

BSc (Hons) in Computing Level C/I/H



INDIVIDUAL ASSIGNMENT

Module Code & Title: OOAEE-2 (COSE50582)

Prepared By: Avishka Dimal Senanayake (CB007077)

Date of Submission: 27th of January 2021

Instructor:

Umanga Pilapitiya

Word Count: 6538

MARKING CRITERIA	%	MARKS OBTAINED
TOTAL (%)		

Acknowledgement

Towards the completion of this assignment, it was a great factor of support to receive financial aid for research along with motivational support to keep pushing forward despite the difficulties and challenges that this assignment had towards completion.

Additionally, I would like to show my utmost gratitude towards Miss Umanga Pilapitiya for providing support and necessary reading material for knowledge gathering and research for the assignment. Including valuable feedback for the assignment topic selected and prompt responses to questions via emails during COVID-19 online learning times.

Extra thanks go to APIIT for having necessary online facilities to easily support and host various learning materials from lecture sessions that were needed to revise on learnt Java based topics while being able to do it at home.

Executive Summary

For this assignment, we have been given the task of developing an application with extensive use of the Gang of Four (GoF) design pattern which are a set of practices that are used with object-oriented programming to make developing more organized and easier for the task at hand.

I have created a standalone application that focuses on common domestic health issues that many people face day to day that can either be short or long term. It involves with medical remedies and providing information.

This application was created using NetBeans IDE with MySQL for its backend database while using XAMPP control panel to host the web server. The GUI design was done by using Java Swing and object-oriented programming concepts with design patterns.

Table of Contents

Contents

Acknowledgement	- 2 -
Executive Summary	- 3 -
Table of Contents	- 4 -
Table of Figures.....	- 7 -
Introduction.....	- 9 -
Analysis or Specification	- 11 -
• Application Functional Specifications	- 11 -
Sign up Function.	- 11 -
Login Function.....	- 11 -
Account Authorization Levels	- 11 -
Tabbed View Control.....	- 11 -
Log out Function.....	- 12 -
Non-Functional Requirements	- 13 -
Usability	- 13 -
Serviceability	- 13 -
Data Integrity	- 13 -
Maintainability	- 13 -
Performance	- 14 -
Requirements.....	- 15 -
Hardware Requirements.....	- 15 -
Software Requirements	- 16 -
Design	- 17 -
Class Diagrams	- 17 -
Factory Design Pattern (Medicine Recommendation).....	- 17 -
Singleton Design Pattern (Database Connection).....	- 19 -
Template Design Pattern (Ailment Info)	- 20 -
Façade Design Pattern (Sign up).....	- 22 -
Proxy Design Pattern (Application Users List)	- 24 -
Decorator Design Pattern (Medicine Purchases)	- 25 -
Implementation	- 27 -
Sign Up (Facade)	- 27 -
• signupinterface	- 27 -
• signupClearall	- 28 -

• validateSignup.....	- 29 -
• Submittodb.....	- 30 -
• signupFacade.....	- 31 -
• signup (facade).....	- 32 -
• Façade Output	- 33 -
Connection Manager for Database (Singleton).....	- 34 -
• ConnectionManager.....	- 34 -
• validate	- 35 -
Ailment info (Template)	- 36 -
• Template Interface	- 36 -
• Obesity	- 37 -
• Migraines	- 38 -
• Hypertension	- 40 -
• Gastritis	- 41 -
• Bloodsugar	- 42 -
• Template Output	- 45 -
Medicine Recommendation (Factory Pattern)	- 47 -
Patient	- 47 -
• Patient_head.....	- 48 -
• Patient_lower_body	- 55 -
• Patient upper body	- 63 -
• Pain_Area.....	- 71 -
• mainpage (factory)	- 73 -
• Factory Pattern output.....	- 75 -
Application Users List (Proxy)	- 77 -
• ProxyExecutor Interface	- 77 -
• Execute.....	- 78 -
• ExecuteProxy	- 79 -
• Login (proxy)	- 80 -
• Proxy Output.....	- 82 -
Medicine purchases (Decorator)	- 84 -
• Meds.....	- 84 -
• Abenol.....	- 85 -
• Aspirin.....	- 86 -
• Ibuprofen.....	- 87 -
• MedDecorator	- 88 -
• _250mg	- 89 -
• _500mg	- 90 -
• _5mg	- 91 -
• mainpage (decorator)	- 92 -
• Decorator Output	- 96 -
Use Case Diagrams	- 98 -
Application users list use case	- 98 -

Login use case	- 99 -
Medicine purchases use case	- 100 -
Ailment info use case	- 101 -
Test cases.....	- 102 -
 Conclusion	 - 106 -
 Bibliography	 - 107 -

Table of Figures

Figure 1. Factory Design Pattern (Medicine Recommendation)	17 -
Figure 2. Singleton Design Pattern (Database Connection)	19 -
Figure 3. Template Design Pattern (Ailment Info)	20 -
Figure 4. Façade Design Pattern (Sign up)	22 -
Figure 5. Proxy Design Pattern (Application Users List)	24 -
Figure 6. Decorator Design Pattern (Medicine Purchases)	25 -
Figure 7.1 signupinterface (facade)	27 -
Figure 8. signupClearall (facade)	28 -
Figure 9. validateSignup (facade)	29 -
Figure 10. Submittodb (facade)	30 -
Figure 11. signupFacade (facade)	31 -
Figure 12. signup (facade)	32 -
Figure 13. Facade Output 1	33 -
Figure 14. Facade Output 2	33 -
Figure 15. ConnectionManager (Singleton)	34 -
Figure 16. Template Interface (Template)	36 -
Figure 17. Obesity (Template)	37 -
Figure 18. Migraines (Template)	38 -
Figure 19. Mentalhealth (Template)	39 -
Figure 20. Hypertension (Template)	40 -
Figure 21. Gastritis (Template)	41 -
Figure 22. Bloodsugar (Template)	42 -
Figure 23. Mainpage (Template)	43 -
Figure 24. Template Output 1	45 -
Figure 25. Template output 2	46 -
Figure 26. Patient (Factory)	47 -
Figure 27. Patient_head (factory)	53 -
Figure 28. Patient_lower_body (factory)	62 -
Figure 29. patient_upper_body (factory)	70 -
Figure 30. Pain_Area (Factory)	72 -
Figure 31. mainpage (factory)	74 -
Figure 32. Factory pattern output 1	75 -
Figure 33. Factory pattern output 2	76 -
Figure 34. ProxyExecutor Interface (Proxy)	77 -
Figure 35. Execute (proxy)	78 -
Figure 36. ExecuteProxy (factory)	79 -
Figure 37. Login (proxy)	80 -
Figure 38. isAdminProxy and isNOTAdminProxy	81 -
Figure 39. Table method for Proxy	81 -
Figure 40. If user is an admin proxy	82 -
Figure 41. If user is NOT an admin Proxy	83 -
Figure 42. Meds (Decorator)	84 -
Figure 43. Abenol (Decorator)	85 -
Figure 44. Aspirin (Decorator)	86 -

Figure 45. Ibuprofen (Decorator).....	- 87 -
Figure 46. MedDecorator (Decorator)	- 88 -
Figure 47. _250mg (decorator)	- 89 -
Figure 48. _500mg (decorator)	- 90 -
Figure 49. _5mg (Decorator)	- 91 -
Figure 50. mainpage (Decorator).....	- 94 -
Figure 51. Decorator Output 1	- 96 -
Figure 52. Decorator Output 2	- 97 -
Figure 53. Application users list use case	- 98 -
Figure 54. Login use case	- 99 -
Figure 55. Medicine purchases use case	- 100 -
Figure 56. Ailment info use case	- 101 -

Introduction

The goal of this assignment was to create an application of either web-based or standalone (Java swing GUI) application by utilizing the Gang of Four design patterns with the utilization of Java object orientation for certain defined tasks during development. Database utilization was also needed for the backend of the application for certain features.

Gang of Four design patterns were based on the popular book from John Vlissides, Erich Gamma, Ralph Johnson, Richard Helm (GoF) and its sole focus was a set of reusable dry of solutions or "patterns" for certain tasks or problems based that developers may face during a development life cycle on a context in software engineering design. The GoF categorized these patterns into 3 distinct groups with its purposes as, Behavioral, Structural and Contextual design patterns. Overall, it is a common pattern or practice for programmers to solve common problems that are recurring in an application. (*Chovatiya, 2020*).

Design patterns are something that recommended to be used because for certain scenarios in programming, the nature of these patterns can make handling tasks in the program much more versatile. These tasks can span from account permission control with features to minimalizing repetition with what is displayed, and many more.

For this assignment, I have created a 'Health based Action Recommendation' application with the name of 'Extra Life Health',

The application consists of the following features:

- **Login** – Already registered users will be able to login to the application to use its features.

- **Sign Up (Façade Pattern)** – Users will be able to register their account in the application to then login and use the application's features in the future.
- **Medicine Recommendation (Factory Pattern)** – By answering a few questions such as age, gender pain location, etc, the application will provide the user with a recommendation of appropriate medicine or to visit the doctor immediately.
- **Ailment Info (Template Pattern)** – The user can select a topic of common health issues to get a more in-depth explanation of its causes and how to remedy them.
- **Medicine Purchases (Decorator Pattern)** – From 3 choices of medicine, the user can select the one they require with its milligram amount and enter the quantity they desire. Where the application will provide information and its basic cost scheme by adding tax per milligram and adding the medicine product's main cost.
- **Application User List (Proxy Pattern)** – During login, if the user enters with an Admin account, they will be able to view a list of the registered users in the application with their Name, Username and Email. However, if the user did not login with an Admin account, they will not be able to view this information.
- **Database Connection (Singleton)** – Where a connection to the database is necessary, a singleton design pattern will handle the connection to the relevant database by providing the relevant login information and password to store or retrieve data.
- **About** – The application will display brief information about the application's creation and its sources for information.

Analysis or Specification

- **Application Functional Specifications**

Sign up Function.

- The application should have a sign-up function where users will be able to enter their relevant information such as first name, email, username, password to register their account. After registering, they should be able to use the information entered, login to the account in order to use the features that the health application provides.

Login Function.

- If users have already registered an account, they will be required to login to the account in order to use the health application's features. Depending on the user account's permissions, a certain feature will be hidden to be only viewable by administrator accounts. Regular users cannot view them.

Account Authorization Levels

- Depending on the user account, certain features that are in the application should be locked off until a user with the necessary permissions logs in. Once they do, the features will be available for viewing. In the application's case, this is a list of all registered users of the application which can be only viewed by admins.

Tabbed View Control

- The health application's features should be listed down in a tab control in the GUI after logging in. After logging in is done, the users will be able to select the relevant functions they require from the application by clicking on the relevant tabs in the GUI.

Log out Function.

- After logging into the relevant account, the user should be able to log out of the account in use in order to be able to use separate account to log in back to the application.

Non-Functional Requirements

Usability

- The application should have a clear GUI design that is easy for the client or user to comprehend and use its features that are on offer. If the UI is not user friendly it will come off as incomprehensible to the one viewing it and will result in an unintuitive design.

Serviceability

- The application's backend should be easy to be serviced and changed whenever the need arises. This can be to add additional features to the list or to improve already existing ones. This is done in the health application by placing a single feature in a distinct tab in the main screen's tab control. This way, it will also be easy to identify what to change in the backend.

Data Integrity

- Certain critical features or information should be hidden from users that does not need to see the confidential data. This could span from login information to payment details. In the health application this is done with the use of the proxy design pattern to hide certain data from users.

Maintainability

- The application should be easy to be maintained in both backend and front end. The code and the fundamental design of the UI should be robust to not create any future issues to the ones that are optimizing the application for future use.

Performance

- When the application is in use, the user should not experience any experience halting effects in terms of performance. When running, the application must provide the user with a smooth and robust experience until the user decides its necessary to exit the application.

Requirements

Hardware Requirements

The creation of this application and the completion of this assignment was done with a desktop with the following hardware specifications:

- **CPU:** AMD Ryzen 7 2700x 8 Core Processor
- **RAM:** 16GB
- **Graphics Card:** Nvidia GeForce RTX 2080
- **Storage:** 1TB HDD
- **Peripherals:** Mouse and Keyboard

A personal computer with similar or containing hardware that is generally new, will be able to run the application with little to no issues, including performance and user experience.

Software Requirements

The application was developed and compiled to completion by utilizing the following software's and operating systems:

- Oracle NetBeans IDE 8.2
- Mozilla Firefox (For research purposes)
- XAMPP Control Panel (For web server)
- MySQL Workbench (Handling database)
- Windows 10 64-bit Operating System
- Draw.io Webpage (For UML design)
- NetBeans IDE “easyUML” Plugin (For UML design)

The Extra Life Health application should run on the following software's with any or most modern operating systems with minimal amounts of issues to deal with in terms of user experience.

Design

Class Diagrams

Factory Design Pattern (Medicine Recommendation)

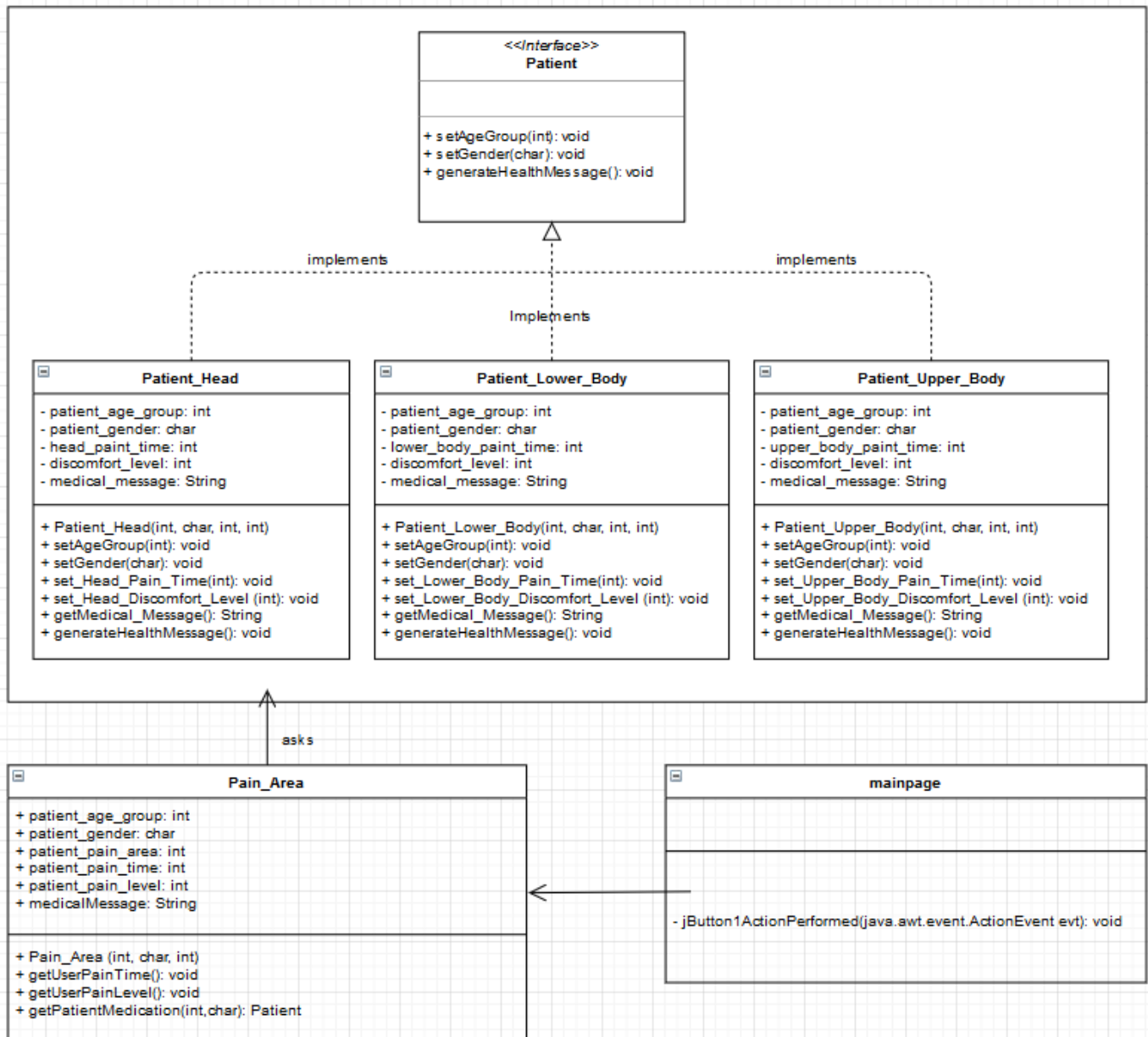


Figure 1. Factory Design Pattern (Medicine Recommendation)

Factory pattern is one of the most commonly used design patterns with object orientation. It falls under the category of the creational type patterns.

The purpose of it is to handle object creations per class.

In the context of this application, I have handled the three sub classes to have unique objects made to retrieve its unique information when needed. In this case, it makes things more organized through the use of an interface.

The user will have an option to select upon a few questions asked in the GUI, and depending on the choices made, will be recommended a type of medicine to remedy the illness at hand, or provide other recommendations to visit a doctor.

The choices are based on the parts of a body spanning from head, lower body and upper body.

Due to this, using a factory pattern is the best choice.

