutes of cardio, steps taken per day, and allows access by your family doctor.

**Development Documentation**

**Class Diagrams**

|  |
| --- |
| **User Class** |
| - user\_id: Integer  - first\_name: String  - last\_name: String  - email: String  - password: String  - is\_doctor: Boolean |
| + User(int user\_id, String first\_name, String last\_name, String email, String password, Boolean is\_doctor)  + addUser()  + editUser()  + deleteUser() |

|  |
| --- |
| **Doctor Class Extends User Class** |
| - dr\_id: Integer  - medLicenseNum: Integer  - specialization: String  - patientList: <<ArrayList>> |
| + Doctor(Integer dr\_id, String first\_name, String last\_name, String email, String password, Boolean is\_doctor, Integer medLicenseNum, String specialization, patientList: ArrayList user) |

|  |
| --- |
| **Health Data Class** |
| - health\_id: Integer  - user\_id: Integer  - gender: String  - age: Int  - weight: Double  - height: Double  - rest\_heart\_rate: Integer  - active\_heart\_rate: Integer  - blood\_pressure: String  - blood\_sugar: Double  - steps: Integer  - cardio\_minutes: Double  - hourse\_slept: Integer  - date: String  - med\_list: String  - allergies: String  - med\_history: String  - dr\_id: Integer |
| + HealthData(int health\_id, int user\_id, String gender, int age, double weight, double height, int restHeartRate, int activeHeartRate, String bloodPressure, double bloodSugar, int steps, int cardioMinutes, int hoursSlept, String date, String medList, String allergies, String medHistory, int dr\_id) |

|  |
| --- |
| **Medicine Reminder Class** |
| - med\_id: Integer  - user\_id: Integer  - med\_name: String  - dosage: String  - schedule: String  - start\_date: String  - end\_date: String |
| + MedReminder(Integer med\_id, Integer user\_id, String med\_name, String dosage, String schedule, String start\_date, String end\_date) |

Recommendations, AuthenticationSystem, DAO, DatabaseConnection and HealthMonitoringApp files are also used.

**Javadocs**

Javadocs for the project were generated using the `javadoc` tool included with the JDK. Run the following command in the root directory of the project:

*javadoc -d docs -sourcepath src -subpackages JavaSprint*

This generated Javadocs for all classes in the `FinalJavaSprint` package and put them in a folder named `docs`.

**Source Code Directory Structure**

* All Java source files are contained within the **FinalJavaSprint** package.
* The documentation folder contains subfolders for different types of documentation, including **docs** for Javadocs, **legal** for legal documents, **project** for project-related documentation, and **script-dir** for any scripts related to the project.
* This directory structure helps maintain a clear separation between source code, documentation, and other project-related files, making it easier to manage and navigate the project.

**System Requirements**

* Java Runtime Environment (JRE) installed on your system.
* Command-line interface (CLI) or Integrated Development Environment (IDE) to run Java programs.
* The system requires a computer or server with sufficient resources to run the software smoothly.

# Java Development Kit (JDK) version 8 or above.

# *System Compatibility:* The software is compatible with Windows, macOS, and Linux operating systems.

**Build Process**

To compile the project, navigate to the `src` directory in your terminal and run the following command:

Ex:

javac JavaSprint/\*.java

This will compile all Java files in the `JavaSprint` package.

**Compiler Time Dependencies**

The project uses the `java.util` package for classes like `List` and `Scanner`. No external libraries or frameworks are used.

**Development Standards**

The project follows standard Java coding conventions. Class names are in PascalCase, method and variable names are in camelCase, and constants are in UPPER\_SNAKE\_CASE.

**Database Setup**

If I was to set up a database for this project, I would set the database up as per the class diagrams above.

**Deployment Documentation**

**Getting Started**

To start using Bookworm Buddy, follow these steps:

1. **Download**: Obtain the Bookworm Buddy software package from the designated source.
2. **Installation**: No installation is required as Bookworm Buddy is a standalone Java application.
3. **Compilation**: Compile the Java files using a Java compiler. You can do this by navigating to the directory containing the Java files and running the following command:

*javac Main.java*

1. **Execution**: Once compiled successfully, execute the main class to launch the application:

*java Main*

1. **Usage**: Follow the on-screen instructions to navigate through the application and utilize its features.

**Support**

For any assistance or queries regarding Smart Health Monitoring App, please contact our support team at info@newfienook.com. We provide prompt assistance and resolving any issues you may encounter while using our software.

**Conclusion**

In conclusion, the Smart Health Monitoring System (Health Guard Pro) is a comprehensive software solution designed to empower users in managing their health effectively. Through a user-friendly interface and robust functionalities, this system aims to streamline health monitoring, encourage healthy habits, and facilitate better communication between users and healthcare professionals.