Swathi Selvakumaran

swathi99@umd.edu

TABLE OF CONTENTS

1 INSTALLATION AND SETUP

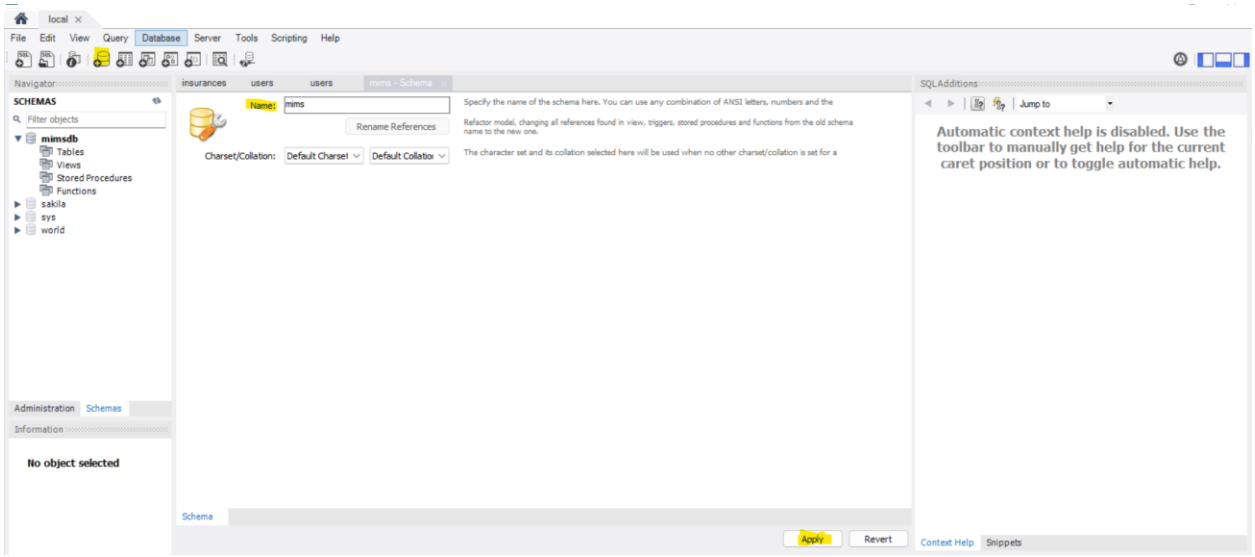
1. Pull the changes from the git repository or download the application zip file from git and extract it in your machine.
2. Install and configure the following dependencies.
    1. JDK and JAVA_HOME
        a. Download executable jdk 11 from the https://www.oracle.com/java/technologies/javase/jdk11-archive-downloads.html.
        b. Install jdk11 by executing the downloaded file in administrative mode and provide the privileges.
        c. Search for "Advanced system settings" in the windows search box and open the "View advanced system settings" present in the control panel.
        d. Select the Advanced tab and click Environment variables option present in the bottom right corner.
        e. Add new System variable "JAVA_HOME" with the path *<location of jdk installed>*. eg: C:\Program Files\Java\jdk-11.0.16.1
        f. Edit the System variable "Path" it will prompt you below dialog box, clicks on New button, and add this "%JAVA_HOME%\bin"
        g. Test the configuration: Open command prompt and type:
            - java -version
            - javac -version
            - echo %JAVA_HOME%

    2. Apache Maven
        a. Download the Maven zip file from https://maven.apache.org/download.cgi eg: apache-maven-3.8.6-bin.zip
        b. Unzip the file at C:\Program Files\ apache-maven-3.8.6 or any other convenient location.
        c. Search for "Advanced system settings" in the windows search box and open the "View advanced system settings" present in the control panel.
        d. Select the Advanced tab and click Environment variables option present in the bottome right corner.
        e. Add new System variable "MAVEN_HOME" with the path *<location of unzipped maven path>*. eg: C:\Program Files\ apache-maven-3.8.6
        f. Edit the System variable "Path" it will prompt you below dialog box, clicks on New button, and add this "%MAVEN_HOME%\bin"
        g. Test the configuration: Open command prompt and type:
            - mvn –version

3. Spring boot

a. Download the spring client zip file from https://repo.spring.io/ui/native/release/org/springframework/boot/spring-boot-cli/2.7.5/spring-boot-cli-2.7.5-bin.zip
b. Unzip the file at C:\Program Files\ spring-boot or any other convenient location.
c. Search for “Advanced system settings” in the windows search box and open the “View advanced system settings” present in the control panel.
d. Select the Advanced tab and click Environment variables option present in the bottom right corner.
e. Add new System variable “SPRING_HOME” with the path <location of unzipped maven path>. eg C:\Program Files\ spring-boot
f. Edit the System variable “Path” it will prompt you below dialog box, clicks on New button, and add this “%SPRING_HOME%\bin”
   Reference: https://docs.spring.io/spring-boot/docs/current/reference/html/getting-started.html

4. MySQL configuration
   a. Download the MySQL application and install MySQL with default configurations.
   b. Remember the password provided during installation.
   c. Install MySQL Workbench.
   d. After installation open the MySQL Workbench.
   e. Add a new database by selecting new schema -> “provide name as mimsdb” -> Apply. Create table with root privileges or grant all permissions if root not used.



In the application.properties file update the following based on the ceredentials used.
spring.datasource.username=<username>

spring.datasource.password=<Password>

5. Run spring boot application

    a. Navigate to the root (<git pulled location\mims>)of the project via command line and execute the command: **mvn spring-boot:run**

    b. Go to google chrome -> settings -> Privacy and security ->Security.

    c. In the Advanced security option select Manage device certificates.

    d. Click import in the following window and add keystore.cer from the following location:

    e. Application is now accessible from https://localhost:443/

To view the code changes in intellij:

1. Open intellij and select File ->open project -> (<git pulled location\mims>).
2. On opening allow intellij to install any unavailable dependency and Run the application from MimsApplication.
3. The test cases for the application can be found at <content root>/src/test.