Singleton Design Pattern (Database Connection)

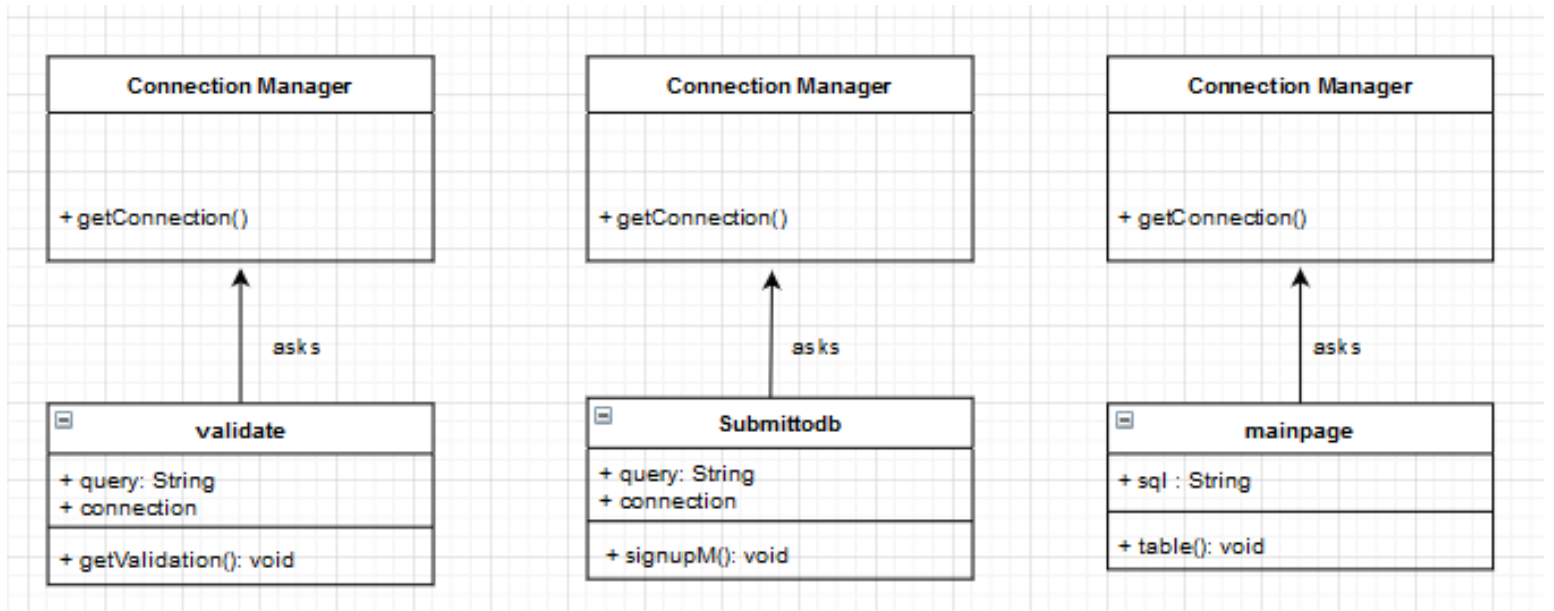


Figure 2. Singleton Design Pattern (Database Connection)

Singleton design pattern is one of the easiest and the most used GoF design patterns and it falls under the creational design pattern category. It deals with object creation.

As its name (singleton) suggests, the design pattern only deals with a single object to be used in another class.

Because the application needs to pull the database connection from only one place for entering and viewing data in the database, the use of a singleton design pattern makes things more adequate as it only needs one object to be used in another class to have database functionality.

Thus, a singleton pattern is the better choice for handling database connections.

Template Design Pattern (Ailment Info)

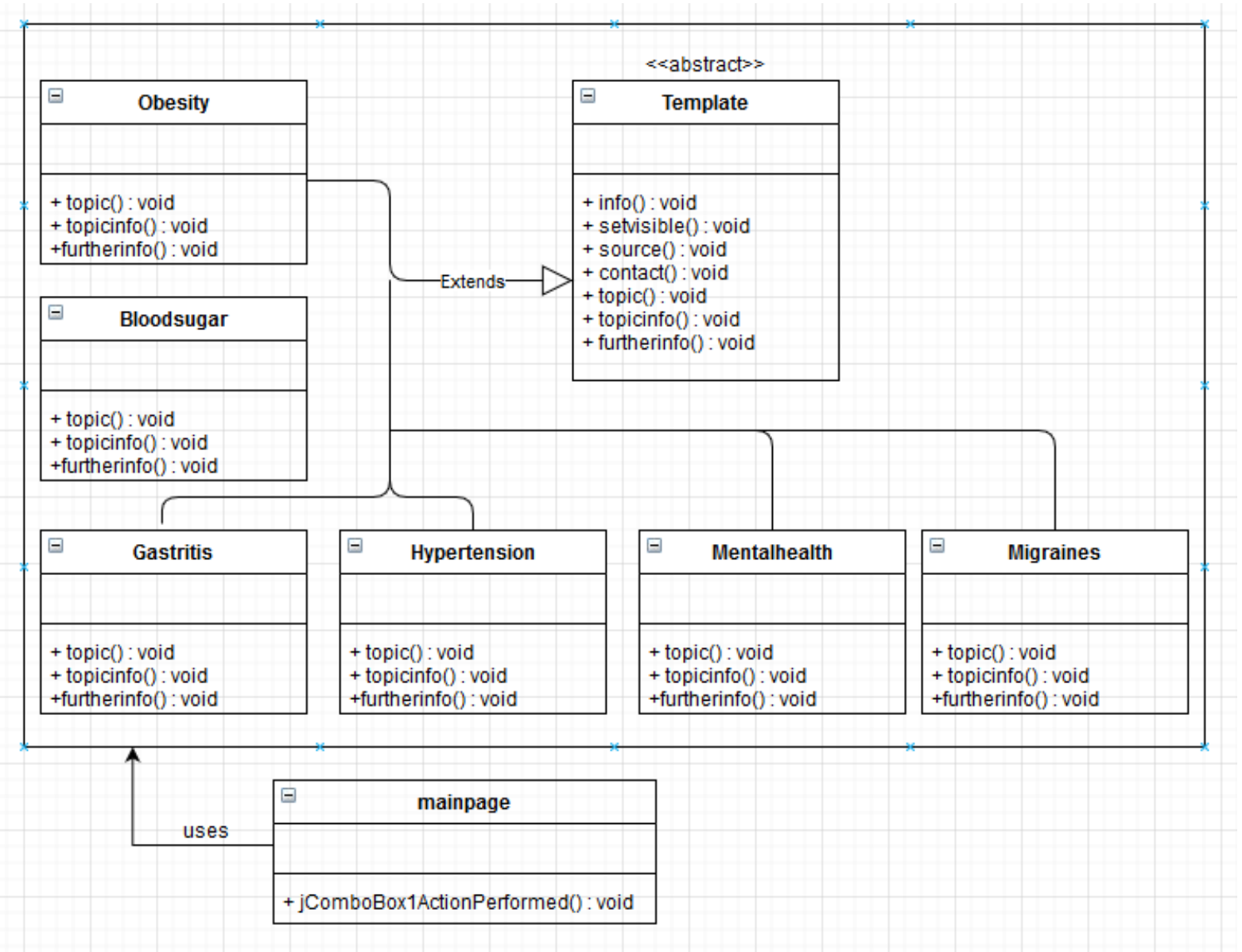


Figure 3. Template Design Pattern (Ailment Info)

Template design pattern falls under the category of behavioral pattern because its main purpose is to eliminate the repetition of certain aspects of a method by giving default values that are common to all in the abstract class, and then overriding the method implementations per subclass with their own unique values and using them.

In the Extra Life Health application, the ‘Ailment info’ feature displays in-depth information about common health ailments and diseases that people face day to day.

However, all the information provided in the feature were taken from reputable online sources, thus they were displayed for all. Along with contact information to the fictional ‘Extra Life Health’ support organization for feedback.

Since all information provided in the ailment info are credited to the sources and provide support details, there is no reason to constantly enter the information per class. Instead, they will be in the abstract class and display for all selections, whereas the subclasses will provide unique information per ailment.

Making the Template design pattern the best choice for this feature to eliminate code repetition for things that are common to all.

Façade Design Pattern (Sign up)

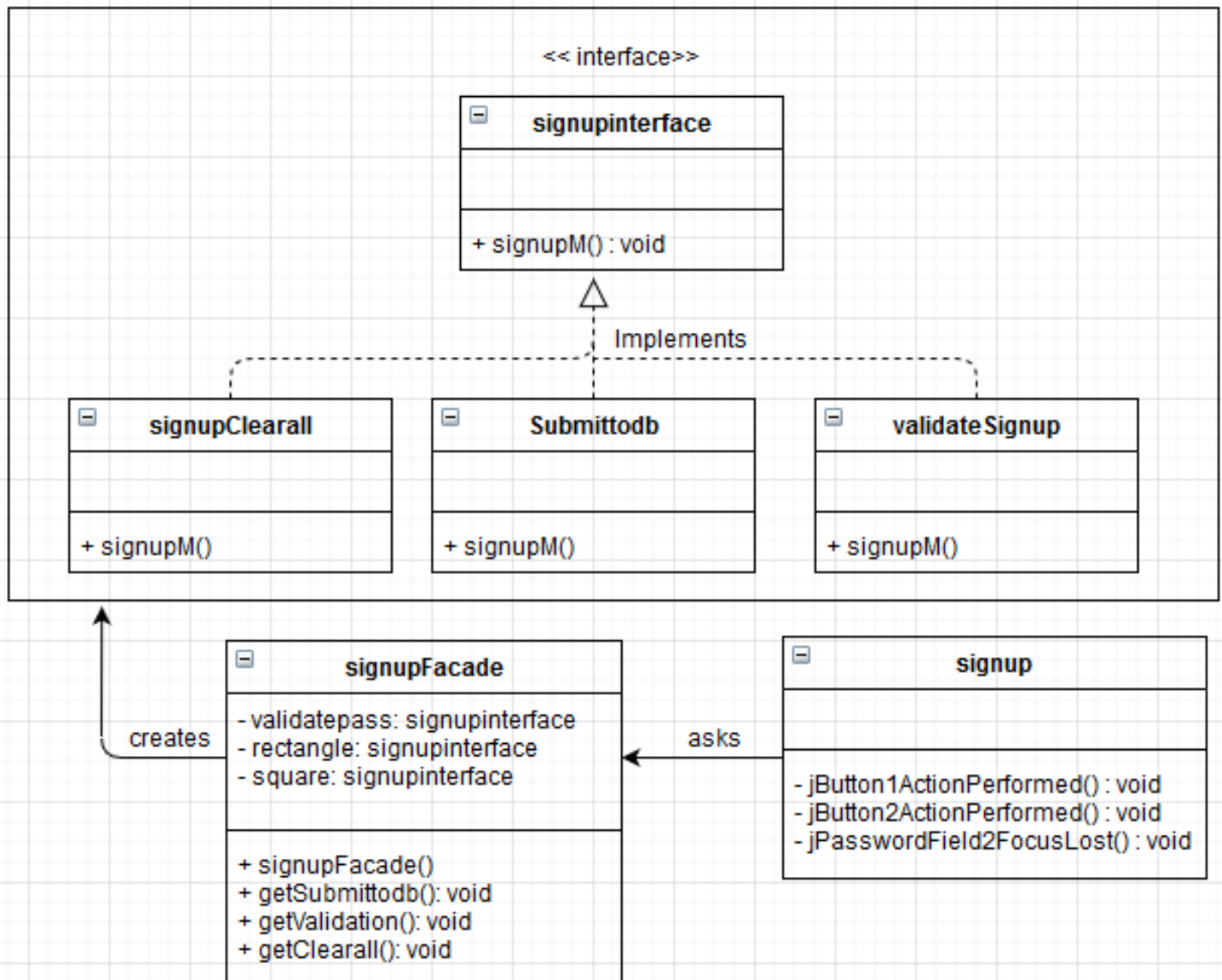


Figure 4. Façade Design Pattern (Sign up)

Façade design pattern falls under structural design pattern and its main purpose is to hide the intricate details or the complexities from the user. Instead, an interface is given so that the user can access the necessary systems or features they require in one place.

Since the façade design pattern's main purpose is to hide the complexities of a system, for the health-based application I have used it to hide the functions/actions from the user for the sign-up page.

This includes clearing everything that is already entered in the `getTextFields`, to validating if the entered password is similar to the confirm password field, and most importantly submitting the entered information to the database to create an account so that the user may access the application with it.

Due to this, things are made much simpler in design in the GUI side of things for the sign-up class, since all we must do is create an object of the façade class and use it to call the necessary features we want to place.

Due to this, façade design pattern is the better choice to hide the complexities of a code in separate classes. It is one of the most simplest and most used design patterns in object oriented programming.

Proxy Design Pattern (Application Users List)

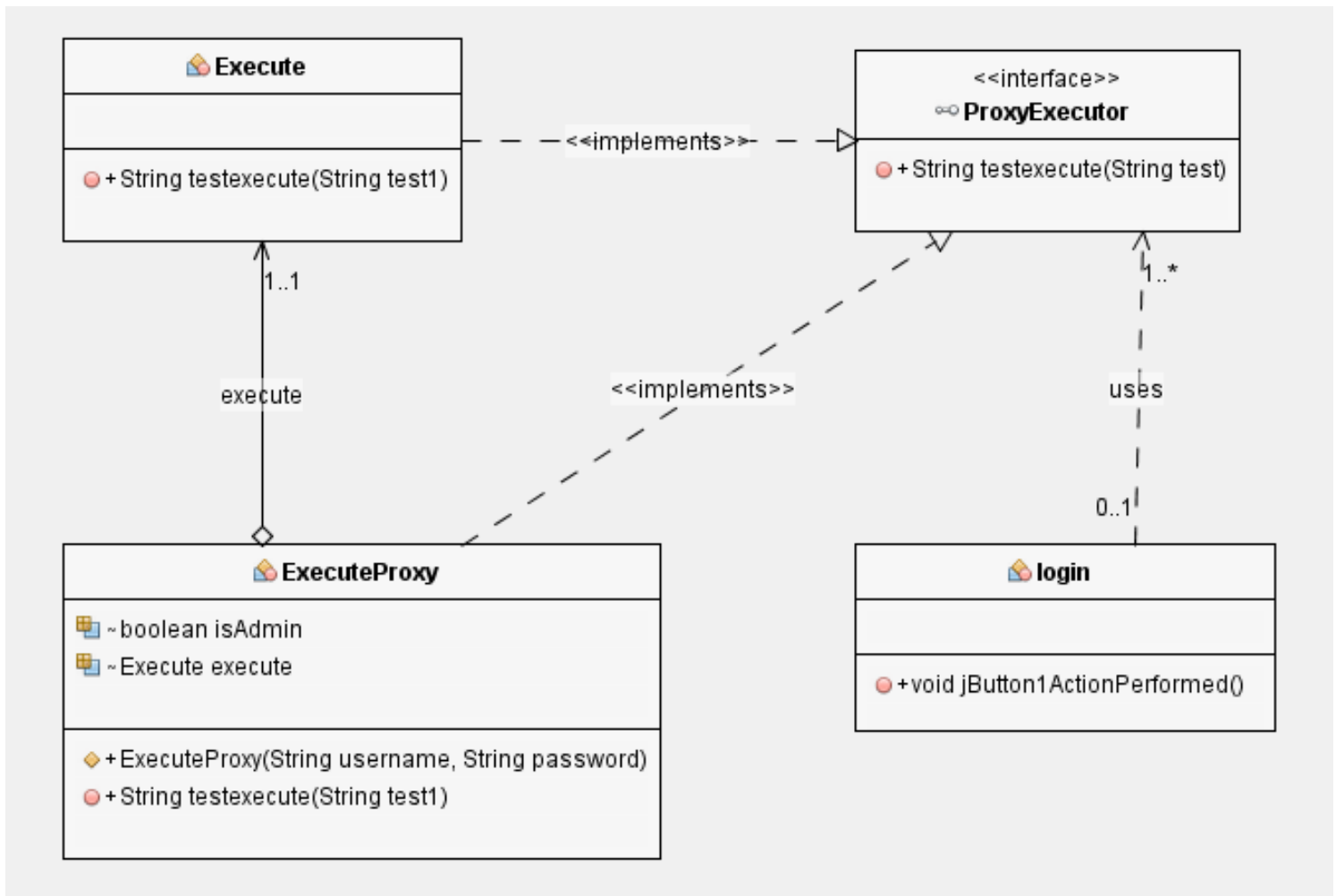


Figure 5. Proxy Design Pattern (Application Users List)

Another addition to the structural design pattern category is the proxy design pattern. The purpose of this design pattern is to control the access.

In the context of 'Extra Life Health' application I have used the proxy design pattern to control the access of certain features depending on the user account's permissions.

Meaning if the user's account is does not have admin privileges, the application will hide certain features. In this case, it is the list of users that are registered in the application.

Whereas users that have admin privileges will be able to view the list of users in the application.

For this purpose, proxy design pattern is the better choice.

