


CostsWatcher

CostsWatcher is a web-based application which has the goal of making the tracking of travel costs easier. It achieves this by offering a feature where people can create groups and post any individual or group expenses incurred during the trip. At any moment, everyone in the group can easily see the total amount of money spent until that moment. Also, everyone can see their individual expenses. This kind of information is available via generated reports. One of the features has the goal of informing the group members when they take part in a group expense. For any unexpected changes during the trip, group members can add new individuals to the group any time and can edit the posted expenses. To make things better, people can take advantage of the application by creating user accounts. This way, the application keeps track of their past groups and expenses.

The features currently implemented in the CostsWatcher App are Authentication, Groups and Expenses.

1. Authentication

A screenshot of the CostsWatcher application's login screen. The background is a solid light yellow color. In the center, the text "CostsWatcher" is displayed in a bold, black, sans-serif font. Below this text, there are two blue rectangular buttons with white text. The top button is labeled "Sign In" and the bottom button is labeled "Sign Up". Both buttons have a slight shadow effect.

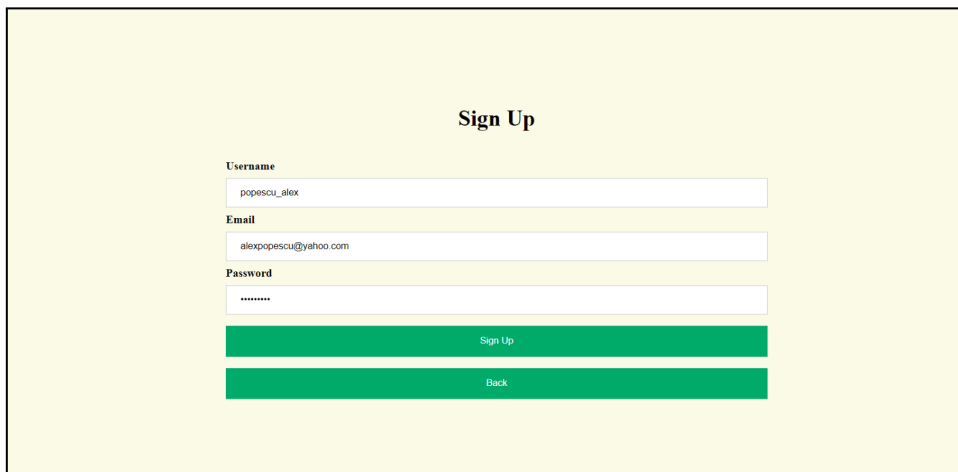
CostsWatcher

Sign In

Sign Up

The authentication system is used for creating and retrieving users in a secure way.

When creating a user, the required inputs are “username,” “email” and “password” to be able to later identify him by the correct data.

A screenshot of the "Sign Up" form in the CostsWatcher application. The background is a solid light yellow color. At the top center, the text "Sign Up" is displayed in a bold, black, sans-serif font. Below this, there are three input fields, each with a label above it: "Username", "Email", and "Password". The "Username" field contains the text "popescu_alex". The "Email" field contains the text "alexpopescu@yahoo.com". The "Password" field contains a series of dots. Below the input fields, there are two green rectangular buttons with white text. The top button is labeled "Sign Up" and the bottom button is labeled "Back". Both buttons have a slight shadow effect.

Sign Up

Username

popescu_alex

Email

alexpopescu@yahoo.com

Password

Sign Up

Back

Retrieving users is done using the “username” and “password” fields, the user is found in the database by the “username” field and the “password” is used to ensure the users data is kept safe.