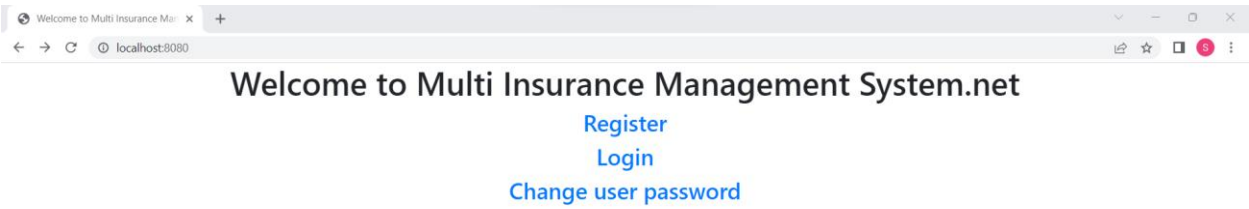
2 FUNCTIONALITY - Use cases

2.1 User Registration

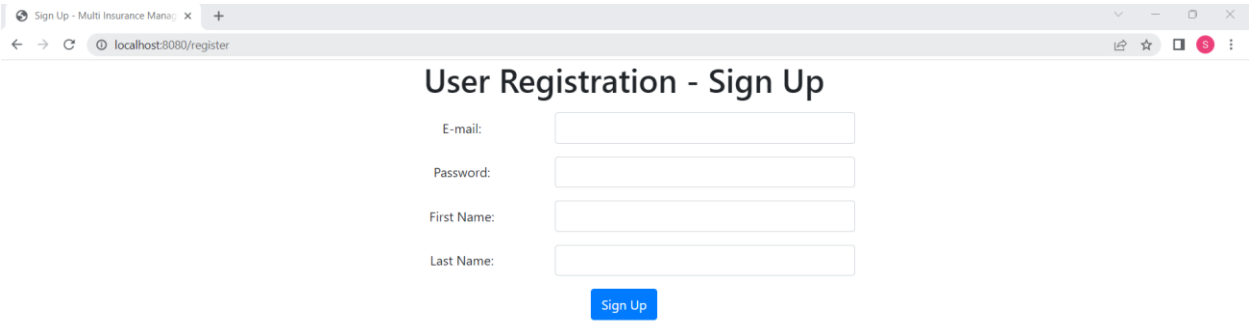
The user registration is implemented as part of ApplicationController.java -> "/register". This feature allows a new user to register as part of the Multi Insurance Management system. The user name will be the user email entered. The user information collected are user email, password, first name and last name. The password entered should have minimum 1 Uppercase letter, minimum 1 lowercase letter, minimum 1 digit, minimum 1 special character, no spaces and a minimum password length of 8 characters.

Registering a new user:

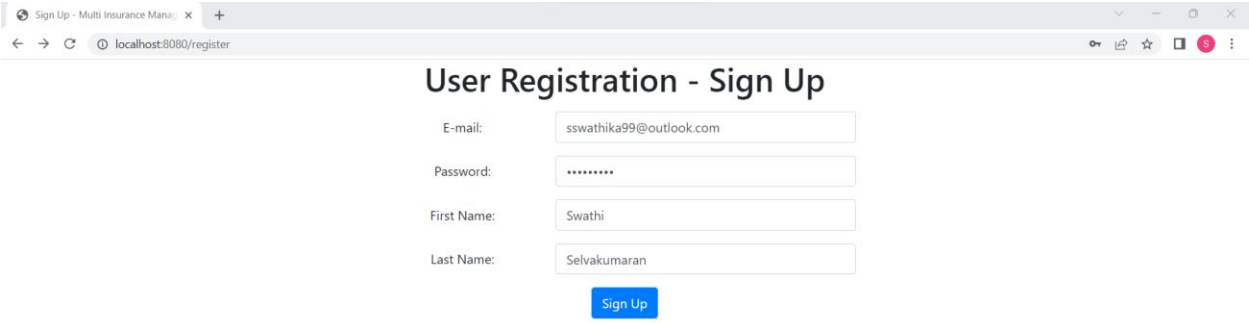
1. Select on Register button displayed when viewing https://localhost:443/.



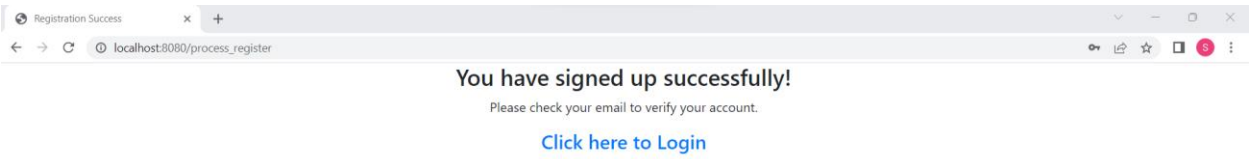
2. The user Registration page will pop up.