Decorator Design Pattern (Medicine Purchases)

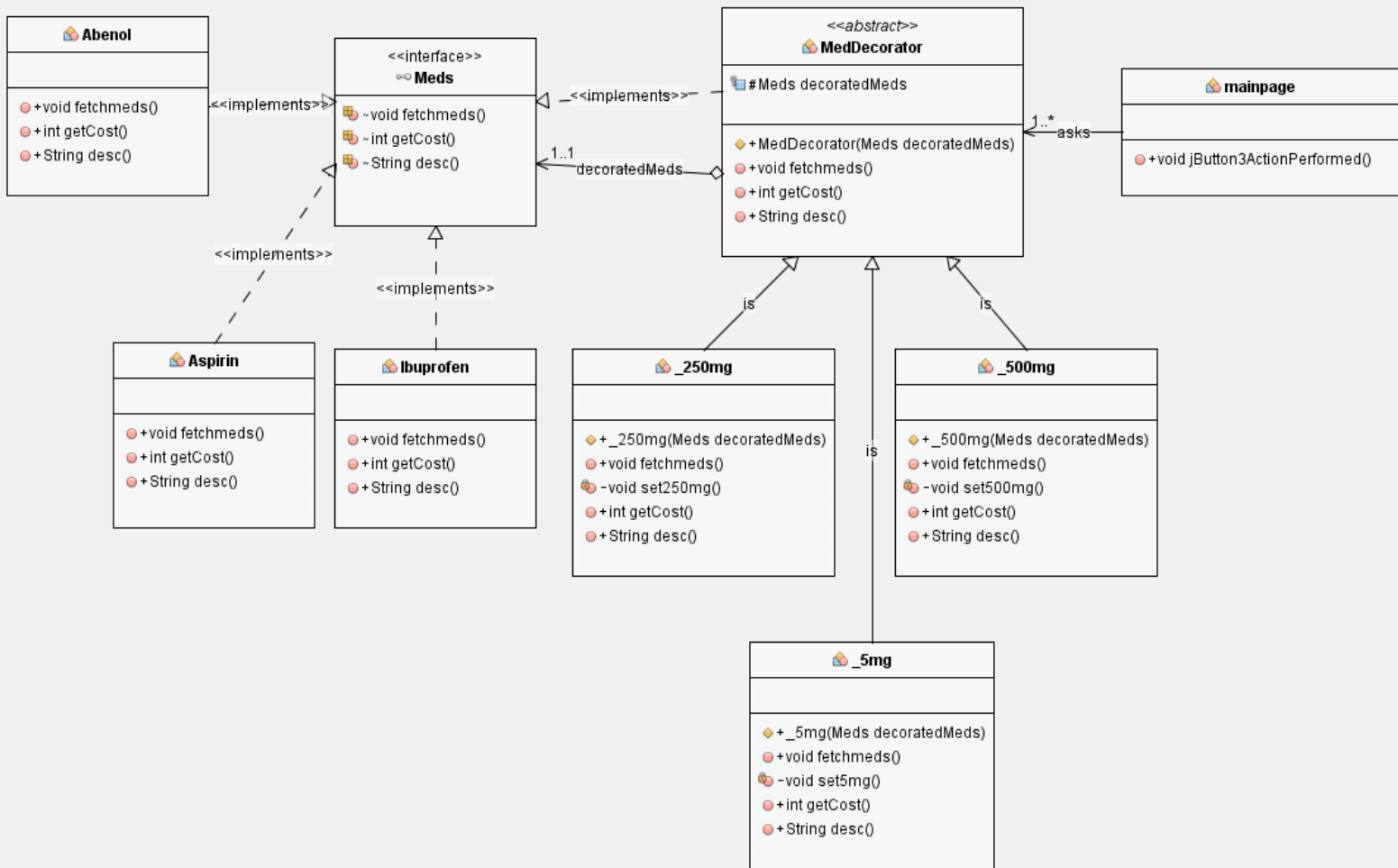


Figure 6. Decorator Design Pattern (Medicine Purchases)

The decorator design pattern comes under the structural design patterns category. The purpose of this design pattern is to add additional usage to already created or existing objects without editing it any further.

In the context of this application the user will be able to select the medicine name they want and depending on the milligram amount of the tablet they want, will be added as tax.

For example:

One box of Ibuprofen = 50rs

250mg tablet tax is = 5rs

If the users decide they want one box of Ibuprofen, the cost will be 55rs total.

Additionally, it will also use the decorator pattern to display information of the selection as “Box of Ibuprofen with Milligram of 250mg per tablet”.

Due to the variety of choice with the user and the tablet amounts the decorator design pattern is the best choice for this feature of the application.

Implementation

Sign Up (Facade)

- signupinterface

```
package Signup_EL_Facade;

public interface signupinterface {

    // will be used for facade pattern on the highest level
    public void signupM();
}
```

Figure 7.1 signupinterface (facade)

A sign up interface is created to be used at the highest level for the façade design pattern. It includes a method called signupM

- signupClearall

```
import javax.swing.JPasswordField;
import javax.swing.JTextField;

public class signupClearall implements signupinterface {

    @Override
    public void signupM() {
        signup.jTextField1.setText("");
        signup.jTextField2.setText("");
        signup.jTextField3.setText("");
        signup.jPasswordField1.setText("");
        signup.jPasswordField2.setText("");
    }
}
```

Figure 8. signupClearall (facade)

Signupclear all class is created that implements the interface with a method that overrides the interface's method with some actions. What this does is when the Signupclear class's method is called, it will reset the text that is already written in the java GUI's text box.

- validateSignup

```
public class validateSignup implements signupinterface{

    @Override
    public void signupM() {

        String pass1 = signup.jPasswordField1.getText();
        String pass2 = signup.jPasswordField2.getText();

        if(!pass1.equals(pass2)) {
            // sets the password border to RED
            signup.jPasswordField2.setBorder(new LineBorder (Color.RED));
        }
    }
}
```

Figure 9. validateSignup (facade)

A validatesignup class is created that implements the interface and overrides its method. Calling this class's method will highlight jpasswordfield2 in the signup page if the entered confirm password by the user does not match the first password entered.

- Submittodb

```
public class Submittodb implements signupinterface{

    Connection connection = null;

    @Override
    public void signupM(){

        try {

            // pulls the getConnection method from the ConnectionManager class for the database.
            // making this a singleton pattern
            connection = ConnectionManager.getConnection();

            String query = "insert into registrations values(?,?,?,?,?)";
            PreparedStatement ps = connection.prepareStatement(query);

            ps.setString(1, signup.jTextField1.getText()); // Name
            ps.setString(2, signup.jTextField2.getText()); // Username
            ps.setString(3, signup.jPasswordField1.getText()); // Password
            ps.setString(4, signup.jPasswordField2.getText()); // Confirm Password
            ps.setString(5, signup.jTextField3.getText()); // Email

            int i = ps.executeUpdate();

            if (i > 0) {
                //JOptionPane.showMessageDialog(this,"Your account has been created and saved!");
                System.out.println("Submission to database successful");
            }

        } catch (SQLException e) {

            System.out.println("Submission to database has failed");

        }

    }

}
```

Figure 10. Submittodb (facade)

Submittodb class is created that implements the interface and overrides its method. This will get a singleton object to get a database connection. It will then read the entered information in the relevant fields in the sign-up GUI and enter it to the database to be registered as a user.

- signupFacade

```

public class signupFacade {

    private signupinterface validatepass;
    private signupinterface clear;
    private signupinterface submittodb;

    public signupFacade() {
        validatepass = new validateSignup();
        clear = new signupClearall();
        submittodb = new Submittodb();
    }

    public void getSubmittodb() {
        submittodb.signupM();
    }

    public void getValidation() {
        validatepass.signupM();
    }
    public void getClearall() {
        clear.signupM();
    }

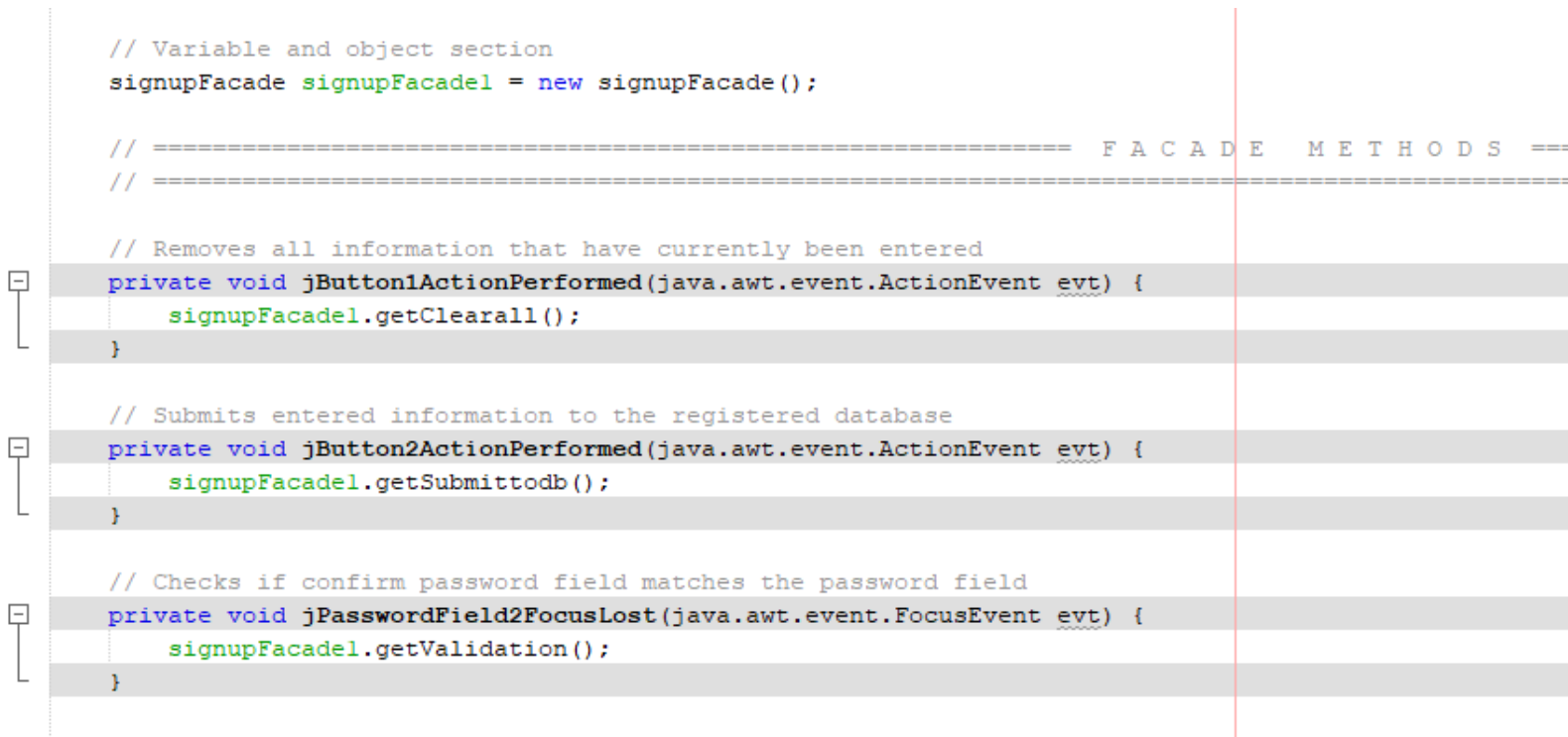
}

```

Figure 11. signupFacade (facade)

Signupfacade class is created. With objects created and linking it to the relevant class's overridden methods.

- signup (facade)



```
// Variable and object section
signupFacade signupFacadel = new signupFacade();

// ===== F A C A D E   M E T H O D S =====
// =====

// Removes all information that have currently been entered
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    signupFacadel.getClearall();
}

// Submits entered information to the registered database
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    signupFacadel.getSubmittodb();
}

// Checks if confirm password field matches the password field
private void jPasswordField2FocusLost(java.awt.event.FocusEvent evt) {
    signupFacadel.getValidation();
}
```

Figure 12. signup (facade)

In the signup class that hosts the swing GUI of the signup, an object of the signupFacade class is created and to the relevant methods, will be called to execute the action.

- It is when jButton1 is clicked, the fields will get reset.
- When jButton2 is clicked, the information existing in the fields will get added into the database.
- When jpasswordfield2 loses its focus, it will check if the password entered jpasswordfield1 matches it. If it does not, the jpasswordfield2 border will be highlighted to signify a mismatch in passwords.

- Façade Output

Test Connection to Remote Database

Name:

Username:

Password:

Confirm Password:

E-mail:

Output - OOA2 (CB007077) (run)

Connection to the database is Successful

Submission to database successful

Figure 13. Façade Output 1

Document	Document	doc1	doc1	testdoc@email.com
Avishka	InvincibleXALE	avishkas12	avishkas12	avis@hotmail.com

Figure 14. Façade Output 2

Connection Manager for Database (Singleton)

- ConnectionManager

```
//Handles the connection to the database
//singleton implementation
public class ConnectionManager {

    private static Connection connection;

    public static Connection getConnection() {

        try {
            connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/cb007077_ooae2","root","123");
            // username = root
            // password = 123

            System.out.println("Connection to the database is Successful");
        }
        catch(Exception e) {
            System.out.println("An error has occurred connecting to the Database");
            e.printStackTrace();
        }
        return connection;
    }
    login log = new login();
}
```

Figure 15. ConnectionManager (Singleton)

The connection manager class is created to handle the database connections all over the application where necessary. It includes the link with the database name and with the root and password for authorization.

XAMPP control panel was also used to achieve this to full potential.

- validate

```
public class validate {

    // Variable section
    Connection connection = null;

    public void getValidation(String textfield, String password){

        // Login object to get fields in Login class
        login log = new login(textfield,password);
        login log2 = new login();

        //Singleton
        connection = ConnectionManager.getConnection();

        try {

            String query = "select Fname, username, pword from registrations where username ='"+ log.getUserName() +"' and pword ='"+ log.getPword() +"'";

            Statement st = connection.createStatement();
            ResultSet set = st.executeQuery(query);

            if(!set.next())
            {
                // if login details do not match
                log.jTextField1.setBorder(new LineBorder (Color.RED)); // changes the border colour
                log.jPasswordField1.setBorder(new LineBorder (Color.RED)); // changes the border colour
            } else {
                // if login information is valid, go to the homepage with options
                mainpage ml = new mainpage();

                ml.welcomeusername.setText(set.getString("Fname"));
                ml.setVisible(true);
            }
        } catch (Exception e){
            System.out.println("An error has occurred");
            e.printStackTrace();
        }
    }
}
```

**** Please zoom in if screenshot is unclear at a glance! ****

The validate class is created to check if the login credentials entered the login page are correct. To check this, we utilize the singleton method of the connection manager class in order to do this.

If the entered information is correct, the login page will be closed, and the user will be moved into the main screen to use the application's features.

Ailment info (Template)

- Template Interface

```
public abstract class Template {  
  
    // Methods to be used for Template design pattern  
    public final void Info(){  
        setVisible(); // To set the invisible labels and table as visible  
        topic();  
        topicinfo();  
        furtherinfo();  
        source();  
        contact();  
    }  
  
    // COMMON FOR ALL  
    public void setVisible(){  
        mainpage.jLabel13.setVisible(true);  
        mainpage.jLabel14.setVisible(true);  
        mainpage.jLabel15.setVisible(true);  
        mainpage.jLabel16.setVisible(true);  
        mainpage.jLabel17.setVisible(true);  
    }  
  
    public void source() {  
        mainpage.jLabel14.setText("<html>All information is provided by valid medical sources, for Cardiovascular, hereditary diseases and other Physiological illnesses. "  
        + "Sources such as WEBMD and the American Heart Health Association were used.</html>");  
    }  
  
    public void contact() {  
        mainpage.jLabel15.setText("Please contact us at support@extralife.org to provide feedback on '" + mainpage.jComboBox1.getSelectedItem().toString() + "' information accuracy.");  
    }  
  
    // Unique methods per class  
    public abstract void topic();  
    public abstract void topicinfo();  
    public abstract void furtherinfo();  
}
```

Figure 16. Template Interface (Template)

**** Please zoom in if screenshot is unclear at a glance! ****

Template interface is created with the source and contact methods having default entries.

These entries will be ones that are common to all sub classes.

The ones that are unique to each sub class, are created below with the names setVisible (which will set the relevant invisible jlabels to visible when executing) topic, topic info and further info.

All those methods are then wrapped into a final method called Info. This is the basic structure of the template.

- Obesity

```
public class Obesity extends Template{

    @Override
    public void topic() {
        mainpage.jLabel6.setText("<html> <u> Overweight and Obesity </u> </html>");
    }

    @Override
    public void topicinfo() {
        mainpage.jLabel7.setText("<html>The term 'Obesity' describes a person that is overweight in terms of body fat. A person is usually diagnosed with obesity by checking their height and weight on a BMI (Body Mass Index) and seeing if they are overweight. <br><br> <br></br> "
        + "The main cause of obesity is consuming high fatty and calorie foods with sugar and not having enough physical activity throughout the day. Suffering from obesity can
        + "long term issues such as High Blood Pressure, Cardiovascular diseases, stroke or heart attack, liver and kidney diseases asthma, sleep apnea, etc. "
        + "<br></br>"
        + "If nothing is done to circumvent or work towards losing weight, there is very high risk of permanent damage to your body, or even loss of life. "
        + "<br></br> "
        + "<b><i> It is vital for any person to be within' the healthy position of the BMI scale to avoid any of these issues further in life.</i></b></html>");

    }

    @Override
    public void furtherinfo() {
        mainpage.jPanel3.setVisible(false);
        mainpage.jLabel3.setText("<html><u>How to overcome Obesity</u>"
        + "<br></br> <br></br> "
        + "There are many ways to treat obesity to get to a better healthy healthy standing in weight. Such as: "
        + "<ul><li> Take part in more physically active activities such as jogging or walking for up to 1-2 hours a day.</li>"
        + "<li> Go on a balanced diet and monitor your calorie intake and eat more veggies.</li>"
        + "<li> Consume less processed or sugar filled junk foods</li>"
        + "<li> Enroll in a weight loss regiment</li></ul>"
        + "<br></br>"
        + "Preventing obesity can keep you in good health and can also avoid long term chronic health conditions. "
        + "<br></br>"
        + "Even getting to a healthy standing might also result in the control of other Cardiovascular diseases such as High Blood Pressure.</html>");
    }
}
```

Figure 17. Obesity (Template)

**** Please zoom in if screenshot is unclear at a glance! ****

The obesity class sets the relevant unique information by overriding methods from the template interface.