Sign In

Username

popescu_alex

Password

Sign In

Back

2. Groups

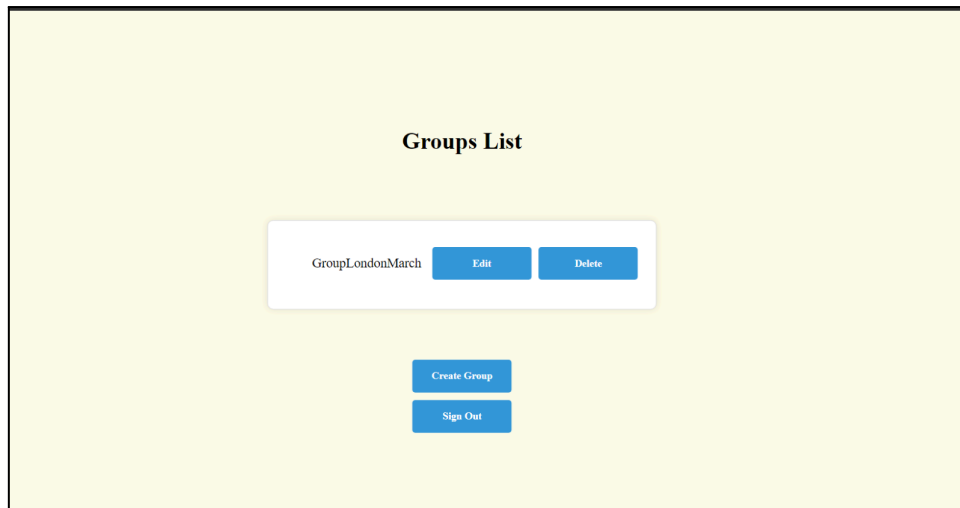
Create Group

Group Name

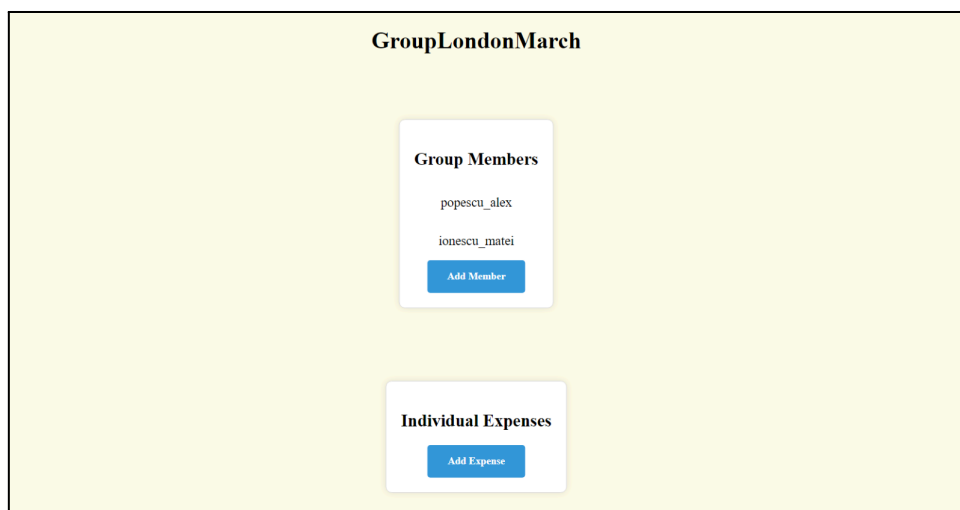
Enter Group Name

Create

Back



The groups are used to keep collections of users and expenses for a given trip. They are created by the logged-in user, and he can invite other users to the trip, so they can see the other trip participants and expenses incurred for that trip.



When inviting a new trip participant, you search them by their username, and if the username is not associated with any user an error message will appear.

Add new member

Username

ionescu_matei

Add

Back

3. Expenses

Expenses are the main functionality of the application, and they are of two types: Individual and Group Expenses.

Group Members

popescu_alex

ionescu_matei

Add Member

Individual Expenses

Add Expense

Group Expenses

Add Expense

The Individual Expenses are associated with an individual group and user. They are useful for keeping track of a person's expenses that are not shared with any other users.

Add new individual expense

Expense name

Dinner

Expense amount (\$)

30

Add

Back

Edit individual expense

Added by

popescu_alex

Date

01/16/2024 10:46:57.654 AM

Expense name

Dinner

Expense amount (\$)

30

Save

Back

The Group Expenses are associated with an individual group and multiple users, and they work like a collection of multiple Individual Expenses would. Their purpose is to reduce the effort of inserting the expense information by each user from his personal account, and delegate this task to only one user that can add the information for all users participating in each expense.