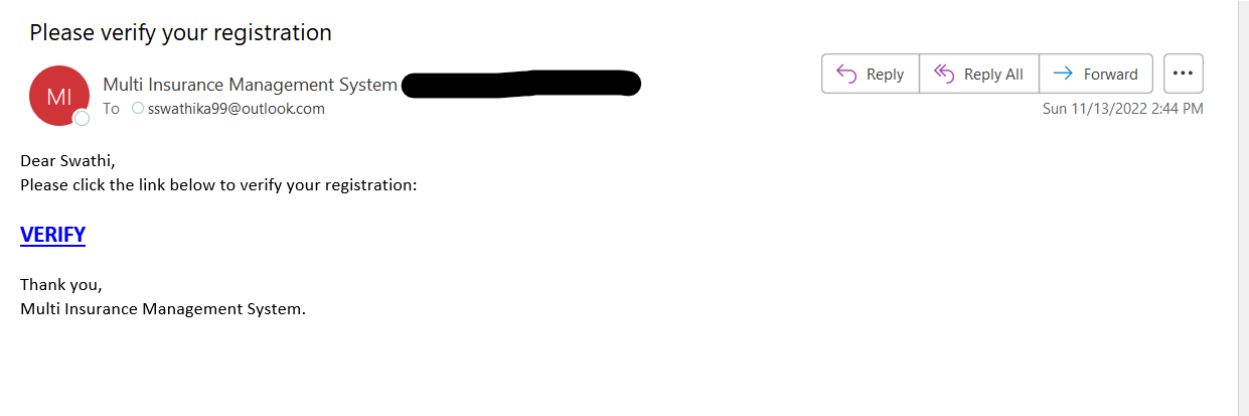


3. Enter valid Email address, password, First name and last name and select sign up button.



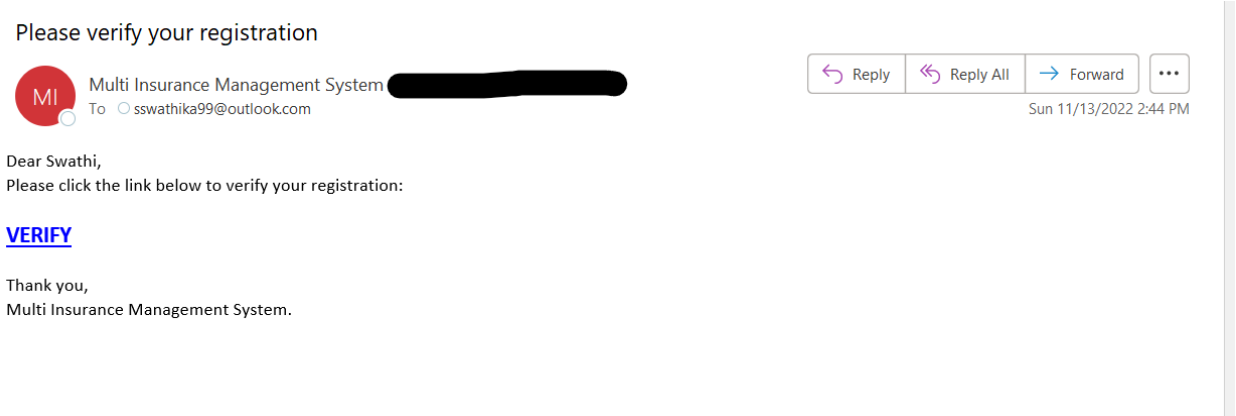
4. The user information will be successfully stored in the database and a user verification email would have been sent to the registered email.





2.2 Validate and notify new user Creation

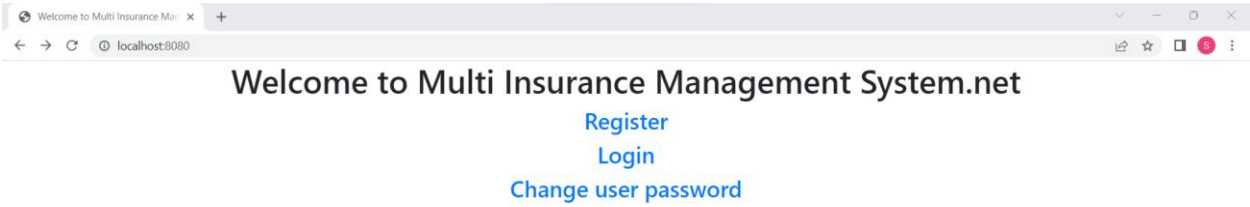
The new user created must be validated to prevent attackers from accessing the application. When a new user is registered a user verification email would have been sent to the registered email. When the registered email is verified the user can then access the application.



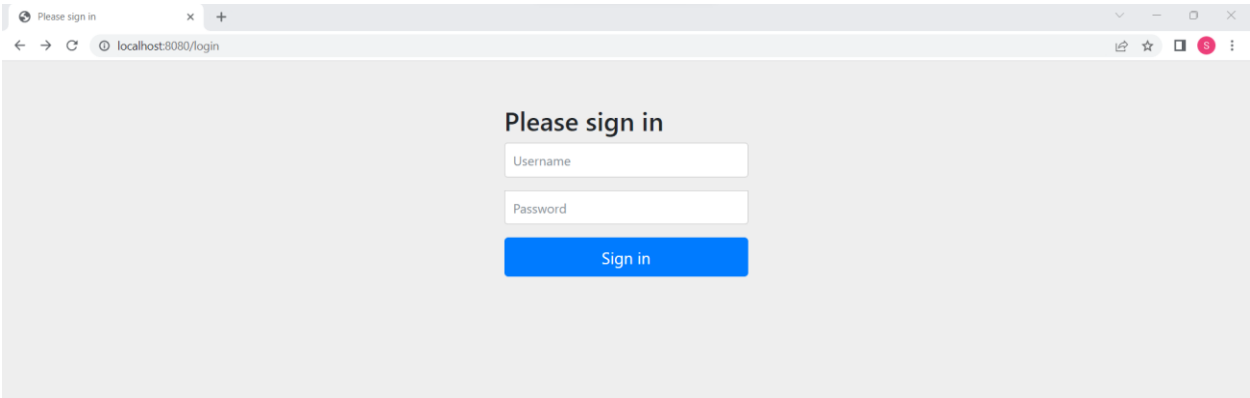
2.3 Login

The user login is implemented as part of spring security form-login. This feature allows a new user to register as part of the Multi Insurance Management system. The user name will be the user email used for registration. The user information collected are user email, password. When a user attempts to login without user id verification then the application denies login. When the user id has been verified and the user credentials are valid then the user login is successful.