HTML formatting is used to get proper text editing features because JLabels support HTML language.

When the method is called, the panel will show up in the GUI, revealing the table.

- Migraines

```
public class Migraines extends Template {

    @Override
    public void topic() {
        mainpage.jLabel16.setText("<html> <u> Migraines </u> </html>");
    }

    @Override
    public void topicinfo() {
        mainpage.jLabel17.setText("<html> Migraine is a heightened form of a headache that brings feelings of nausea, or vomiting and can go on for many hours or even days. It is very co
        + \"among people who have a very stressful day to day life to experience constant migraines. \"
        + \"<br><br><br></br> The causes of migraines are related to changes in your brain's activity and can \"
        + \"even be triggered through other means such as blood flow to the brain. This can also be involved with fatigue or exposure to changing weather conditions such as a lo
        + \"and sunny morning.</html>");
    }

    @Override
    public void furtherinfo() {
        mainpage.jPanel13.setVisible(false);
        mainpage.jLabel13.setText("<html> <u> Remedies:</u> \"
        + \"<br><br><br></br>\"
        + \"There ways to remedy having a migraine at the comfort of your home,such as:\"
        + \"<br><br>\"
        + \"<ul><li> Resting on a comfortable surface with your eyes closed in a dark surrounding</li>\"
        + \"<li> Consuming plenty of liquids. Especially water.</li>\"
        + \"<li> Placing an ice pack on your forehead for an extended period of time</li>\"
        + \"<li> Drinking Asprin</li></ul>\"
        + \"<br><br> <br><br> If the Migraine still persists:\"
        + \"<b><p style=\\\"color:red;\\\">Please contact your doctor IMMEDIATELY!</p></b> </html>");
    }
}
```

Figure 18. Migraines (Template)

**** Please zoom in if screenshot is unclear at a glance! ****

The Migraines class sets the relevant unique information by overriding methods from the template interface.

- Mentalhealth

```
public class Mentalhealth extends Template {

    @Override
    public void topic() {
        mainpage.jLabel6.setText("<html> <u> Mental Health </u> </html>");
    }

    @Override
    public void topicinfo() {
        mainpage.jLabel7.setText("<html> Mental health is something majority of people in the world today suffer from be it either emotional, psychological or social well-being. It play
        + \"big part in stress and influences our ability to carry out day to day tasks and emotional activities. Due to that, having a good mental health standing is vital for
        + \"any growing human being. <br></br><br></br>Most common of all the causes is due to life experiences but other biological factors can play into this as well. <br></br>
        + \"Because its a condition that can effect a person's day to day life it is vital that you get professional assistance as soon as possible if you are showing early\"
        + \" signs of a mental health issue. </html>");
    }

    @Override
    public void furtherinfo() {
        mainpage.jPanel3.setVisible(false);
        mainpage.jLabel3.setText("<html><u><b>What are the early warning signs?</b></u> <br></br>\"
        + \"<ul><li> Constant Quarelling with family or friends.</li>\"
        + \"<li> Considering self harm.</li>\"
        + \"<li> Feeling of helplessness of hopelessness.</li>\"
        + \"<li> Experiencing random mood swings.</li>\"
        + \"<li> Inability to perform daily tasks.</li>\"
        + \"<li> Feeling of numbness.</li></ul>\"
        + \"<br></br>\"
        + \"To overcome these feelings it is recommended to see a medical professional at your earliest convenience to get proper medication and guidance on how to proceed with
        + \"It helps to also practice with mediation and other recreational activities to circumvent mental health issues and better your daily health.</html>");
    }
}
```

Figure 19. Mentalhealth (Template)

**** Please zoom in if screenshot is unclear at a glance! ****

The Mentalhealth class sets the relevant unique information by overriding methods from the template interface.

- Hypertension

```
// Part of template pattern
public class Hypertension extends Template {

    @Override
    public void topic() {
        mainpage.jLabel6.setText("<html> <u>Hypertension / High Blood Pressure</u> </html>");
    }

    @Override
    public void topicinfo() {
        mainpage.jLabel7.setText("<html>Hypertension, or more commonly known as High Blood pressure is called a 'silent killer'. Most adults live out the most of their lives "
            + "without even knowing they are suffering from it.<br/> It is an ailment that shows little to no symptoms in the longrun and are common with obese or elderly people."
            + "The cause of Hypertension is when your blood pressure is higher than normal and can damage your blood vessels and arteries, which could lead to "
            + "a heart attack or stroke in the long term. However with constant monitoring & diet control with medication, it can be controlled. <br> </br> "
            + "<br> </br> Below are readings for high BP:</html>");
    }

    @Override
    public void furtherinfo() {
        mainpage.jPanel3.setVisible(true);
        mainpage.jLabel3.setText("<html>If your Blood Pressure readings are in the Hypertensive Crisis threshold: <b><p style='color:red;'>Please contact your doctor IMMEDIATELY!</p></html>");
    }
}
```

Figure 20. Hypertension (Template)

**** Please zoom in if screenshot is unclear at a glance! ****

The Hypertension class sets the relevant unique information by overriding methods from the template interface.

A jpanel containing a table with high blood pressure readings will be set to visible as well. This is because jTable cannot turn its full visibility off. So instead, it is wrapped in a jpanel and the panel is made invisible.

- Gastritis

```
public class Gastritis extends Template {

    @Override
    public void topic() {
        mainpage.jLabel6.setText("<html> <u> Gastritis </u> </html>");
    }

    @Override
    public void topicinfo() {
        mainpage.jLabel7.setText("<html> Gastritis is an inflammation or irritation of the stomach and can bring pain or discomfort to the person experiencing it. <br><br>"
            + "It can be caused by excessive alcohol consumption, stress, stomach infections or incorrect use of medicine or even consumption of food that didn't sit well.<br><br>"
            + "Gastritis is diagnosed from your doctor after viewing relecant family and medical history with a physical examination. "
            + " <br><br> <br><br> <i> The tests are an Upper Endoscopy, Blood test and a Fecal occult blood test.</i> </html>");
    }

    @Override
    public void furtherinfo() {
        mainpage.jPanel3.setVisible(false);
        mainpage.jLabel13.setText("<html> <u> Gastritis Symptoms </u> "
            + "<ul><li> Abdominal Pain</li>"
            + "<li> Vomiting</li>"
            + "<li> Abdominal Bloating</li>"
            + "<li> Burning sensation in the stomach</li>"
            + "<li> Loss of appetite</li>"
            + "<li> Nausea or upset stomach</li></ul>"
            + "<br><br>"
            + "Treatment for Gastritis usually includes taking Antiacids or avoiding the consumption of spicy or hot food. <br><br>"
            + "Medication is also recommended to recover during time at home.</html>");
    }
}
```

Figure 21. Gastritis (Template)

**** Please zoom in if screenshot is unclear at a glance! ****

The Gastritis class sets the relevant unique information by overriding methods from the template interface.

- Bloodsugar

```
// Part of template pattern
public class Bloodsugar extends Template{

    @Override
    public void topic(){
        mainpage.jLabel16.setText("<html> <u>Blood Sugar Management</u> </html>");
    }

    @Override
    public void topicinfo() {
        mainpage.jLabel17.setText("<html> Also known as Blood Glucose, this can damage the person overtime along the years if they have a history of diabetes. </br> It can cause proble
        + \"kidney failiure, weakened immune system, nerve damage strokes, vision loss etc. It is imperitive to control your blood sugar levels through dieting with lesss carbo
        + \" and constant exercise to keep yourself at a stable level to prevent long term damage to your body. \"
        + \"<br></br> <br></br> Doctors diagnose a paient with diabetes by checking 3 types of tests called: \"
        + \" <ul><li>Fasting plasma glucose test - Tests are carried out to check if your blood sugar levels after are higher than 126mg/dL after 8 hours</li> \"
        + \"<li>Oral glucose tolerance test - The patient is given a sugar heavy drink after 8 hours of fasting to check if the sugar level is over 200</li> \"
        + \"<li>Random check - It is checked if the patient is urinating more, constantly thirsty or lost a lot of weight, and will be done a random of the two above tests.</li>
    }

    @Override
    public void furtherinfo(){
        mainpage.jPanel13.setVisible(false);
        mainpage.jLabel13.setText("<html><u><b>How do I control my Blood Sugar levels?</b></u><br></br><br></br> It is possible to test your blood sugar yourself at home by using a Glu
        + \"It is considered to be a normal level if your readings are below 100mg/dL after fasting for 8 hours and less than 140mg/DL after 2 hours after eating. \"
        + \"<br></br><br></br>Levels tend to be the lowest before their meal routine per day, and varies with normal levels for people that do not suffer with diabetes.</html>\"
    }
}
```

Figure 22. Bloodsugar (Template)

**** Please zoom in if screenshot is unclear at a glance! ****

The Bloodsugar class sets the relevant unique information by overriding methods from the template interface.

- Mainpage (Template)

```
// TEMPLATE PATTERN METHODS
Template hypertension = new Hypertension();
Template bloodsugar = new Bloodsugar();
Template mentalhealth = new Mentalhealth();
Template migraines = new Migraines();
Template obesity = new Obesity();
Template gastritis = new Gastritis();

// =====
private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {

    if(evt.getSource() == jComboBox1) {

        String selection = (String)jComboBox1.getSelectedItem();

        switch (selection){

            case "None Selected":
                setInvisible();
                break;

            case "Hypertension or High Blood Pressure":
                hypertension.Info();
                break;

            case "Blood Sugar":
                bloodsugar.Info();
                break;

            case "Mental Health":
                mentalhealth.Info();
                break;

            case "Migraines":
                migraines.Info();
                break;

            case "Obesity":
                obesity.Info();
                break;

            case "Gastritis":
                gastritis.Info();
                break;

        }

    }

}
```

Figure 23. Mainpage (Template)

Here, the user will be able to select from a drop-down menu the information they want to receive based on the ailment.

When the ailment is selected from the drop down, the relevant template class method is called after creating the needed objects. When done so, the unique information will be displayed per method called.

- Template Output

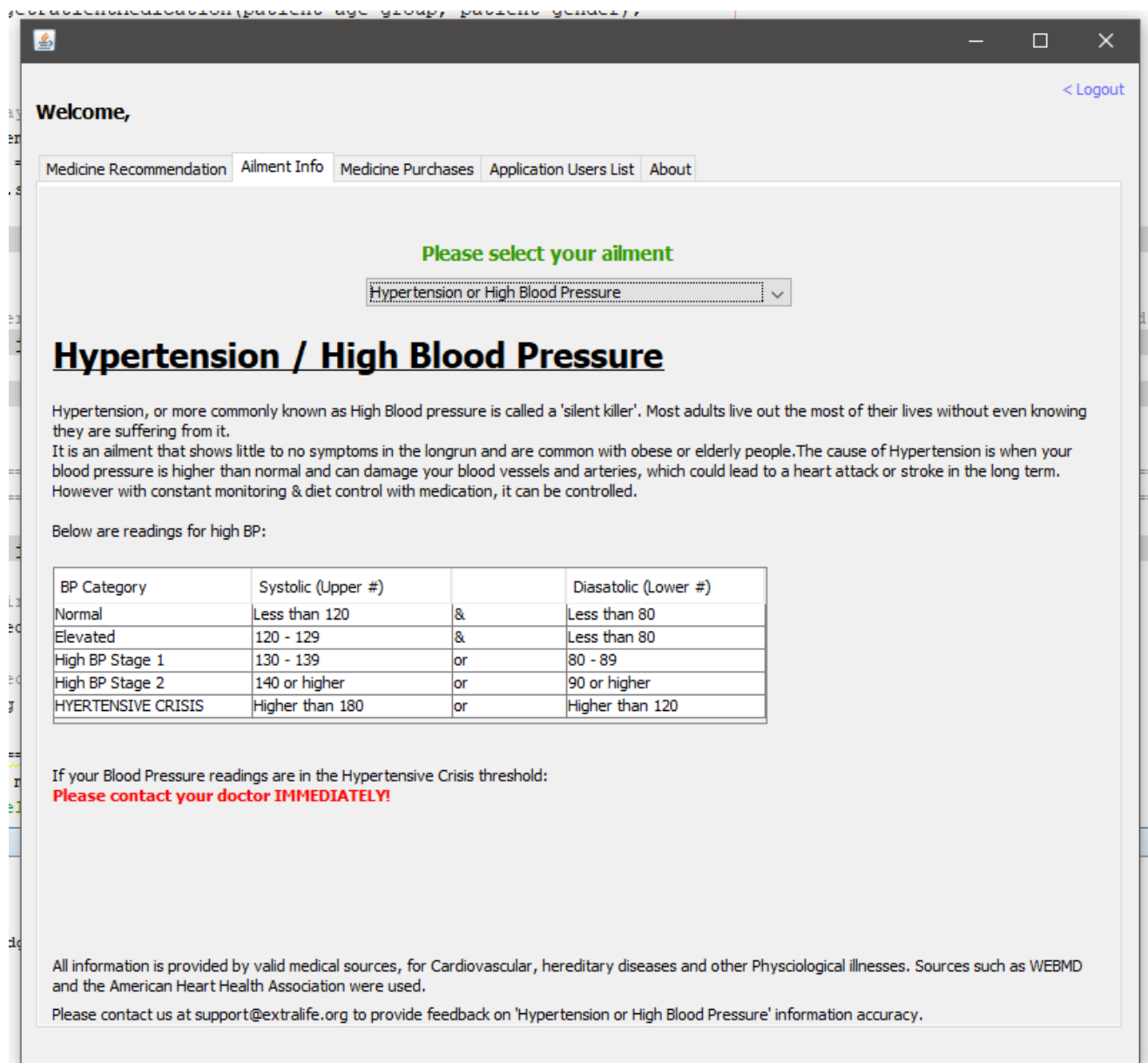


Figure 24. Template Output 1

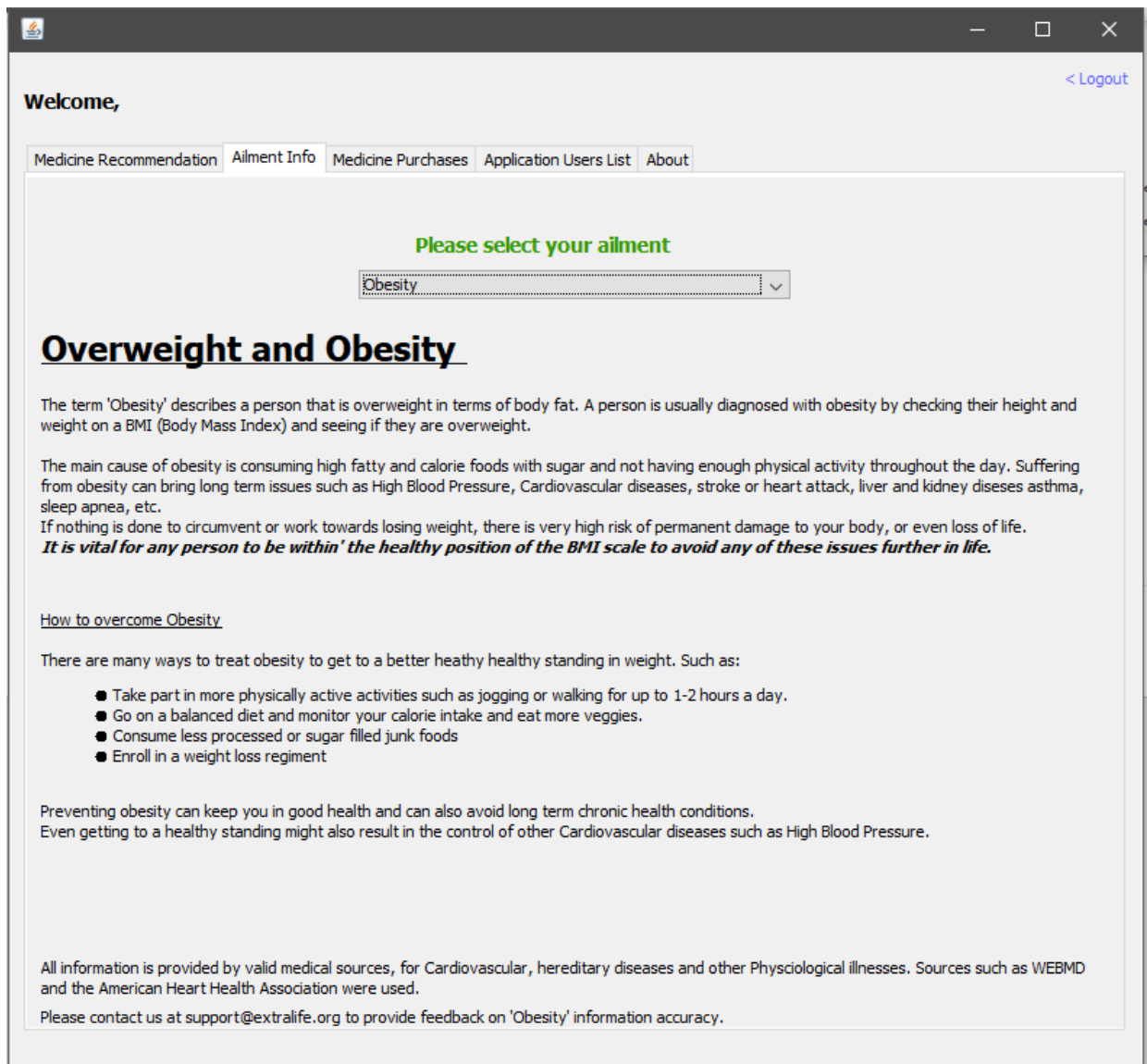


Figure 25. Template output 2

The rest of the items in the drop-down menu takes the same format.