Add group expense

Name of the expense

Lunch

First member's name

popescu_alex

First member's amount (\$)

40

Second member's name

ionescu_matei

Second member's amount (\$)

35

Add

Back

Edit group expense

Expense name

Lunch

Save

Date

01/16/2024 10:47:31 -275 AM

Group member

popescu_alex

Amount (\$)

40

Delete

Group member

ionescu_matei

Amount (\$)

35

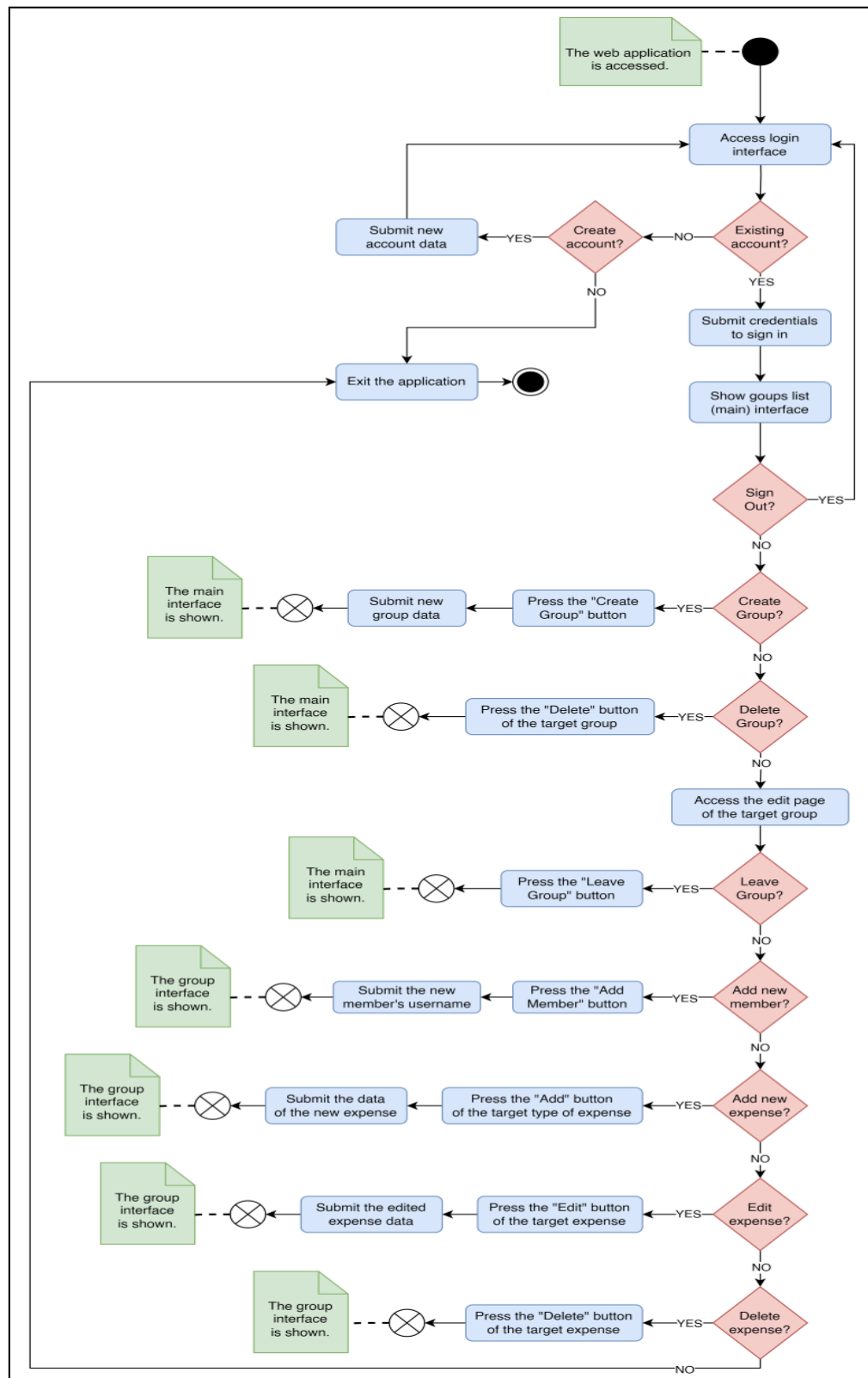
Delete

Add participant

Back