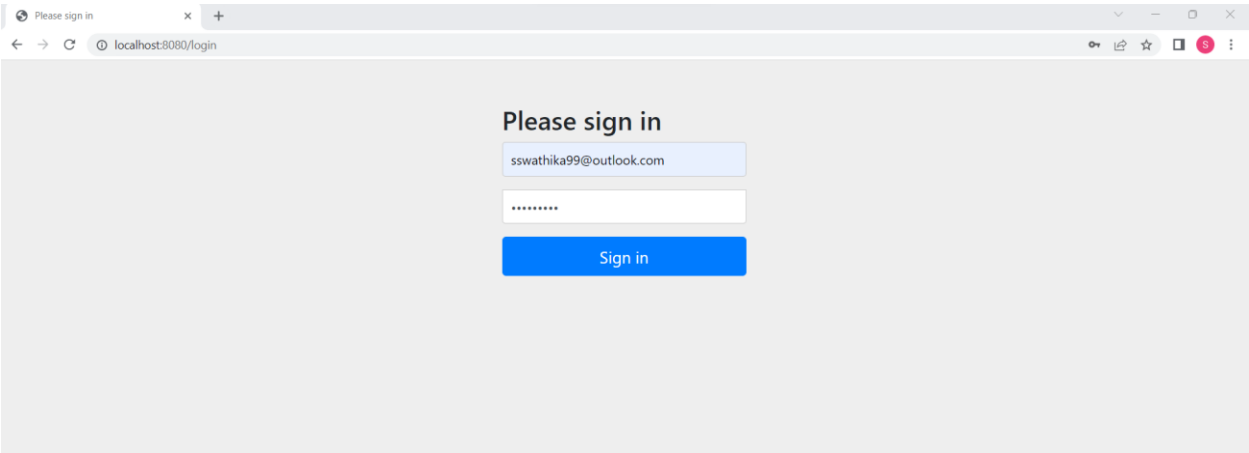
1. Select on Login button displayed when viewing https://localhost:443/.



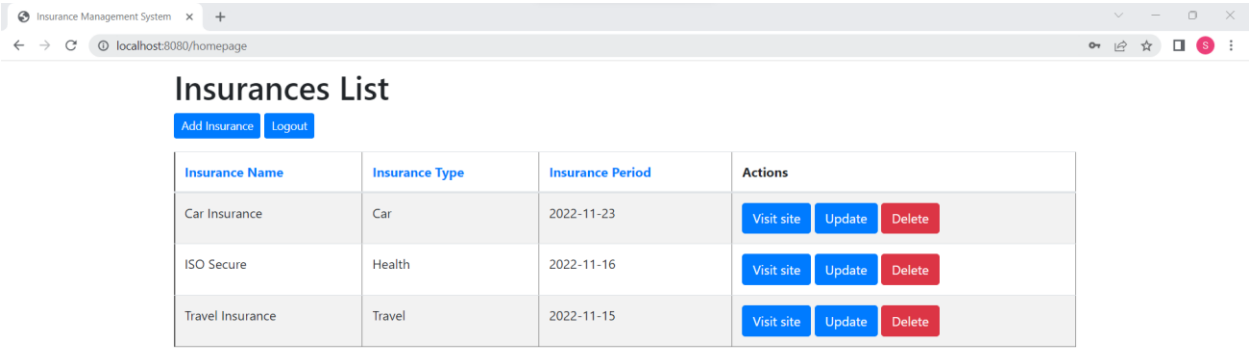
2. The login page will pop up.



3. Enter valid user name and password and select Sign in.



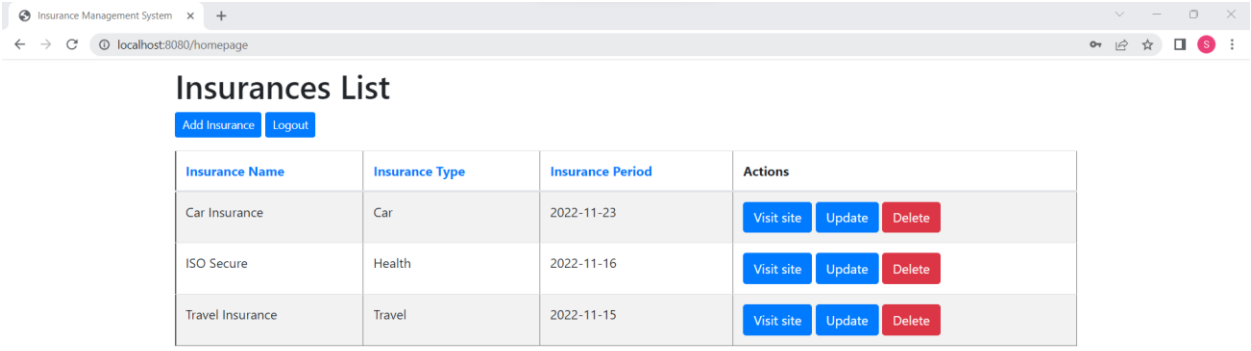
4. The user will be logged into the application and the user can view the list of insurances added.



2.4 View Home Page – view user insurances

On successful login the user will be direct to the ApplicationController.java -> "/homepage". This page lists all the user insurances managed by the Multi Insurance Management System.

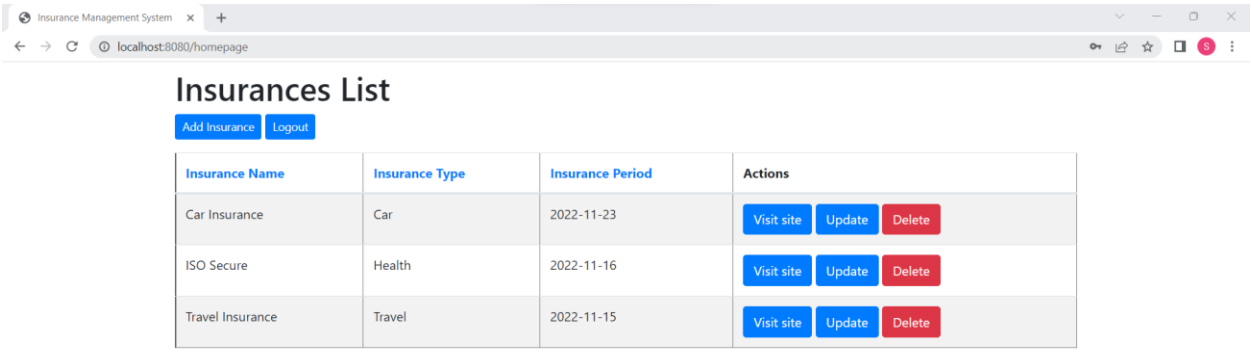
Upon successful login the following page will be viewed.