The last 3 lines are the texts that are common to all.
 Displaying the template pattern in use for the ailment info feature of the application.

Medicine Recommendation (Factory Pattern)

Patient

```
public interface Patient
{
    // Setters for common Patient Questions
    void setAgeGroup(int age);

    void setGender(char gender);

    // Message Generating
    void generateHealthMessage();
}
```

Figure 26. Patient (Factory)

The interface that will be used for the factory pattern, creates three methods.

- Patient_head

```
public class Patient_Head implements Patient{

    // Basic Patient Information
    private int patient_age_group;
    private char patient_gender;

    // Head based values
    private int head_pain_time;
    private int discomfort_level;

    // Variable to hold message
    private String medical_message;

    // Constructor
    public Patient_Head(int patient_age_group, char patient_gender, int head_pain_time, int discomfort_level)
    {
        this.patient_age_group = patient_age_group;
        this.patient_gender = patient_gender;
        this.head_pain_time = head_pain_time;
        this.discomfort_level = discomfort_level;
        generateHealthMessage();
    }

    // Setters for Common Patient Questions
    @Override
    public void setAgeGroup(int age)
    {
        this.patient_age_group = age;
    }

    @Override
    public void setGender(char gender)
    {
        this.patient_gender = gender;
    }

    // Setters for Head Based Questions

    public void set_Head_Pain_Time(int time)
    {
        this.head_pain_time = time;
    }
}
```

Here are the constructor and getters and setters of the patient_head class.


```

@Override
public void generateHealthMessage()
{
    switch (patient_gender)
    {
        case 'M':
            switch (patient_age_group)
            {
                case 1:
                    switch (head_pain_time)
                    {
                        case 1:
                            switch (discomfort_level)
                            {
                                case 1:
                                    medical_message = "Based on your choice's we recommend Aspirin";
                                    break;
                                case 2:
                                    medical_message = "Based on your choice's we recommend Ibuprofen";
                                    break;
                                case 3:
                                    medical_message = "Based on your choice's we recommend Naproxen";
                                    break;
                            }
                            break;
                        case 2:
                            switch (discomfort_level)
                            {
                                case 1:
                                    medical_message = "Based on your choice's we recommend Ibuprofen";
                                    break;
                                case 2:
                                    medical_message = "Based on your choice's we recommend Fenoprofen";
                                    break;
                                case 3:
                                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                                    break;
                            }
                            break;
                    }
                break;
            }
        break;
    }
}

```

```

        break;
        case 3:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Ibuprofen";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Methocarbamol";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
            }
            break;
        }
        break;
        case 2:
            switch (head_pain_time)
            {
                case 1:
                    switch (discomfort_level)
                    {
                        case 1:
                            medical_message = "Based on your choice's we recommend Aspirin";
                            break;
                        case 2:
                            medical_message = "Based on your choice's we recommend Acetaminophen";
                            break;
                        case 3:
                            medical_message = "Based on your choice's we recommend Ibuprofen";
                            break;
                    }
                break;
            }
        break;
    }
}

```

2000

```

        break;
    case 2:
        switch (discomfort_level)
        {
            case 1:
                medical_message = "Based on your choice's we recommend Ibuprofen";
                break;
            case 2:
                medical_message = "Based on your choice's we recommend Cybclorazaprine";
                break;
            case 3:
                medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                break;
        }
        break;
    case 3:
        switch (discomfort_level)
        {
            case 1:
                medical_message = "Based on your choice's we recommend Ibuprofen";
                break;
            case 2:
                medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                break;
            case 3:
                medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                break;
        }
        break;
    }
    break;
}
break;

```

```

case 'F':
{
    switch (patient_age_group)
    {
        case 1:
            switch (head_pain_time)
            {
                case 1:
                    switch (discomfort_level)
                    {
                        case 1:
                            medical_message = "Based on your choice's we recommend Aspirin";
                            break;
                        case 2:
                            medical_message = "Based on your choice's we recommend Ibuprofen";
                            break;
                        case 3:
                            medical_message = "Based on your choice's we recommend Naproxen";
                            break;
                    }
                    break;
                case 2:
                    switch (discomfort_level)
                    {
                        case 1:
                            medical_message = "Based on your choice's we recommend Ibuprofen";
                            break;
                        case 2:
                            medical_message = "Based on your choice's we recommend Ketoprofen";
                            break;
                        case 3:
                            medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                            break;
                    }
                    break;
            }
        }
    }
}

```

```

case 3:
    switch (discomfort_level)
    {
        case 1:
            medical_message = "Based on your choice's we recommend Ibuprofen";
            break;
        case 2:
            medical_message = "Based on your choice's we recommend Ketorolac";
            break;
        case 3:
            medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
            break;
    }
    break;
}
break;
case 2:
    switch (head_pain_time)
    {
        case 1:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Diclofenac";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Acetaminophen";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Ibuprofen";
                    break;
            }
            break;
        case 2:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Ibuprofen";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Methocarbamol";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
            }
    }
}

```


The patient head class deals with the switch case selections with the head area of pain.

It checks for gender, discomfort level, age group, pain's amount.

Depending on the choice, recommendation of either medicine or professional guidance is given.

- Patient_lower_body

```
public class Patient_Lower_Body implements Patient
{
    // Basic Patient Information
    private int patient_age_group;
    private char patient_gender;

    // Upper Body based values
    private int lower_body_pain_time;
    private int discomfort_level;

    // Variable to hold message
    private String medical_message;

    // Constructor
    public Patient_Lower_Body(int patient_age_group, char patient_gender, int lower_body_pain_time, int discomfort_level)
    {
        this.patient_age_group = patient_age_group;
        this.patient_gender = patient_gender;
        this.lower_body_pain_time = lower_body_pain_time;
        this.discomfort_level = discomfort_level;
        generateHealthMessage();
    }

    // Setters for Common Patient Questions
    @Override
    public void setAgeGroup(int age)
    {
        this.patient_age_group = age;
    }

    @Override
    public void setGender(char gender)
    {
        this.patient_gender = gender;
    }

    // Setters for lower body Based Questions

    public void set_Lower_Body_Pain_Time(int time)
    {
        this.lower_body_pain_time = time;
    }

    public void set_Lower_Body_Discomfort_Level(int level)
    {
        this.discomfort_level = level;
    }
}
```

```
@Override
public void generateHealthMessage()
{
    switch (patient_gender)
    {
        case 'M':
            switch (patient_age_group)
            {
                case 1:
                    switch (lower_body_pain_time)
                    {
                        case 1:
                            switch (discomfort_level)
                            {
                                case 1:
                                    medical_message = "Based on your choice's we recommend ibuprofen";
                                    break;
                                case 2:
                                    medical_message = "Based on your choice's we recommend naproxen";
                                    break;
                                case 3:
                                    medical_message = "Based on your choice's we recommend acetaminophen";
                                    break;
                            }
                            break;
                        case 2:
                            switch (discomfort_level)
                            {
                                case 1:
                                    medical_message = "Based on your choice's we recommend ibuprofen";
                                    break;
                                case 2:
                                    medical_message = "Based on your choice's we recommend diclofenac";
                                    break;
                                case 3:
                                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                                    break;
                            }
                            break;
                    }
                break;
            }
        break;
    }
}
```



```

case 3:
    switch (discomfort_level)
    {
        case 1:
            medical_message = "Based on your choice's we recommend Voltaren";
            break;
        case 2:
            medical_message = "Based on your choice's we recommend Advil";
            break;
        case 3:
            medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
            break;
    }
    break;
}
break;
case 2:
    switch (lower_body_pain_time)
    {
        case 1:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Benazepril";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Captopril";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Enalapril";
                    break;
            }
            break;
        case 2:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend ibuprofen";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Anaprox";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
            }
    }
}

```

```

        break;
    case 3:
        switch (discomfort_level)
        {
            case 1:
                medical_message = "Based on your choice's we recommend Naprelan";
                break;
            case 2:
                medical_message = "Based on your choice's we recommend Mapap";
                break;
            case 3:
                medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                break;
        }
        break;
    }
    break;
case 3:
    switch (lower_body_pain_time)
    {
        case 1:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Quinapril";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Ramipril";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Trandolapril ";
                    break;
            }
            break;
        case 2:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend ibuprofen";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend acetaminophen";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
            }
        }
    }
}

```

```

        case 3:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Advil";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
            }
            break;
        }
        break;
    }
    break;
case 'F':
{
    switch (patient_age_group)
    {
        case 1:
            switch (lower_body_pain_time)
            {
                case 1:
                    switch (discomfort_level)
                    {
                        case 1:
                            medical_message = "Based on your choice's we recommend ibuprofen";
                            break;
                        case 2:
                            medical_message = "Based on your choice's we recommend diclofenac";
                            break;
                        case 3:
                            medical_message = "Based on your choice's we recommend acetaminophenn";
                            break;
                    }
                    break;
                case 2:
                    switch (discomfort_level)
                    {
                        case 1:
                            medical_message = "Based on your choice's we recommend Paracetamol";
                            break;
                    }
            }
        }
    }
}

```

```

case 2:
    switch (discomfort_level)
    {
        case 1:
            medical_message = "Based on your choice's we recommend Paracetamol";
            break;
        case 2:
            medical_message = "Based on your choice's we recommend Aleve";
            break;
        case 3:
            medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
            break;
    }
    break;
case 3:
    switch (discomfort_level)
    {
        case 1:
            medical_message = "Based on your choice's we recommend Voltaren";
            break;
        case 2:
            medical_message = "Based on your choice's we recommend Motrin";
            break;
        case 3:
            medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
            break;
    }
    break;
}
break;
case 2:
    switch (lower_body_pain_time)
    {
        case 1:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Abenol";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Actamin";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Addaprin";
                    break;
            }
        case 2:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Paracetamol";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Aleve";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
            }
        case 3:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Voltaren";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Motrin";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
            }
    }
    break;
}
break;

```

```

case 2:
    switch (discomfort_level)
    {
        case 1:
            medical_message = "Based on your choice's we recommend Apra";
            break;
        case 2:
            medical_message = "Based on your choice's we recommend Cetafen";
            break;
        case 3:
            medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
            break;
    }
    break;
case 3:
    switch (discomfort_level)
    {
        case 1:
            medical_message = "Based on your choice's we recommend Genpril";
            break;
        case 2:
            medical_message = "Based on your choice's we recommend Nuprin";
            break;
        case 3:
            medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
            break;
    }
    break;
}
break;
case 3:
    switch (lower_body_pain_time)
    {
        case 1:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Aflaxen";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Nuprin";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend EC-Naprosyn";
                    break;
            }
        .
    }

```

```

        case 2:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Aflexeryl-MC";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Monopril";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
            }
            break;
        case 3:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Nuprin";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
            }
            break;
    }
    break;
}
break;
}
}
}

```

Figure 28. Patient_lower_body (factory)

The patient lower body class deals with the switch case selections with the head area of pain.

It checks for gender, discomfort level, age group, pain's amount.

Depending on the choice, recommendation of either medicine or professional guidance is given.

- Patient upper body

```
public class Patient_Upper_Body implements Patient
{
    // Basic Patient Information
    private int patient_age_group;
    private char patient_gender;

    // Upper Body based values
    private int upper_body_pain_time;
    private int discomfort_level;

    // Variable to hold message
    private String medical_message;

    // Constructor
    public Patient_Upper_Body(int patient_age_group, char patient_gender, int upper_body_pain_time, int discomfort_level)
    {
        this.patient_age_group = patient_age_group;
        this.patient_gender = patient_gender;
        this.upper_body_pain_time = upper_body_pain_time;
        this.discomfort_level = discomfort_level;
        generateHealthMessage();
    }

    // Setters for Common Patient Questions
    @Override
    public void setAgeGroup(int age)
    {
        this.patient_age_group = age;
    }

    @Override
    public void setGender(char gender)
    {
        this.patient_gender = gender;
    }

    // Setters for upper Based Questions

    public void set_Upper_Body_Pain_Time(int time)
    {
        this.upper_body_pain_time = time;
    }

    public void set_Upper_Body_Discomfort_Level(int level)
    {
        this.discomfort_level = level;
    }
}
```

```

@Override
public void generateHealthMessage()
{
    switch (patient_gender)
    {
        case 'M':
            switch (patient_age_group)
            {
                case 1:
                    switch (upper_body_pain_time)
                    {
                        case 1:
                            switch (discomfort_level)
                            {
                                case 1:
                                    medical_message = "Based on your choice's we recommend Nitroglycerin";
                                    break;
                                case 2:
                                    medical_message = "Based on your choice's we recommend Clopidogrel";
                                    break;
                                case 3:
                                    medical_message = "Based on your choice's we recommend Dipyridamole";
                                    break;
                            }
                        break;
                    }
                case 2:
                    switch (discomfort_level)
                    {
                        case 1:
                            medical_message = "Based on your choice's we recommend Nitroglycerin";
                            break;
                        case 2:
                            medical_message = "Based on your choice's we recommend Dipyridamole";
                            break;
                        case 3:
                            medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                            break;
                    }
                break;
            }
        }
    }
}

```



```

case 3:
    switch (discomfort_level)
    {
        case 1:
            medical_message = "Based on your choice's we recommend Nitroglycerin";
            break;
        case 2:
            medical_message = "Based on your choice's we recommend Prasugrel";
            break;
        case 3:
            medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
            break;
    }
    break;
}
break;
case 2:
    switch (upper_body_pain_time)
    {
        case 1:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Benazepril";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Captopril";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Enalapril";
                    break;
            }
            break;
        case 2:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Fosinopril ";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Lisinopril";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
            }
            break;
        case 3:

```

```

        break;
    case 3:
        switch (discomfort_level)
        {
            case 1:
                medical_message = "Based on your choice's we recommend Perindopril";
                break;
            case 2:
                medical_message = "Based on your choice's we recommend Moexipril";
                break;
            case 3:
                medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                break;
        }
        break;
    }
    break;
case 3:
    switch (upper_body_pain_time)
    {
        case 1:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Quinapril";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Ramipril";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Trandolapril ";
                    break;
            }
            break;
        case 2:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Azilsartan";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Candesartane";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
            }
        }
    }
}

```

```

case 2:
    switch (discomfort_level)
    {
        case 1:
            medical_message = "Based on your choice's we recommend Azilsartan";
            break;
        case 2:
            medical_message = "Based on your choice's we recommend Candesartane";
            break;
        case 3:
            medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
            break;
    }
    break;
case 3:
    switch (discomfort_level)
    {
        case 1:
            medical_message = "Based on your choice's we recommend Telmisartan";
            break;
        case 2:
            medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
            break;
        case 3:
            medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
            break;
    }
    break;
}
break;

```

```

        break;
    case 'F':
    {
        switch (patient_age_group)
        {
            case 1:
                switch (upper_body_pain_time)
                {
                    case 1:
                        switch (discomfort_level)
                        {
                            case 1:
                                medical_message = "Based on your choice's we recommend Valsartan";
                                break;
                            case 2:
                                medical_message = "Based on your choice's we recommend Azilsartan";
                                break;
                            case 3:
                                medical_message = "Based on your choice's we recommend Naproxen";
                                break;
                        }
                    break;
                    case 2:
                        switch (discomfort_level)
                        {
                            case 1:
                                medical_message = "Based on your choice's we recommend Eprosartan";
                                break;
                            case 2:
                                medical_message = "Based on your choice's we recommend Irbesartan ";
                                break;
                            case 3:
                                medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                                break;
                        }
                    break;
                    case 3:
                        switch (discomfort_level)
                        {
                            case 1:
                                medical_message = "Based on your choice's we recommend Azilsartan";
                                break;
                            case 2:
                                medical_message = "Based on your choice's we recommend Telmisartan";
                                break;
                            case 3:
                                medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";

```

```

case 2:
    switch (upper_body_pain_time)
    {
        case 1:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Benazepril";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Captopril";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Enalapril";
                    break;
            }
            break;
        case 2:
            switch (discomfort_level)
            {
                case 1:
                    medical_message = "Based on your choice's we recommend Fosinopril";
                    break;
                case 2:
                    medical_message = "Based on your choice's we recommend Prinivil";
                    break;
                case 3:
                    medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                    break;
            }
            break;
        case 3:

```

```

        break;
    case 3:
        switch (upper_body_pain_time)
        {
            case 1:
                switch (discomfort_level)
                {
                    case 1:
                        medical_message = "Based on your choice's we recommend Carisprodol";
                        break;
                    case 2:
                        medical_message = "Based on your choice's we recommend Acetaminophen";
                        break;
                    case 3:
                        medical_message = "Based on your choice's we recommend Ibuprofen";
                        break;
                }
                break;
            case 2:
                switch (discomfort_level)
                {
                    case 1:
                        medical_message = "Based on your choice's we recommend Vasotec";
                        break;
                    case 2:
                        medical_message = "Based on your choice's we recommend Monopril";
                        break;
                    case 3:
                        medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                        break;
                }
                break;
            case 3:
                switch (discomfort_level)
                {
                    case 1:
                        medical_message = "Based on your choice's we recommend Lotensin";
                        break;
                    case 2:
                        medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                        break;
                    case 3:
                        medical_message = "Based on your choice's we recommend Visiting a doctor for further diagnosis";
                        break;
                }
                break;
        }
    }
    break;

```

Figure 29. patient_upper_body (factory)

The patient upper body class deals with the switch case selections with the head area of pain.

It checks for gender, discomfort level, age group, pain's amount.

Depending on the choice, recommendation of either medicine or professional guidance is given.