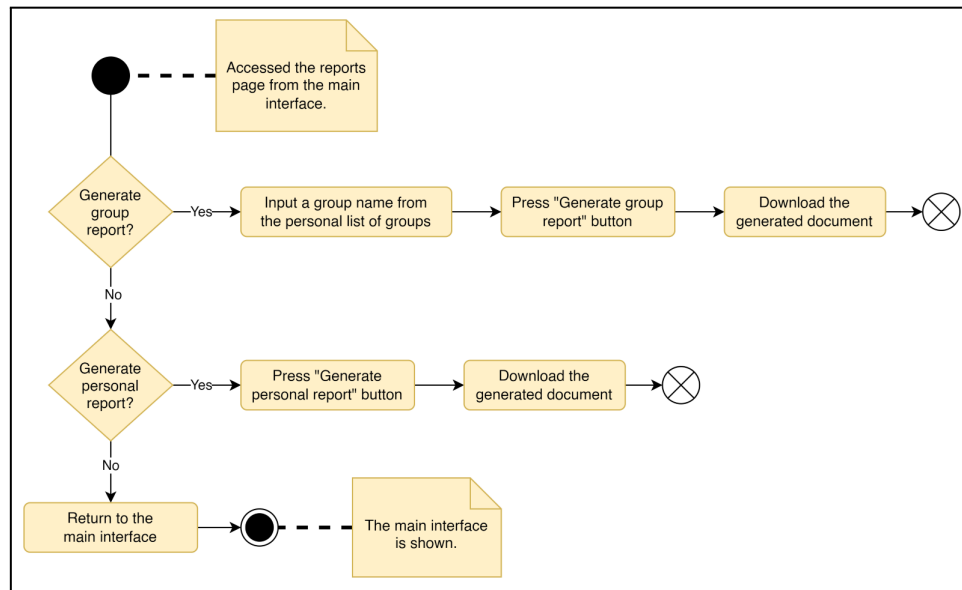
Diagrams:

1. General Diagram



General Workflow Diagram

2. Reports Diagram

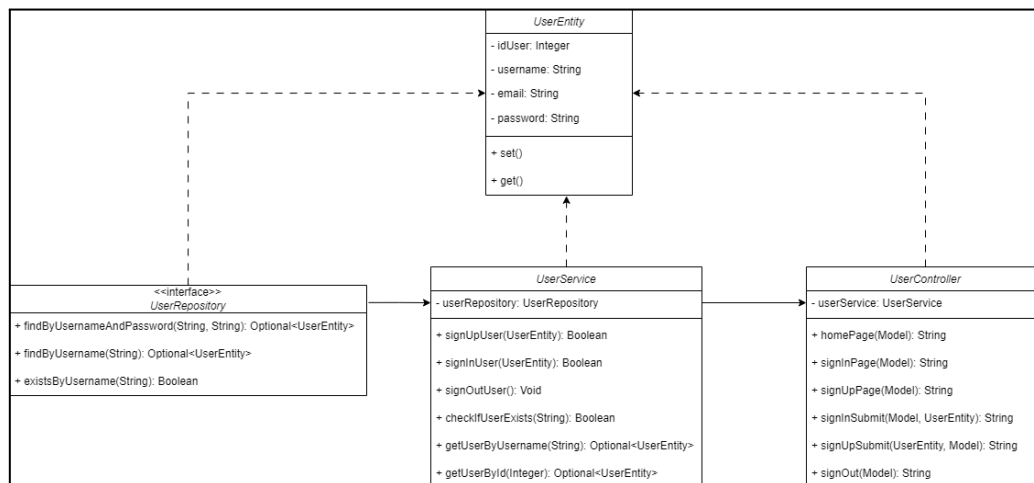


Reports Workflow Diagram

The Group Report is used by any user of a given group to calculate the total sum spent by all the group's participants.

The Personal Report is used by a user to calculate the total sum spent in each of the groups he is a part of. Each user can only access his personal report.