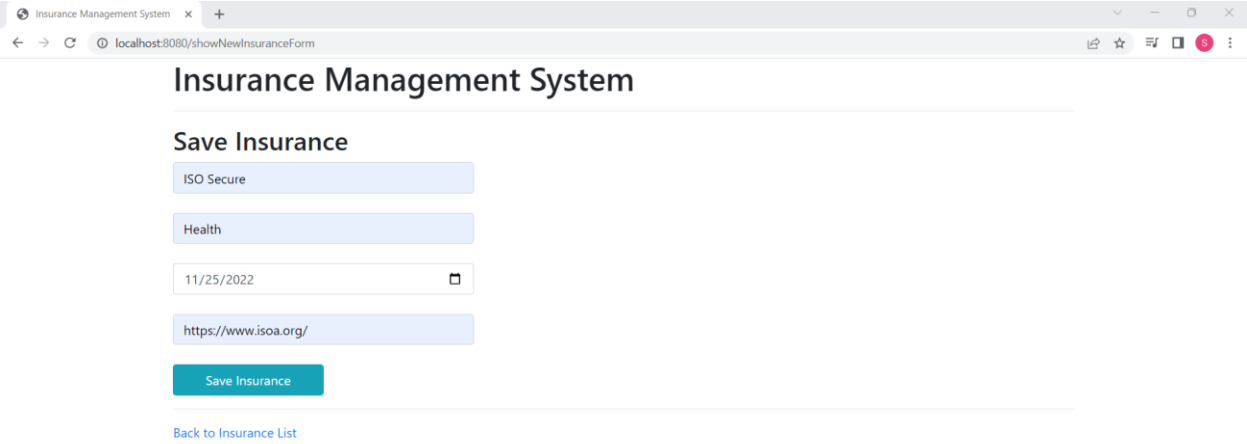
2.5 Add Insurance

The option to add insurances to be managed can be selected in the home page. The add insurance redirect the user to insurance form page where the user can add the insurance with details such as insurance name, insurance type, insurance link and insurance expiry date.

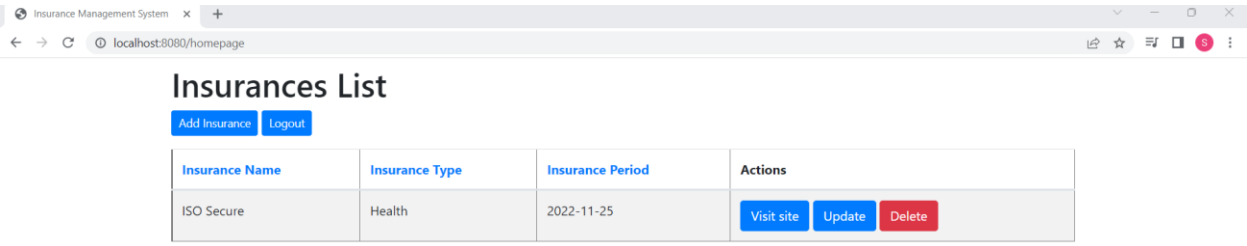
1. Select the Add Insurance button in the home page.



2. Add the valid insurance details and save the details.

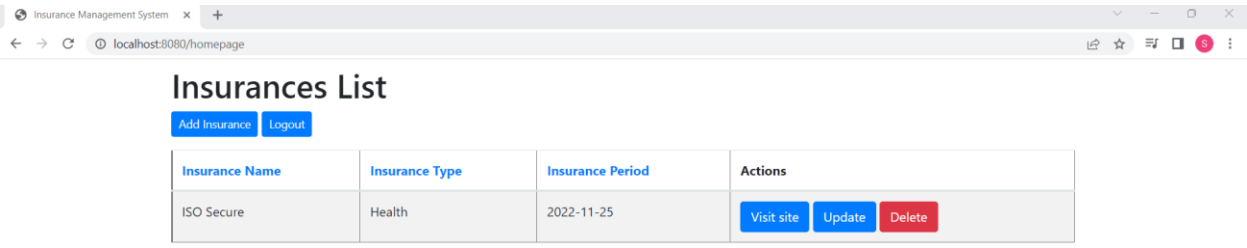


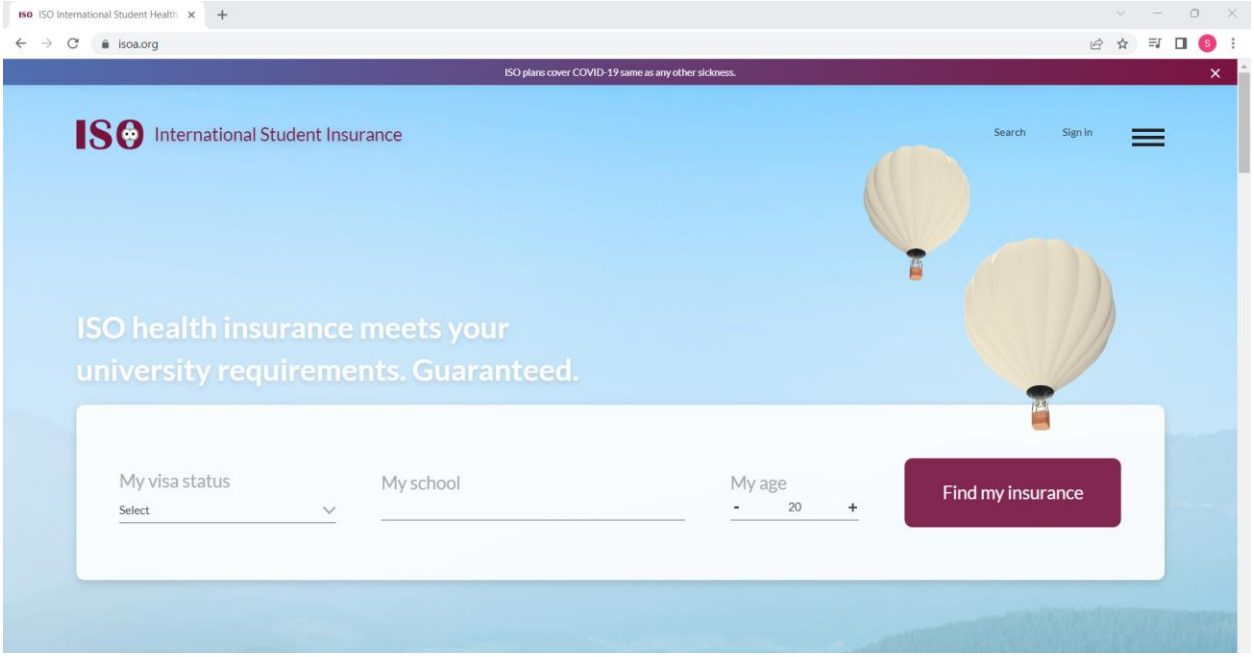
3. The new insurance can be viewed in the homepage.