- Pain_Area

```
public class Pain_Area
{
    // Basic Patient Information
    int patient_age_group;
    char patient_gender;

    // Patient Pain Information
    int patient_pain_area;
    int patient_pain_time;
    int patient_pain_level;

    // Temp Medical Message Store
    public String medicalMessage;

    public Pain_Area(int patient_age_group, char patient_gender, int patient_pain_area)
    {
        this.patient_age_group = patient_age_group;
        this.patient_gender = patient_gender;
        this.patient_pain_area = patient_pain_area;
    }

    // Question 4 //////////////////////////////////////
    public void getUserPainTime()
    {
        if (mainpage.jRadioButton9.isSelected() == true) {
            patient_pain_time = 1;

        } else if (mainpage.jRadioButton10.isSelected() == true) {
            patient_pain_time = 2;

        } else if (mainpage.jRadioButton11.isSelected() == true) {
            patient_pain_time = 3;

        }

    }
}
```

```

// Question 5 //////////////////////////////////
public void getUserPainLevel()
{
    if (mainpage.jRadioButton12.isSelected() == true) {
        patient_pain_level = 1;

    } else if (mainpage.jRadioButton13.isSelected() == true) {
        patient_pain_level = 2;

    } else if (mainpage.jRadioButton14.isSelected() == true) {
        patient_pain_level = 3;

    }

}

public Patient getPatientMedication(int age_group, char gender)
{
    if (patient_pain_area == 1)
    {
        Patient_Head obj = new Patient_Head(patient_age_group, patient_gender, patient_pain_time, patient_pain_level);
        medicalMessage = obj.getMedical_message();
        return obj;
    }
    else if (patient_pain_area == 2)
    {
        Patient_Upper_Body obj = new Patient_Upper_Body(patient_age_group, patient_gender, patient_pain_time, patient_pain_level);
        medicalMessage = obj.getMedical_message();
        return obj;
    }
    else if (patient_pain_area == 3)
    {
        Patient_Lower_Body obj = new Patient_Lower_Body(patient_age_group, patient_gender, patient_pain_time, patient_pain_level);
        medicalMessage = obj.getMedical_message();
        return obj;
    }
    else
    {
        return null;
    }
}
}

```

Figure 30. Pain_Area (Factory)

This class is responsible for getting the input from the user and creating objects based on the results of the data that is pulled from the user.

It will then relay the information to one of the three relevant classes (either head, patient lower body, or patient upper body) after creating the objects and will then get the relevant replies to display on the GUI side.

- mainpage (factory)

sign History | 

```
// ===== F A C T O R Y   P A T T E R N =====
// =====
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

    // Get the Age from the user
    int patient_age_group = 0;
    int patient_pain_area = 0;
    char patient_gender;

    // Temp String to hold gender String
    String tempGender = null;

    // Question 1 //////////////////////////////////////
    if (jRadioButton1.isSelected() == true) {
        patient_age_group = 1;

    } else if (jRadioButton2.isSelected() == true) {
        patient_age_group = 2;

    } else if (jRadioButton3.isSelected() == true) {
        patient_age_group = 3;
    }

    // Question 2 //////////////////////////////////////

    if (jRadioButton4.isSelected() == true) {
        tempGender = "F";

    } else if (jRadioButton5.isSelected() == true) {
        tempGender = "M";

    }

    //

    if (tempGender.equals("M")) {
        patient_gender = 'M';
    }
    else {
        patient_gender = 'F';
    }
}
```

```

// Question 3 //////////////////////////////////////

if (jRadioButton6.isSelected() == true) {
    patient_pain_area = 1;

} else if (jRadioButton7.isSelected() == true) {
    patient_pain_area = 2;

} else if (jRadioButton8.isSelected() == true) {
    patient_pain_area = 3;
}

// Assign the Variables to the constructor of the Pain Area Class

// Create Patient Object
Pain_Area patient = new Pain_Area(patient_age_group, patient_gender, patient_pain_area);

// Call the Pain_Area class Method for getting the pain level
patient.getUserPainLevel();

// Call the Pain_area class Method for getting the pain time
patient.getUserPainTime();

// Generate the medication message
patient.getPatientMedication(patient_age_group, patient_gender);

// Display the medical recommendation
String tempData;
tempData = patient.medicalMessage;
jLabel10.setText(tempData);

}

```

Figure 31. mainpage (factory)

This section on the GUI takes the necessary information and relays it to the patient area class for further use for the factory pattern.

- Factory Pattern output

The screenshot shows a web application window with a title bar and standard window controls. The main content area has a header 'Welcome, Avishka' and a navigation bar with tabs: 'Medicine Recommendation' (active), 'Ailment Info', 'Medicine Purchases', 'Application Users List', and 'About'. Below the navigation bar, the title 'Medication or Remedy Recommendation' is displayed in green. The form is divided into two main sections. The top section contains three columns of radio button options: 'Select your age group' (Age 18-25, Age 26-35 (selected), Age 35 and Up), 'Select your gender' (F, M (selected)), and 'Select The Location your pain area:' (Head, Upper Body (Chest Area) (selected), Lower Body (Abdomen and below)). The bottom section contains two columns: 'Select the time you were experiencing pain:' (Less than 2 hours, 2 - 6 hours (selected), More than 6 hours) and 'Please Select The Level of pain you are facing:' (Mild Pain, Moderate Pain (selected), Severe Pain). Below these sections, a message states 'Based on your choice's we recommend Lisinopril'. At the bottom, there are two buttons: 'Enter' (highlighted with a blue dashed border) and 'Reset'.

Welcome, Avishka [< Logout](#)

Medicine Recommendation Ailment Info Medicine Purchases Application Users List About

Medication or Remedy Recommendation

Select your age group:

☐ Age 18-25

☒ Age 26-35

☐ Age 35 and Up

Select your gender:

☐ F

☒ M

Select The Location your pain area:

☐ Head

☒ Upper Body (Chest Area)

☐ Lower Body (Abdomen and below)

Select the time you were experiencing pain:

☐ Less than 2 hours

☒ 2 - 6 hours

☐ More than 6 hours

Please Select The Level of pain you are facing:

☐ Mild Pain

☒ Moderate Pain

☐ Severe Pain

Based on your choice's we recommend Lisinopril

Figure 32. Factory pattern output 1

Content

Admin

Medicine

Links

Comments

Header & Footer

Welcome, Avishka

< Logout

Medicine RecommendationAilment InfoMedicine PurchasesApplication Users ListAbout

Medication or Remedy Recommendation

Select your age group:

☐ Age 18-25

☐ Age 26-35

☒ Age 35 and Up

Select your gender:

☐ F

☒ M

Select The Location your pain area:

☐ Head

☐ Upper Body (Chest Area)

☒ Lower Body (Abdomen and below)

Select the time you were experiencing pain:

☒ Less than 2 hours

☐ 2 - 6 hours

☐ More than 6 hours

Please Select The Level of pain you are facing:

☒ Mild Pain

☐ Moderate Pain

☐ Severe Pain

Based on your choice's we recommend Quinapril

Enter

Reset

Figure 33. Factory pattern output 2

The rest of the selections with the radio buttons takes the same format

Application Users List (Proxy)

- ProxyExecutor Interface

```
package EL_Proxy;

public interface ProxyExecutor {

    // used to restrict access per user depending on admin privileges
    public void testexecute(String test) throws Exception;

}
```

Figure 34. ProxyExecutor Interface (Proxy)

This interface will be used on the rest of the factory pattern by overriding it in the subclasses when necessary.

- Execute

```
import DesignP_Assignment_Common.mainpage;

public class Execute implements ProxyExecutor{

    @Override
    public void testexecute(String test1) throws Exception {
        System.out.println("The follwing feature will be available due to account having admin priviledges: " + test1);

        mainpage.jPanel6.setVisible(true);
        mainpage.jPanel7.setVisible(false);
    }
}
```

Figure 35. Execute (proxy)

This section implements the interface and overrides its method. This class holds the code if the proxy pattern if the logged in user is an admin.

It will display a jpanel consisting with the users list for the admin account.

And will hide the other panel displaying a message that the user is not an admin.

- ExecuteProxy

```

import DesignP_Assignment_Common.mainpage;

public class ExecuteProxy implements ProxyExecutor{

    boolean isAdmin;
    Execute execute;

    // issue should be here
    public ExecuteProxy(String username, String password) {

        if (username == "admin" && password == "admin123") {
            isAdmin = true;
        }
        execute = new Execute();
    }

    @Override
    public void testexecute(String test1) throws Exception {
        if(isAdmin) {
            execute.testexecute(test1);
        } else {
            if(test1.equals("AdminDetails")) {
                System.out.println("Certain access is blocked due to account not having administrator priviledges!");
                mainpage.jPanel6.setVisible(false);
                mainpage.jPanel7.setVisible(true);
            } else {
                execute.testexecute(test1);
            }
        }
    }
}

```

Figure 36. ExecuteProxy (factory)

The execute proxy class will check if the entered user is an admin, and if they are, will set the Boolean variable to true.

Then in testexecute method if the Boolean is true, it will run the Execute because the user is an admin and will display the necessary information.

If not, the user will be displayed a message that the user is not an admin in the GUI.

And the registered users list will be hidden from viewing.

- Login (proxy)

```
// ===== P R O X Y   P A T T E R N =====  
// =====  
  
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
  
    String getUsername = jTextField1.getText();  
    String getPassword = jPasswordField1.getText();  
  
    // Validation to check if username or password field is empty  
    if((jTextField1.getText().equals("") || jPasswordField1.getText().equals(""))){  
        JOptionPane.showMessageDialog(this, "The Username or Password field cannot be empty! Please try again.");  
    } else {  
        dispose();  
        vali.getValidation(jTextField1.getText(),jPasswordField1.getText());  
  
        // checks username entered in jtext and jpassword fields in the login page  
        if (getUsername.equals("admin") && getPassword.equals("admin123")) {  
            try {  
                isAdminproxy();  
            } catch (Exception ex) {  
                System.out.println("an error has occurred with Admin proxy");  
            }  
        } else {  
            try {  
                isNOTAdminproxy();  
            } catch (Exception ex) {  
                System.out.println("an error has occurred with NOT Admin proxy");  
            }  
        }  
    }  
}
```

Figure 37. Login (proxy)

In this section, if the relevant admin logins are entered into a text field, the isAdminproxy method will be run to display the registered users list table.

If the entered logins are not of an admin, it will run the isNOTAdminproxy to hide access from the list because the necessary admin privileges are missing.


```
// PROXY PATTERN METHODS
public void isAdminproxy() throws Exception {

    ProxyExecutor isAdmin = new ExecuteProxy("admin", "admin123");
    isAdmin.testexecute("AdminDetails");

}

public void isNOTAdminproxy() throws Exception {

    ProxyExecutor nonAdmin = new ExecuteProxy("notadmin", "notadmin");
    nonAdmin.testexecute("AdminDetails");

}
```

Figure 38. isAdminProxy and isNOTAdminProxy

```
// To display the users list in the users tab in a jTable (Only accessible by admin accounts through Proxy pattern)
public void table() {

    try {

        connection = ConnectionManager.getConnection();

        Statement st = connection.createStatement();
        String sql = "select * from registrations";
        ResultSet rs = st.executeQuery(sql);

        while (rs.next()) {

            String fname = rs.getString("Fname");
            String username = rs.getString("username");
            String email = rs.getString("email");

            String tabledata[] = {fname, username, email};
            DefaultTableModel tbmodel = (DefaultTableModel)jTable2.getModel();

            tbmodel.addRow(tabledata);

        }

    }catch (SQLException ex) {

        System.out.println("Error in Admin section db pull for table");

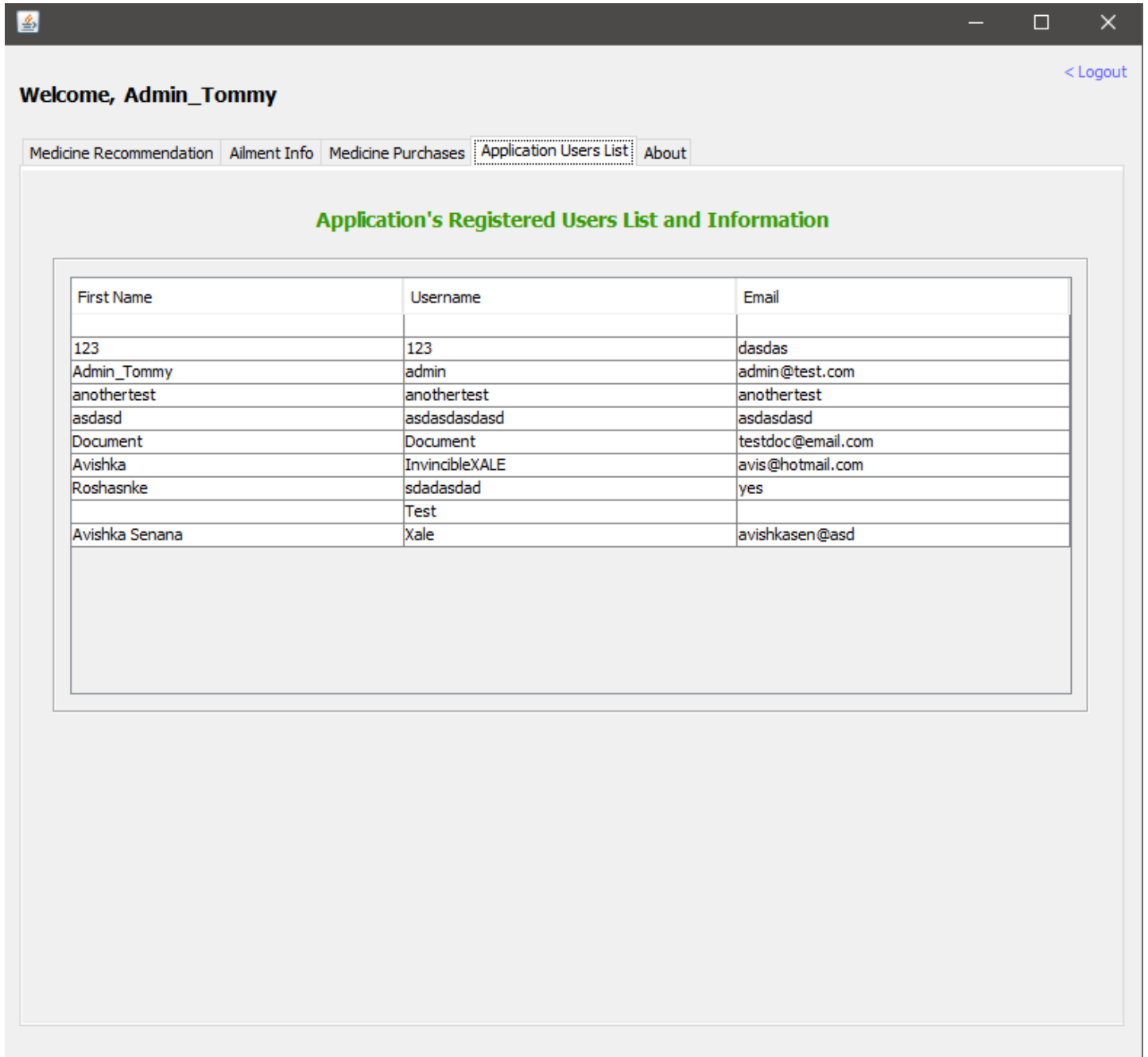
    }

}
```

Figure 39. Table method for Proxy

- Proxy Output

If user is an admin:



The screenshot shows a web application window with a dark title bar. The main content area has a light gray background. At the top left, it says "Welcome, Admin_Tommy". At the top right, there is a "< Logout" link. Below the welcome message, there is a navigation bar with five tabs: "Medicine Recommendation", "Ailment Info", "Medicine Purchases", "Application Users List" (which is highlighted with a dotted border), and "About". Below the navigation bar, there is a section titled "Application's Registered Users List and Information" in green text. This section contains a table with three columns: "First Name", "Username", and "Email". The table has 10 rows of data. Below the table, there is a large empty rectangular box.