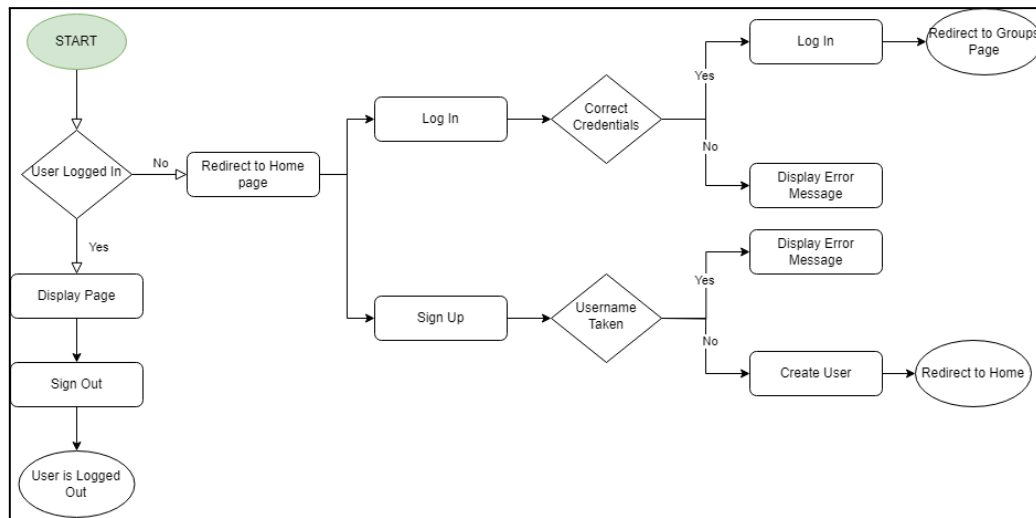
3. Authentication Diagrams



Authentication Class Diagram

The classes follow a certain pattern (because of the Spring framework). The Authentication (Users) feature has a Controller, Service, Repository and Entity classes. In Spring, the developer declares the Repository as an interface used by the framework to generate the class. In short, the Controller handles the HTTP request by calling the Service, the Service performs the bulk of

processing and uses the Repository to make database related operations. An Entity class represents a table in the database where its operations are available through its associated Repository.



Authentication Workflow Diagram

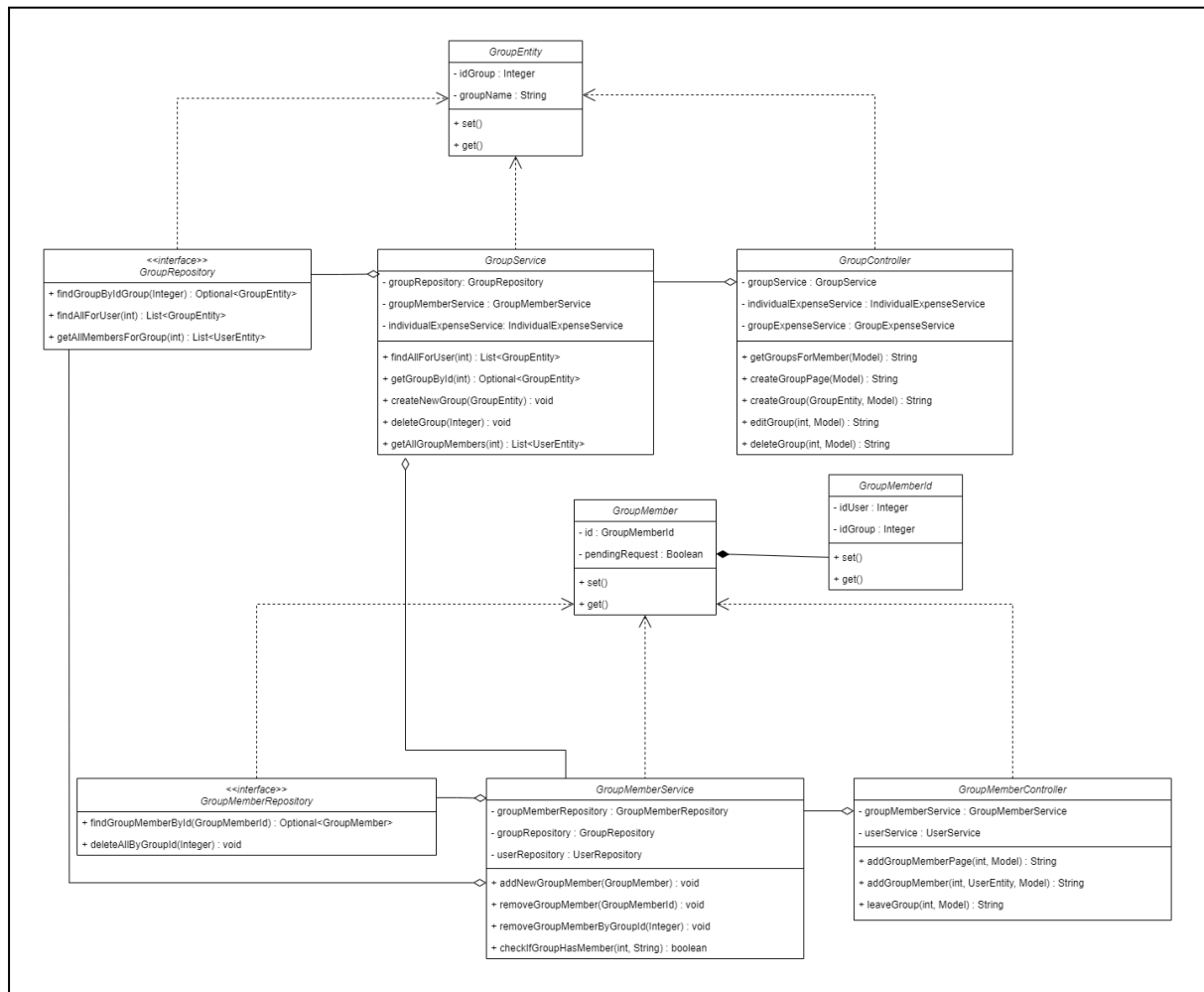
The workflow diagram for the authentication (users) starts on the Home page, which can take two forms: LoggedIn Home page (Groups list page for the logged in User), NotLoggedIn Home Page, which displays the options to go to the Log In Page or the Sign Up Page.