2.6 View Insurance policy

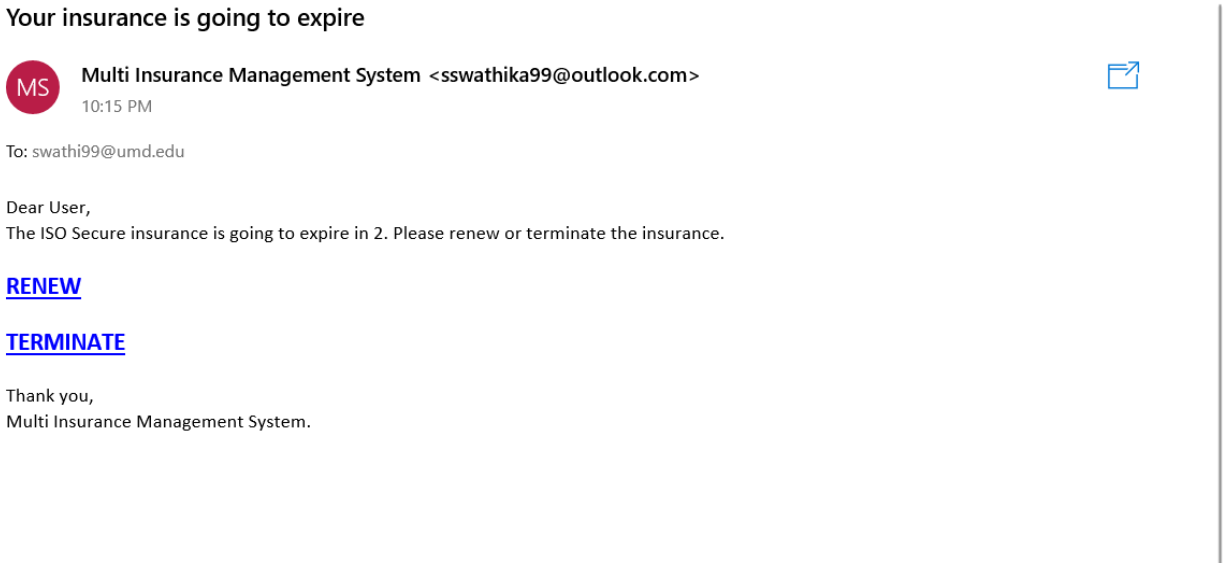
In the home page a Visit site button is present beside all the insurances. On selecting this option the user is directed to insurance policy page.





2.7 Track Insurance Status

The Insurance expiry date is tracked by the Multi Insurance Management System daily. A notification mail is sent to the user informing the upcoming insurance expiry with the options to renew/terminate.



2.8 Notify insurance Expiry

The Insurance expiry date is tracked by the Multi Insurance Management System daily. A notification mail is sent to the user informing the upcoming insurance expiry with the options to renew/terminate.

Your insurance is going to expire

MS **Multi Insurance Management System <sswathika99@outlook.com>**
10:15 PM

To: swathi99@umd.edu

Dear User,
The ISO Secure insurance is going to expire in 2. Please renew or terminate the insurance.