First Name	Username	Email
123	123	dasdas
Admin_Tommy	admin	admin@test.com
another test	another test	another test
asdasd	asdasdasdasd	asdasdasd
Document	Document	testdoc@email.com
Avishka	InvincibleXALE	avis@hotmail.com
Roshasnke	sdadasdad	yes
	Test	
Avishka Senana	Xale	avishkasen@asd

Figure 40. If user is an admin proxy

If user is NOT an admin:

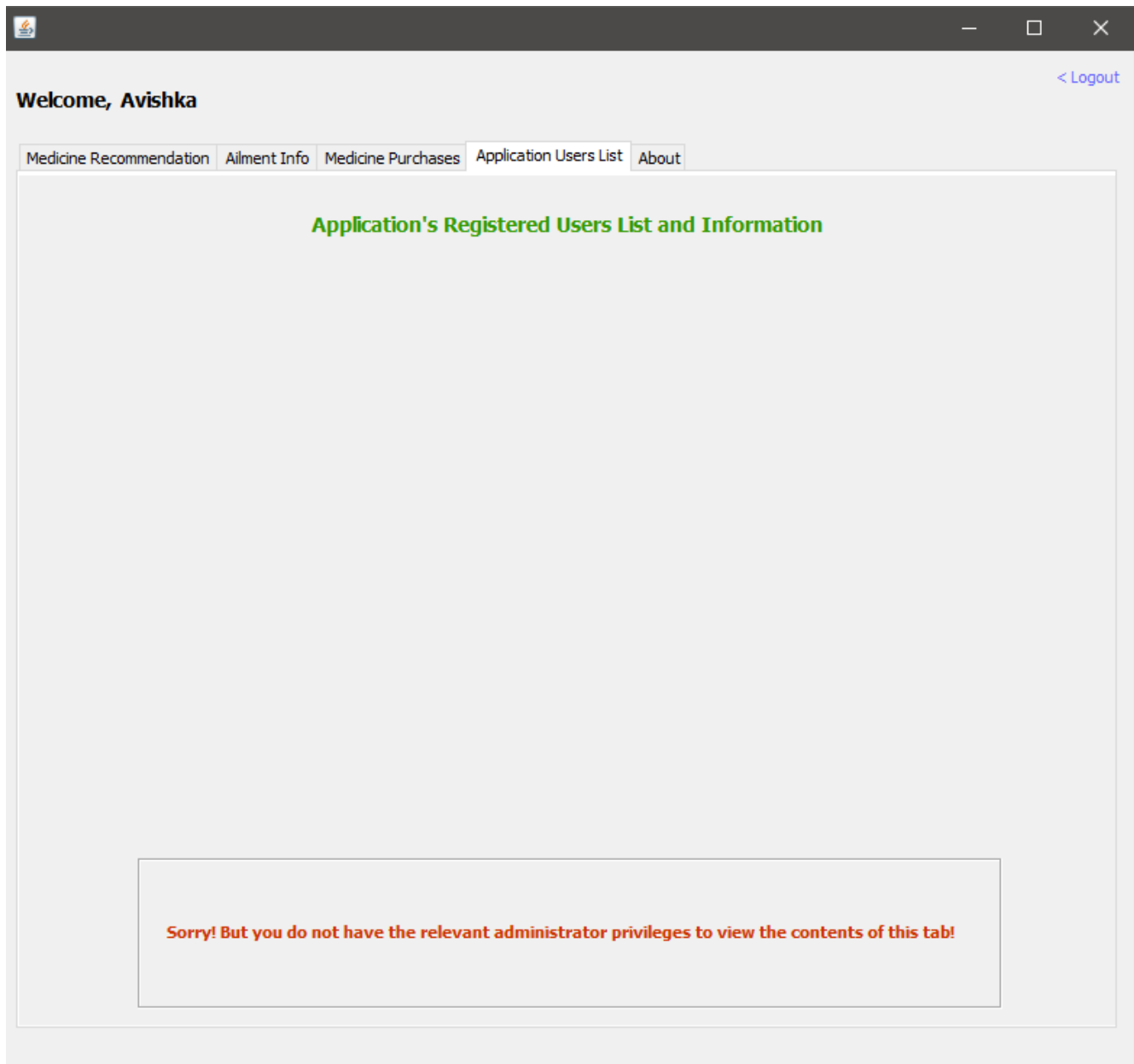


Figure 41. If user is NOT an admin Proxy

Medicine purchases (Decorator)

- Meds

```
public interface Meds {  
  
    void fetchmeds();  
  
    // Used to get the price of each medicine and adding tax based on the milligram choice  
    int getCost();  
  
    // Used to get the needed information and pull it to a JLabel  
    String desc();  
  
}
```

Figure 42. Meds (Decorator)

The meds interface will be creating three methods to get the description and cost of individual items for the decorator class by overriding them when necessary.

- Abenol

```
public class Abenol implements Meds {  
  
    @Override  
    public void fetchmeds() {  
        //System.out.println("Box of Abenol with the Milligram of ");  
    }  
  
    // base cost of Abenol  
    @Override  
    public int getCost(){  
        return 20;  
    }  
  
    // new stuff  
    @Override  
    public String desc(){  
        return "Box of Abenol with the Milligram of ";  
    }  
  
}
```

Figure 43. Abenol (Decorator)

The Abenol class implements meds and sets a default value for the medication of Abenol. It returns a message and has a base cost of \$20.

- Aspirin

```
public class Aspirin implements Meds{

    @Override
    public void fetchmeds() {
        // System.out.print("Box of Aspirin with the milligram of ");
    }

    // Base cost of Aspirin
    @Override
    public int getCost(){
        return 10;
    }

    @Override
    public String desc(){
        return "Box of Aspirin with the milligram of ";
    }

}
```

Figure 44. Aspirin (Decorator)

The Aspirin class implements meds and sets a default value for the medication of Aspirin.

It returns a message and has a base cost of \$10.

- Ibuprofen

```
public class Ibuprofen implements Meds {  
  
    @Override  
    public void fetchmeds() {  
        //System.out.println("Box of Ibuprofen with the Miligram of ");  
    }  
  
    //Base cost of Ibuprofen  
    @Override  
    public int getCost() {  
        return 30;  
    }  
  
    // new stuff  
    @Override  
    public String desc(){  
        return "Box of Ibuprofen with the Miligram of ";  
    }  
}
```

Figure 45. Ibuprofen (Decorator)

The Ibuprofen class implements meds and sets a default value for the medication of Ibuprofen.

It returns a message and has a base cost of \$30.

- MedDecorator

```
abstract class MedDecorator implements Meds {  
  
    protected Meds decoratedMeds;  
  
    public MedDecorator(Meds decoratedMeds) {  
        this.decoratedMeds = decoratedMeds;  
    }  
  
    @Override  
    public void fetchmeds() {  
        decoratedMeds.fetchmeds();  
    }  
  
    @Override  
    public int getCost() {  
        return decoratedMeds.getCost();  
    }  
  
    @Override  
    public String desc() {  
        return decoratedMeds.desc();  
    }  
  
}
```

Figure 46. MedDecorator (Decorator)

An abstract class of MedDecorator is created to pull the above medication's description and cost when needed for the rest of the decorator pattern.

- 250mg

--

```
public class _250mg extends MedDecorator {

    public _250mg(Meds decoratedMeds) {
        super(decoratedMeds);
    }

    @Override
    public void fetchmeds() {

        super.fetchmeds();
        set250mg();
        getCost();
    }

    private void set250mg() { // Meds decoratedMeds in parameters
        // new thing
        // System.out.print("250 per tablet.");
    }

    // Tax for 250 Milligram meds
    @Override
    public int getCost() {
        return super.getCost() + 7;
    }

    @Override
    public String desc() {
        return super.desc() + "250 per tablet.";
    }

}
```

Figure 47. _250mg (decorator)

The _250mg class is extending MedDecorator and handles the tax cost (\$7 tax) and sub description of the medicine per 250 milligrams.

- _500mg

```

public class _500mg extends MedDecorator {

    public _500mg(Meds decoratedMeds) {
        super(decoratedMeds);
    }

    @Override
    public void fetchmeds() {

        super.fetchmeds();
        set500mg();
        getCost();
    }

    private void set500mg(){ // Meds decoratedMeds in parameters
        // new thing
        // System.out.println("500 per tablet.");
    }

    // Tax for 500 Milligram meds
    @Override
    public int getCost(){
        return super.getCost() + 10;
    }

    @Override
    public String desc(){
        return super.desc() + "500 per tablet.";
    }

}

```

Figure 48. _500mg (decorator)

The _500mg class is extending MedDecorator and handles the tax cost (\$10 tax) and sub description of the medicine per 500 milligrams.

- _5mg

```
public class _5mg extends MedDecorator {

    public _5mg(Meds decoratedMeds) {
        super(decoratedMeds);
    }

    @Override
    public void fetchmeds() {

        super.fetchmeds();
        set5mg();
        getCost();
    }

    private void set5mg(){ // Meds decoratedMeds in parameters
        // new thing
        // System.out.println("5 per tablet.");
    }

    // Tax for 5 Milligram meds
    @Override
    public int getCost(){
        return super.getCost() + 5;
    }

    @Override
    public String desc(){
        return super.desc() + "5 per tablet.";
    }

}
```

Figure 49. _5mg (Decorator)

The _5mg class is extending MedDecorator and handles the tax cost (\$5 tax) and sub description of the medicine per 5 milligrams.

- mainpage (decorator)

```
// ===== D E C O R A T O R   P A T T E R N =====
// =====

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

    // for first combo box
    String med = (String)jComboBox2.getSelectedItem();

    // for second combo box
    String mg = (String)jComboBox3.getSelectedItem();

    if (med == "Abenol" && mg == "5 Milligrams" ) {
        Meds med1 = new _5mg(new Abenol());
        jLabel115.setText(med1.desc());
        jLabel116.setText("Price of Individual product is: $" + Integer.toString(med1.getCost()));

        int finalcost = med1.getCost() * Integer.parseInt(jTextField1.getText());
        jLabel117.setText("The final amount is: $" + Integer.toString(finalcost));

        jLabel122.setText(info.Abeno());
        jLabel123.setVisible(true);
    } else if (med == "Abenol" && mg == "250 Milligrams" ) {
        Meds med2 = new _250mg(new Abenol());
        jLabel115.setText(med2.desc());
        jLabel116.setText("Price of Individual product is: $" + Integer.toString(med2.getCost()));

        int finalcost = med2.getCost() * Integer.parseInt(jTextField1.getText());
        jLabel117.setText("The final amount is: $" + Integer.toString(finalcost));

        jLabel122.setText(info.Abeno());
        jLabel123.setVisible(true);
    } else if (med == "Abenol" && mg == "500 Milligrams" ) {
        Meds med3 = new _500mg(new Abenol());
        jLabel115.setText(med3.desc());
        jLabel116.setText("Price of Individual product is: $" + Integer.toString(med3.getCost()));

        int finalcost = med3.getCost() * Integer.parseInt(jTextField1.getText());
        jLabel117.setText("The final amount is: $" + Integer.toString(finalcost));

        jLabel122.setText(info.Abeno());
        jLabel123.setVisible(true);
    }
}
```

```

} else if (med == "Aspirin" && mg == "5 Milligrams" ) {
    Meds med4 = new _5mg(new Aspirin());
    jLabel115.setText(med4.desc());
    jLabel116.setText("Price of Individual product is: $" + Integer.toString(med4.getCost()));

    int finalcost = med4.getCost() * Integer.parseInt(jTextField1.getText());
    jLabel117.setText("The final amount is: $" + Integer.toString(finalcost));

    jLabel122.setText(info.Aspr());
    jLabel123.setVisible(true);

} else if (med == "Aspirin" && mg == "250 Milligrams" ) {
    Meds med5 = new _250mg(new Aspirin());
    jLabel115.setText(med5.desc());
    jLabel116.setText("Price of Individual product is: $" + Integer.toString(med5.getCost()));

    int finalcost = med5.getCost() * Integer.parseInt(jTextField1.getText());
    jLabel117.setText("The final amount is: $" + Integer.toString(finalcost));

    jLabel122.setText(info.Aspr());
    jLabel123.setVisible(true);

} else if (med == "Aspirin" && mg == "500 Milligrams" ) {
    Meds med6 = new _500mg(new Aspirin());
    jLabel115.setText(med6.desc());
    jLabel116.setText("Price of Individual product is: $" + Integer.toString(med6.getCost()));

    int finalcost = med6.getCost() * Integer.parseInt(jTextField1.getText());
    jLabel117.setText("The final amount is: $" + Integer.toString(finalcost));

    jLabel122.setText(info.Aspr());
    jLabel123.setVisible(true);

} else if (med == "Ibuprofen" && mg == "5 Milligrams") {
    Meds med7 = new _5mg(new Ibuprofen());
    jLabel115.setText(med7.desc());
    jLabel116.setText("Price of Individual product is: $" + Integer.toString(med7.getCost()));

    int finalcost = med7.getCost() * Integer.parseInt(jTextField1.getText());
    jLabel117.setText("The final amount is: $" + Integer.toString(finalcost));

    jLabel122.setText(info.Ibupro());
    jLabel123.setVisible(true);

```

```

} else if (med == "Ibuprofen" && mg == "250 Milligrams") {
    Meds med8 = new _250mg(new Ibuprofen());
    jLabel15.setText(med8.desc());
    jLabel16.setText("Price of Individual product is: $" + Integer.toString(med8.getCost()));

    int finalcost = med8.getCost() * Integer.parseInt(jTextField1.getText());
    jLabel17.setText("The final amount is: $" + Integer.toString(finalcost));

    jLabel22.setText(info.Ibupro());
    jLabel23.setVisible(true);

} else if (med == "Ibuprofen" && mg == "500 Milligrams") {
    Meds med9 = new _500mg(new Ibuprofen());
    jLabel15.setText(med9.desc());
    jLabel16.setText("Price of Individual product is: $" + Integer.toString(med9.getCost()));

    int finalcost = med9.getCost() * Integer.parseInt(jTextField1.getText());
    jLabel17.setText("The final amount is: $" + Integer.toString(finalcost));

    jLabel22.setText(info.Ibupro());
    jLabel23.setVisible(true);

} else if (med == "None Selected" || mg == "None Selected") {
    JOptionPane.showMessageDialog(this, "Please select the medicine, the milligram amount you require "
        + "from the dropdown menus and enter the amount of boxes you require!");

} else {
    JOptionPane.showMessageDialog(this, "An error has occurred!");
}
}

```

Figure 50. mainpage (Decorat

In this section, the application checks the two selections from the drop boxes (medicine and milligram amount respectively) that the user has selected. Then, depending on the selection they have made, in a set of if else statements will display the necessary information after a button click is made.

For example,

If the selects Abenol of 5mg, an object of the Meds interface called 'med1' is created that is linked with _5mg class. It is then wrapped in with the information of the Abenol class. Making it a perfect use of the decorator pattern.

Afterwards, the description of the selection will be displayed on a JLabel with med1.desc() to get the description of "Box of Abenol with the Milligram of 5 per tablet".

Then with the med1.getCost() method, the base price of the medicine with the tax will be added and displayed.

The user will also be able to select the amount they want by a JTextField. The entered value will be retrieved and multiplied with the getCost() method to display the total and full amount.

Information on the Abenol medicine will also be displayed by pulling a method from a class that is NOT related to the decorator pattern and will display a thank you image.

- Decorator Output

Welcome, Avishka [< Logout](#)

Medicine Recommendation Ailment Info **Medicine Purchases** Application Users List About

Medicine purchase and pricing scheme

Select Medicine you want:

Select Milligram Amount:

Enter the quantity you want:

Get Medicine Amount Reset

Box of Abenol with the Milligram of 5 per tablet.

Price of Individual product is: \$25

The final amount is: \$1250

Abenol is a pain relever called 'analgesics' for minor discomforts. It takes about 1-2 hours to take action after consumption.





Figure 51. Decorator Output 1



Welcome, Avishka

< Logout

Medicine RecommendationAilment InfoMedicine PurchasesApplication Users ListAbout

Medicine purchase and pricing scheme

Select Medicine you want:

Aspirin

Select Milligram Amount:

250 Milligrams

Enter the quantity you want:

42

Get Medicine Amount

Reset

Box of Aspirin with the milligram of 250 per tablet.

Price of Individual product is: \$17

The final amount is: \$714

The Aspirin medication is used to reduce or momentarily eliminate the sense of pain such as headaches or fever. It can be used for many mild to moderate pain symptoms as is very common in the day to day household. Recommended dosage per day is 1 tablet for children and 2 for adults.




Figure 52. Decorator Output 2

The rest of the selections with the drop-down buttons and quantity entered in the `(jTextField)` take the same format as the examples above