From the LoggedIn Home Page, the user can Log Out, in which case his session is ended.

From the Sign Up Page, the user can register himself, assuming the data he entered is not already used by another user.

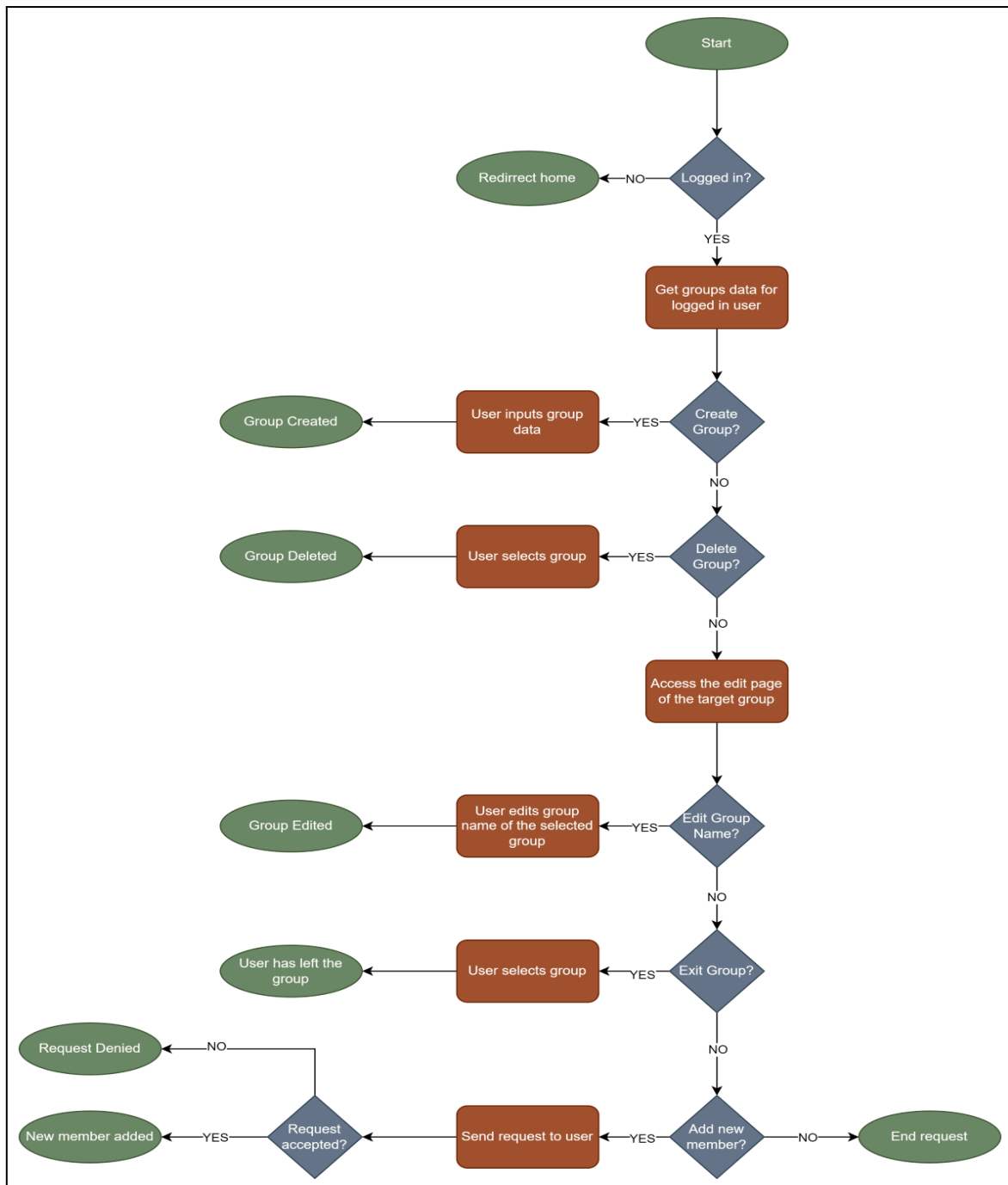
From the Log In Page, the user can enter his username and password, and, after validation is either logged in or shown an error.