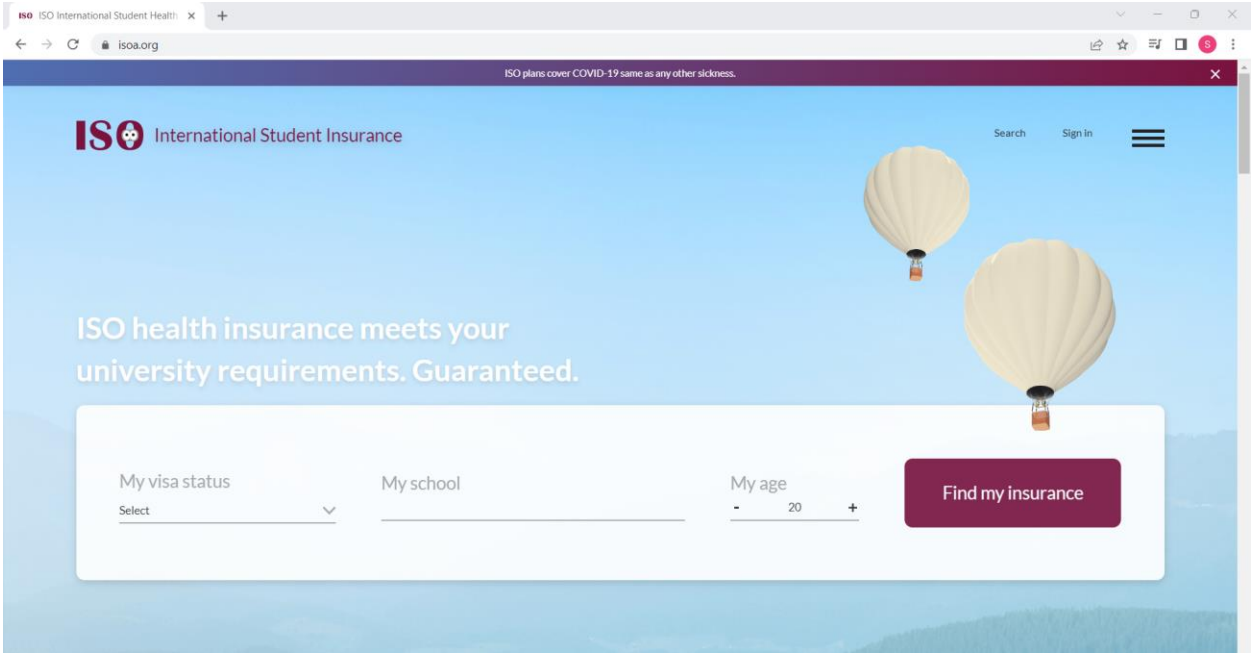
RENEW

TERMINATE

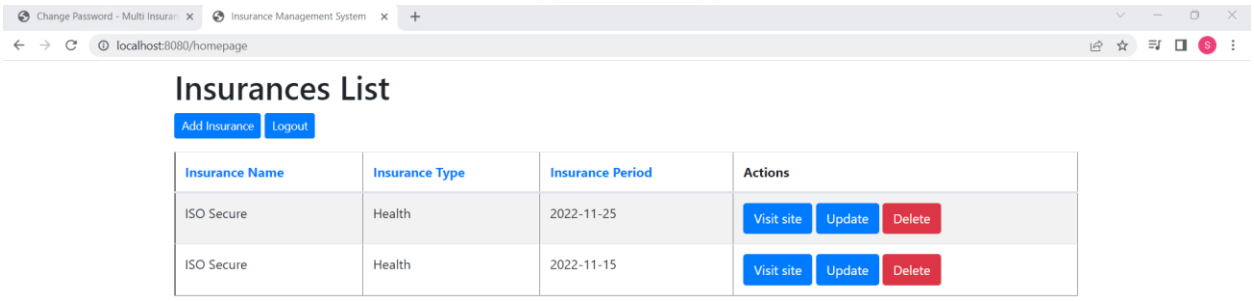
Thank you,
Multi Insurance Management System.

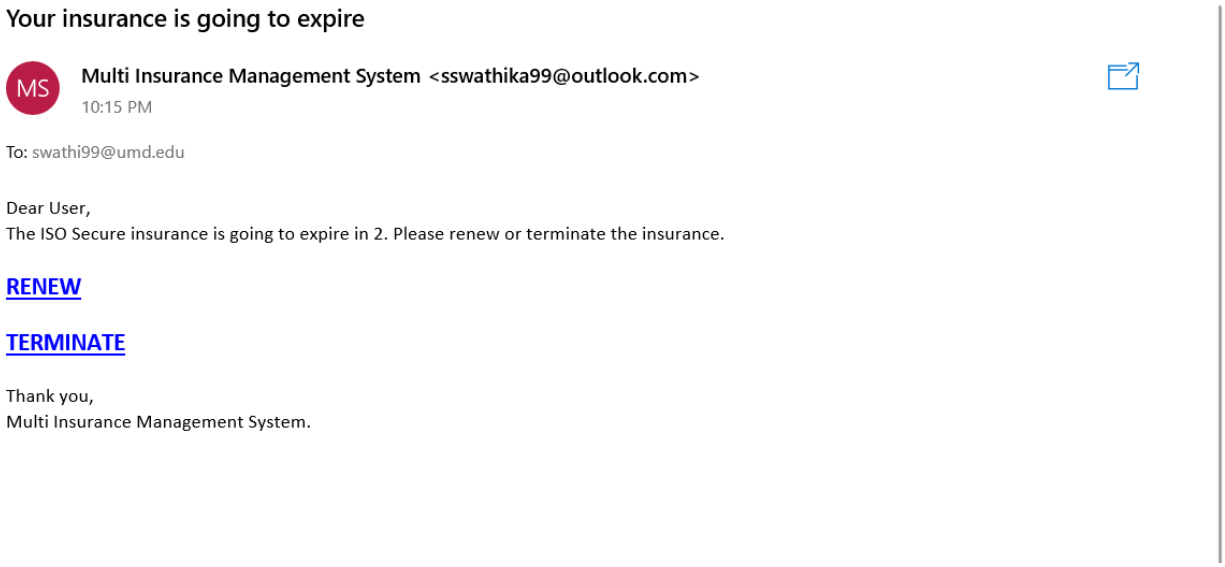
2.9 Renew or Terminate Insurance

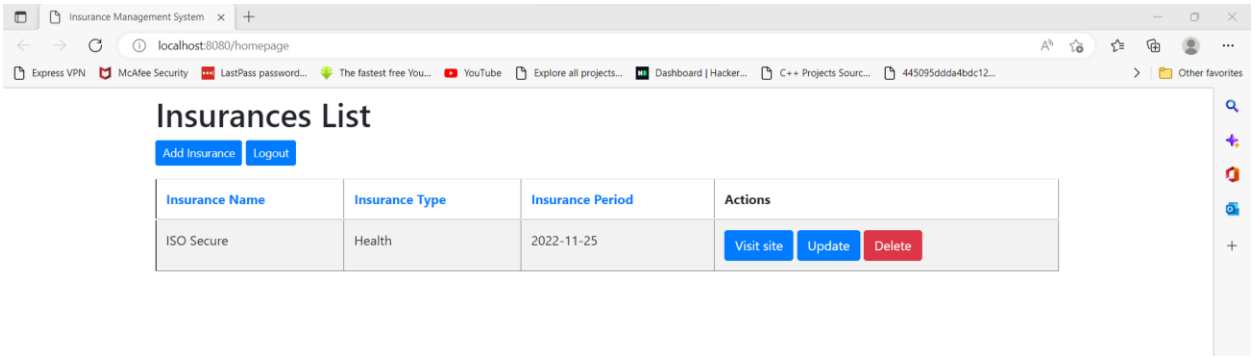
RENEW: Selecting the renew option directs the user to insurance policy location.



TERMINATE: Selecting the terminate option deletes the insurance from the insurance list.