- 97 - | Page

Use Case Diagrams

Application users list use case

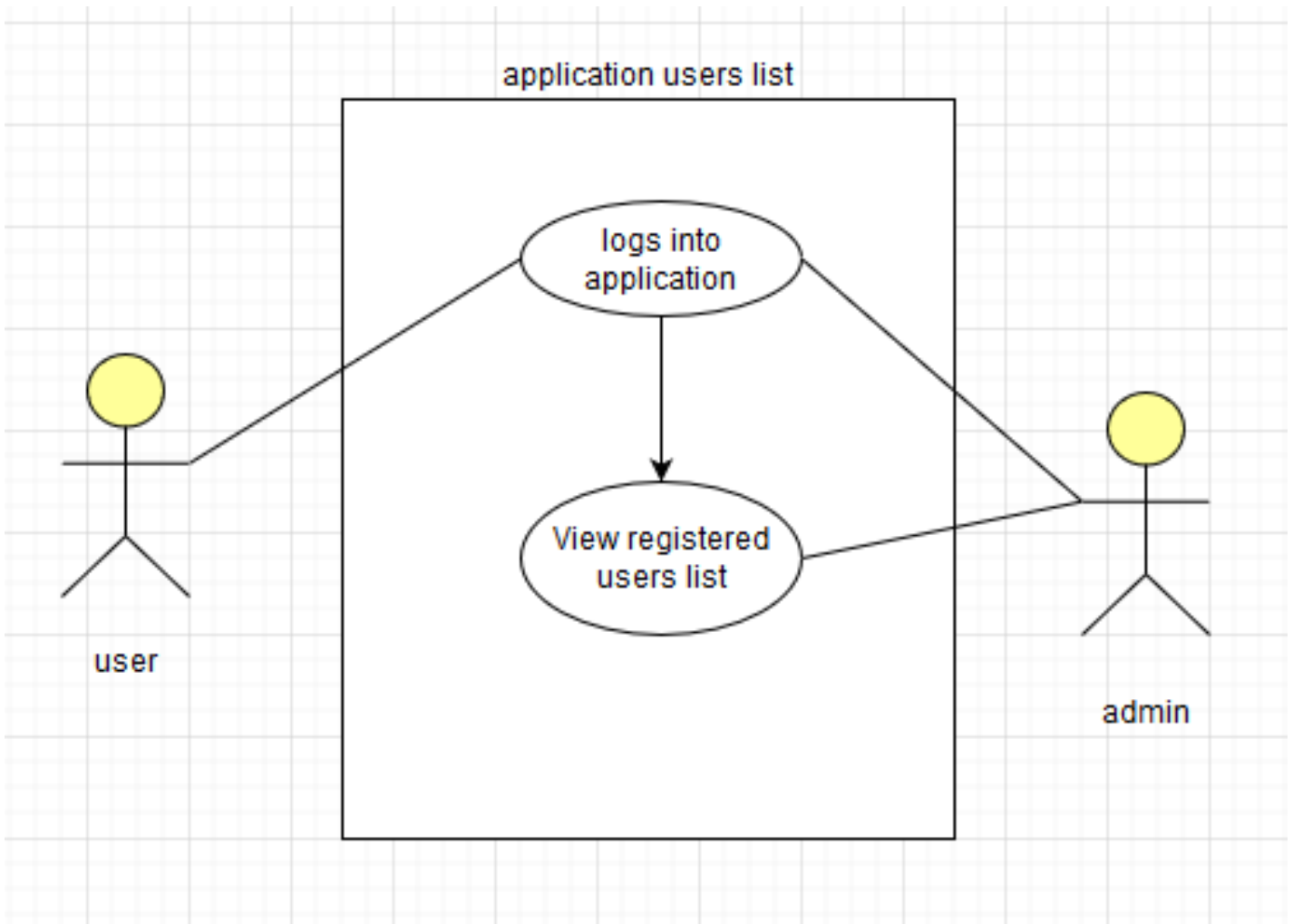


Figure 53. Application users list use case

Login use case

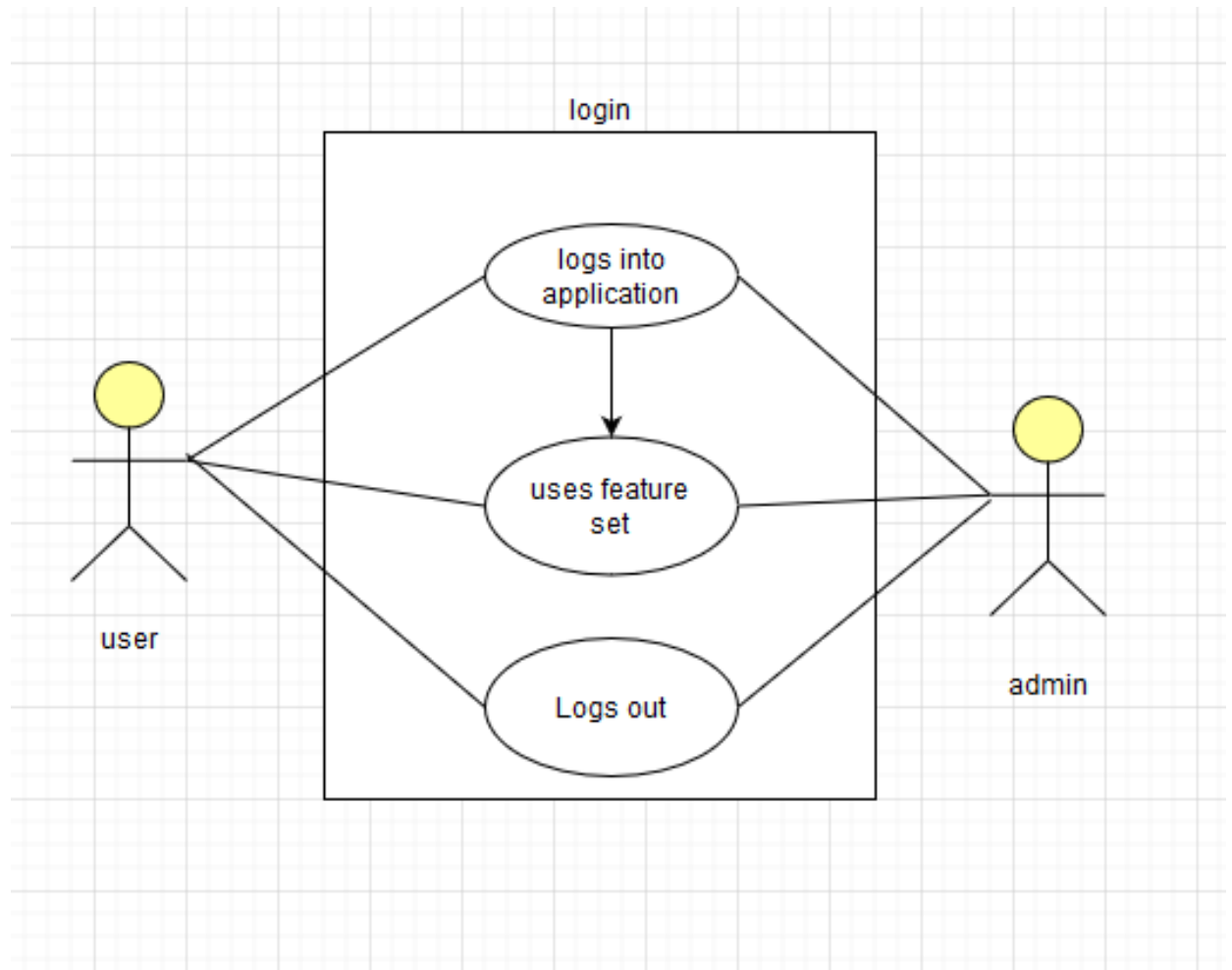


Figure 54. Login use case

Medicine purchases use case

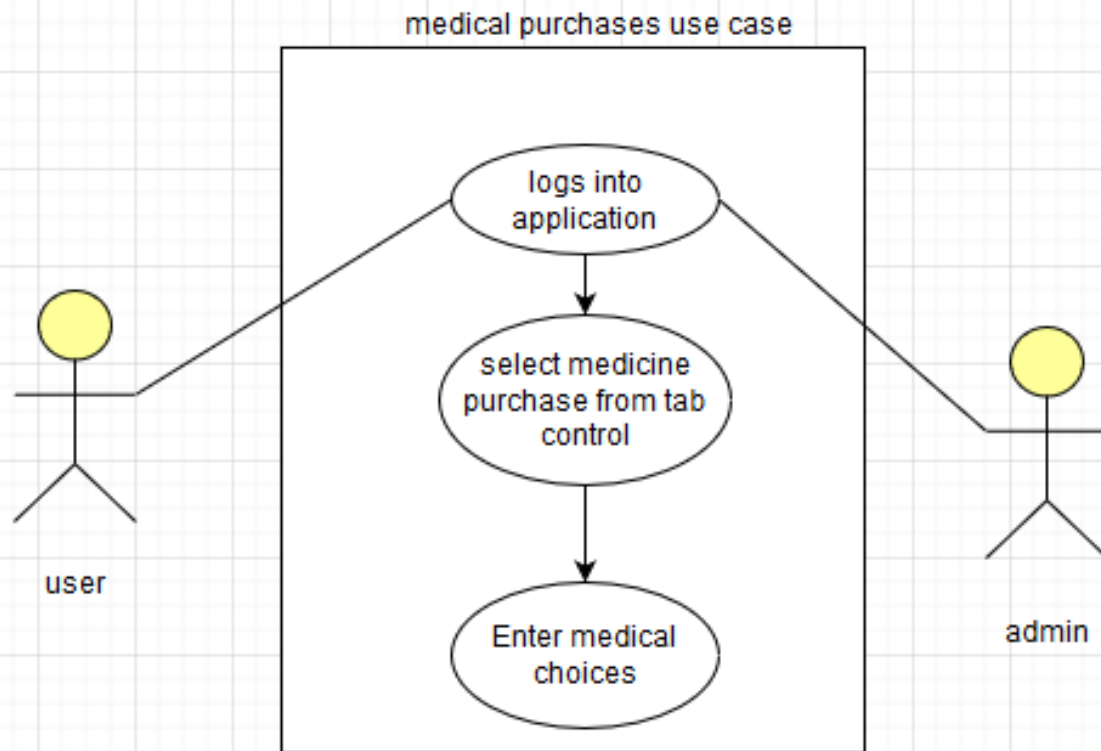


Figure 55. Medicine purchases use case

Ailment info use case

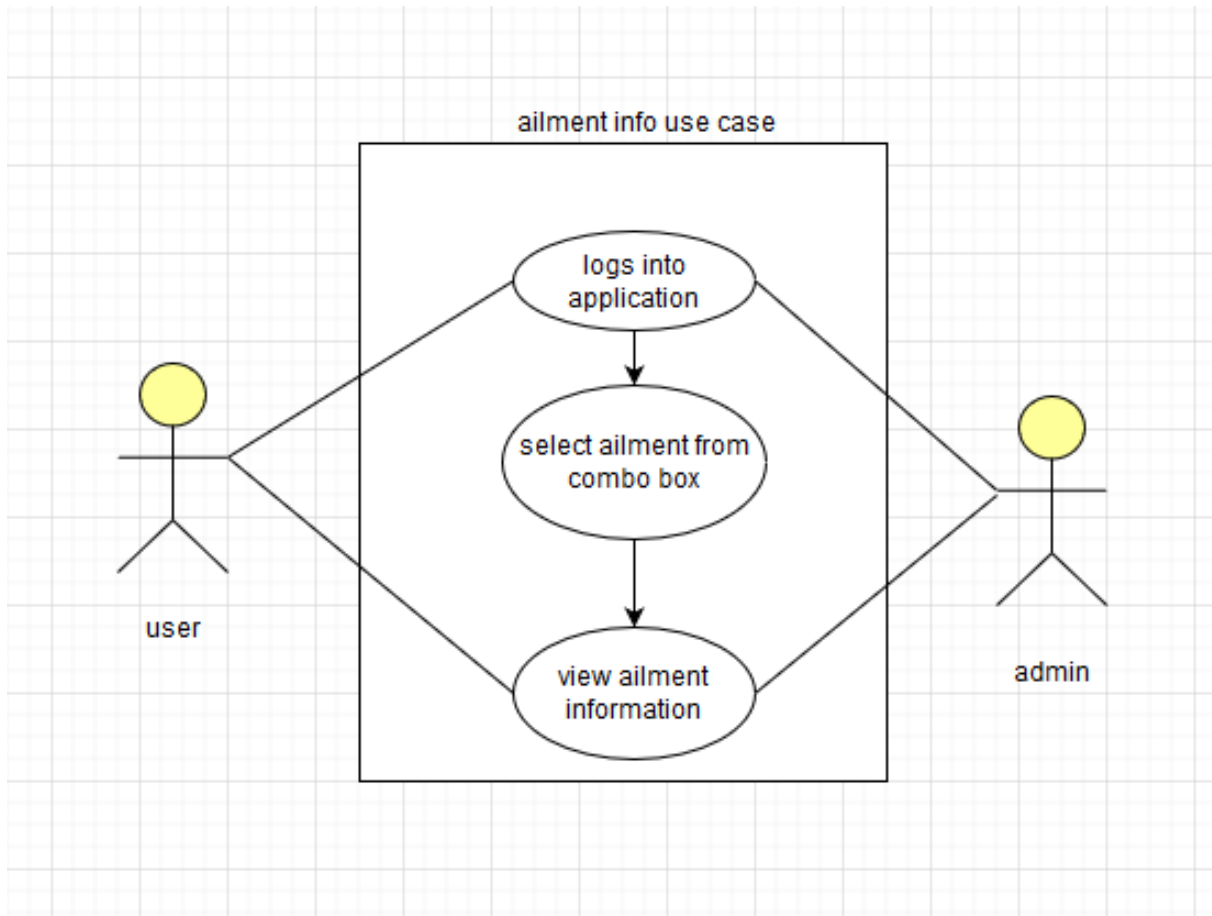


Figure 56. Ailment info use case

Test cases

Test cases:

No.	Description	Expected Outcome	Actual Outcome	Pass / Fail
TSC_01	When pressing Sign up, the login screen will close and go to the sign-up screen.	Login page closes and sign-up page opens	Login page closes and sign-up page opens	Pass
TSC_02	When pressing Go back in the sign-up screen, the sign-up screen will close and go to the login screen.	Sign up page closes and the login page open	Sign up page closes and the login page open	Pass
TSC_03	When pressing login with empty text fields, and error message pops up saying the fields cannot be empty.	A message pops up saying fields cannot be empty	A message pops up saying fields cannot be empty	Pass
TSC_04	After entering the relevant information and pressing register in the sign-up page, the data will be uploaded to the database.	Data is uploaded to the database	Data is uploaded to the database	Pass
TSC_05	Entering incorrect details in the logins section will display an error message	Application should show an error message	Application displays an error but does not continue from there.	Fail
TSC_06	Entering correct details in the logins section will move onto the main page.	The Main page will be displayed	The Main page is Displayed	Pass
TSC_07	Entering unmatching password in confirm password field in the sign-up page, will highlight the confirm password text field's border to highlight a discrepancy.	Confirm password border will get highlighted in red	Confirm password border gets highlighted in red	Pass

TSC_08	Logging in with a non admin account will hide the user list feature in the application's main page.	User list feature will not be displayed	User list feature is not displayed	Pass
TSC_09	Logging in with a admin account will show the user list feature in the application's main page.	User list feature will be displayed	User list feature is displayed	Pass
TSC_11	Logging in with an account with display the username on the top of the main page	Username will be displayed at the top of the main page.	Username is displayed at the top of the main page.	Pass
TSC_12	Making selections in the medicine recommendation will then display an answer when pressed reply for the factory pattern.	Medicine recommendation is given	Medicine recommendation is given	Pass
TSC_13	Pressing reset in the medicine recommendation will reset all the choices from the button groups.	Button group choices reset	Button group choices reset	Pass
TSC_14	Selecting Hypertension or high blood pressure in the ailment info's drop box will display	Hypertension info will be displayed	Hypertension info displayed	Pass
TSC_15	Selecting Blood Sugar in the ailment info's drop box will display	Blood Sugar info will be displayed.	Blood Sugar info displayed.	Pass
TSC_16	Selecting Mental health in the ailment info's drop box will display	Mental health info will be displayed.	Mental health info displayed.	Pass
TSC_17	Selecting Migraines in the ailment info's drop box will display	Migraines info will be displayed.	Migraines info displayed.	Pass
TSC_18	Selecting Obesity or in the ailment info's drop box will display	Obesity info will be displayed.	Obesity info displayed.	Pass

TSC_20	Selecting Gastritis or in the ailment info's drop box will display.	Gastritis info will be displayed.	Gastritis info displayed.	Pass
TSC_21	Pressing log out in the main page will exit the main page and display the login page.	Login page will be displayed	Login page is displayed	Pass
TSC_22	Pressing get medicine amount in medicine purchases tab while nothing is selected will bring up a pop up displaying an error.	An error pop up will be displayed	Error pop up is displayed	Pass
TSC_24	Selecting Abenol and 5mg with an amount will display data when get medicine amount is pressed.	Abenol and 5mg information is displayed	Abenol and 5mg information is displayed	Pass
TSC_25	Selecting Abenol and 250mg with an amount will display data when get medicine amount is pressed.	Abenol and 250mg information is displayed	Abenol and 250mg information is displayed	Pass
TSC_26	Selecting Abenol and 500mg with an amount will display data when get medicine amount is pressed.	Abenol and 500mg information is displayed	Abenol and 500mg information is displayed	Pass
TSC_27	Selecting Aspirin and 5mg with an amount will display data when get medicine amount is pressed.	Aspirin and 5mg information are displayed	Aspirin and 5mg information are displayed	Pass
TSC_28	Selecting Aspirin and 250mg with an amount will display data when get medicine amount is pressed.	Aspirin and 250mg information are displayed	Aspirin and 250mg information are displayed	Pass
TSC_29	Selecting Aspirin and 500mg with an amount will display data when get medicine amount is pressed.	Aspirin and 500mg information are displayed	Aspirin and 500mg information are displayed	Pass

TSC_30	Selecting Ibuprofen and 5mg with an amount will display data when get medicine amount is pressed.	Ibuprofen and 5mg information are displayed	Ibuprofen and 5mg information are displayed	Pass
TSC_31	Selecting Ibuprofen and 250mg with an amount will display data when get medicine amount is pressed.	Ibuprofen and 250mg information are displayed	Ibuprofen and 250mg information are displayed	Pass
TSC_32	Selecting Ibuprofen and 500mg with an amount will display data when get medicine amount is pressed.	Ibuprofen and 500mg information are displayed	Ibuprofen and 500mg information are displayed	Pass
TSC_32	Pressing the reset button in the medicine purchases tab will reset everything back to its default.	Everything is reset to its default	Everything is reset to its default	Pass
TSC_33	Pressing the about tab in the main page will display the about information in the application.	About information is displayed	About information is displayed	Pass

Conclusion

The existence of the gang of four (GoF) design patterns makes a developer's lives much easier and more organized and precise for the specific usage of it in question, for a feature. Handling and creating Object oriented related applications and can span across multiple projects despite its scope and provides proper transparency to what needs to be seen. The purpose of design patterns is to give the programmer established methods to tackle problems that they may face in each project. (*Vogel, 2019*)

Thanks to this assignment and getting the opportunity to create a Java Swing based GUI, I now have a much higher knowledge with object-oriented programming with java and a much higher understanding in tandem with design patterns and how it makes handling various tasks much more versatile with 6 design patterns in use.

Extra Life Health application is a vastly robust application that fulfils its purpose to the end user of displaying the necessary information with the features. That is its best strength. Adding to this, a GUI having backend database handling for user accounts and feature blocking was also bonus.

The difficulty of this assignment was integrating the GUI to the design patterns that we were tasked to use with it. However, this has given me more notes to properly conduct an application like this in the future with design patterns from now on.

Bibliography

- Chovatiya, V., 2020. *What Is A Design Pattern? - Dzone Java*. [online] dzone.com. Available at: <<https://dzone.com/articles/what-is-design-pattern>> [Accessed 24 January 2021].
- Vogel, L., 2019. *Design Patterns In Java - Tutorial*. [online] Vogella.com. Available at: <<https://www.vogella.com/tutorials/DesignPatterns/article.html>> [Accessed 25 January 2021].