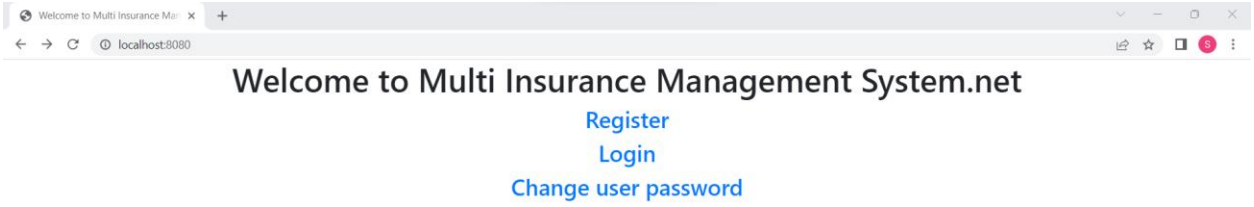


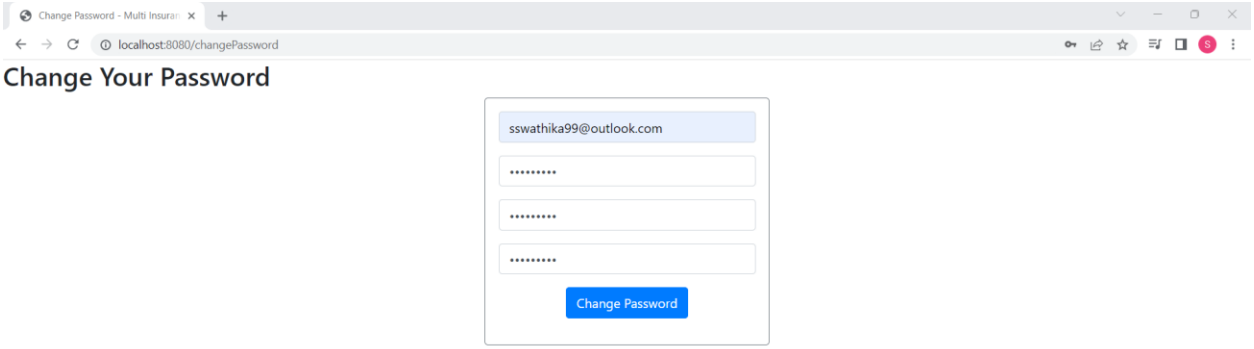




2.10 Change User password

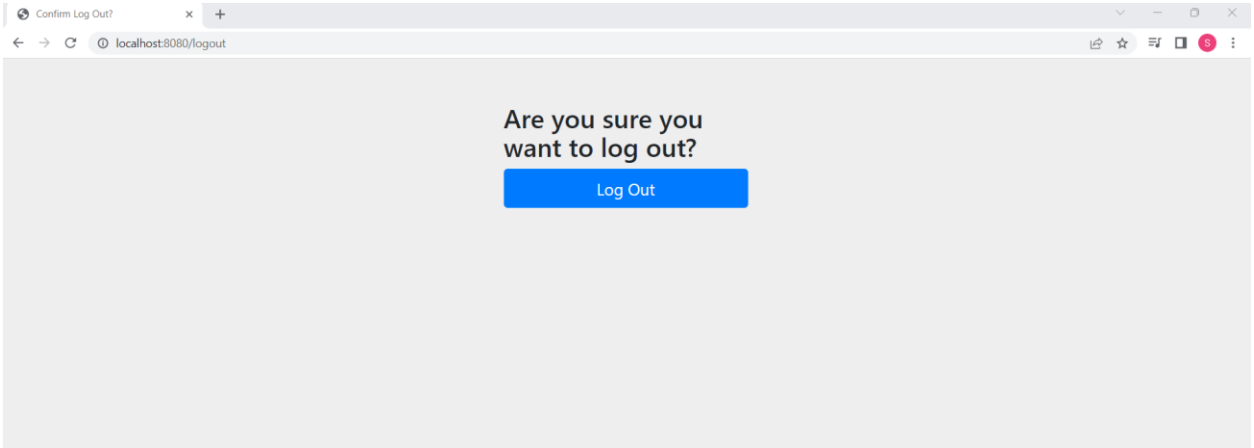
The user can change their password by selecting the change password option in the https://localhost:443/. Enter the user name, old password and new password and select change password.





2.11 Logout

In the home page a Logout button is present above the insurance list. On selecting this option the user is directed to the logout confirmation page. In the confirmation when the log out is confirmed user is logged out by clearing the session.



3 TEST CASE

1. Register user with valid input.
2. Register user with invalid input.
3. Don't register existing user.
4. Login a valid and verified user.
5. Validate the user manually by verifying through mail.
6. Validate the token expiry after user is validated.
7. Find the user by username.
8. Track user Insurances and notify the users of possible insurance expiry.
9. Create a new insurance.
10. Update an existing insurance.
11. Create invalid insurance with invalid url.
12. Delete an insurance.

Note: Some of then are manual testing.

Test by running the class MimsApplicationTests.class which will execute existing tests.

4 SECURITY IMPLEMENTATION

1. Brute force attack prevention: User names are configured with lockout mechanism to ensure that brute force attack. The user will be locked for 24 hours if the number of invalid attempts is more that 5.
2. CSRF Protection: The Spring security by default configures prevents the application from csrf attacks.
3. Session management: The session is set to invalid at logout.
4. XSS protection is provided by http.headers().xssProtection()
5. Password Complexity: The password is set based on regex pattern matching with the constraints:
   Minimum 1 Uppercase letter
   Minimum 1 Lowercase letter
   Minimum 1 Special character
   Minimum 1 Digit
   Minimum password length of 8 characters.
   No white spaces allowed.
6. Password hashing: Password is hashed using secure BCryptPasswordEncoder to save the hashed password.
7. Secure transport of request: Secure password transportation has been implemented by adding the self-signed ssl certificate to the application.

5 REFERENCES

1. https://www.codejava.net
2. https://www.geeksforgeeks.org/unit-testing-in-spring-boot-project-using-mockito-and-junit/