4. Groups Diagrams



Groups Class Diagram

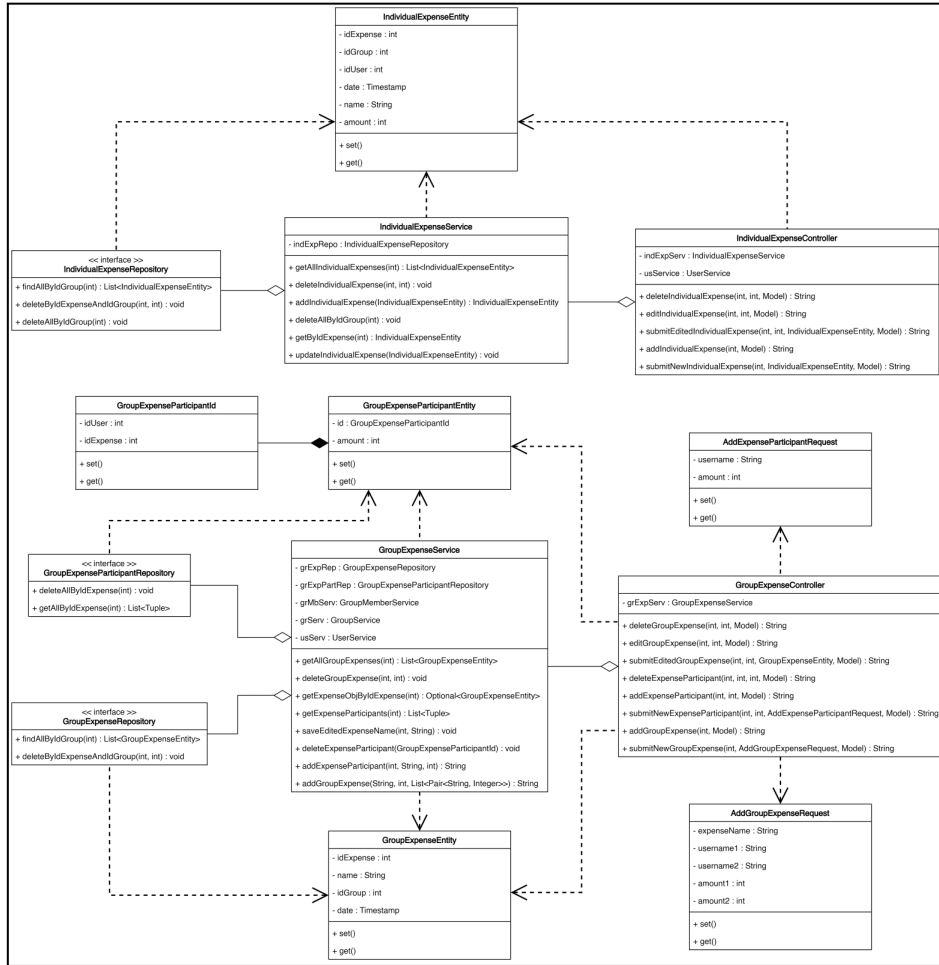
The classes follow a certain pattern (because of the Spring framework). The Groups feature has a Controller, Service, Repository and Entity classes. In Spring, the developer declares the Repository as an interface used by the framework to generate the class. In short, the Controller handles the HTTP request by calling the Service, the Service performs the bulk of processing and uses the Repository to make database related operations. An Entity class represents a table in the database where its operations are available through its associated Repository.



Groups Workflow Diagram

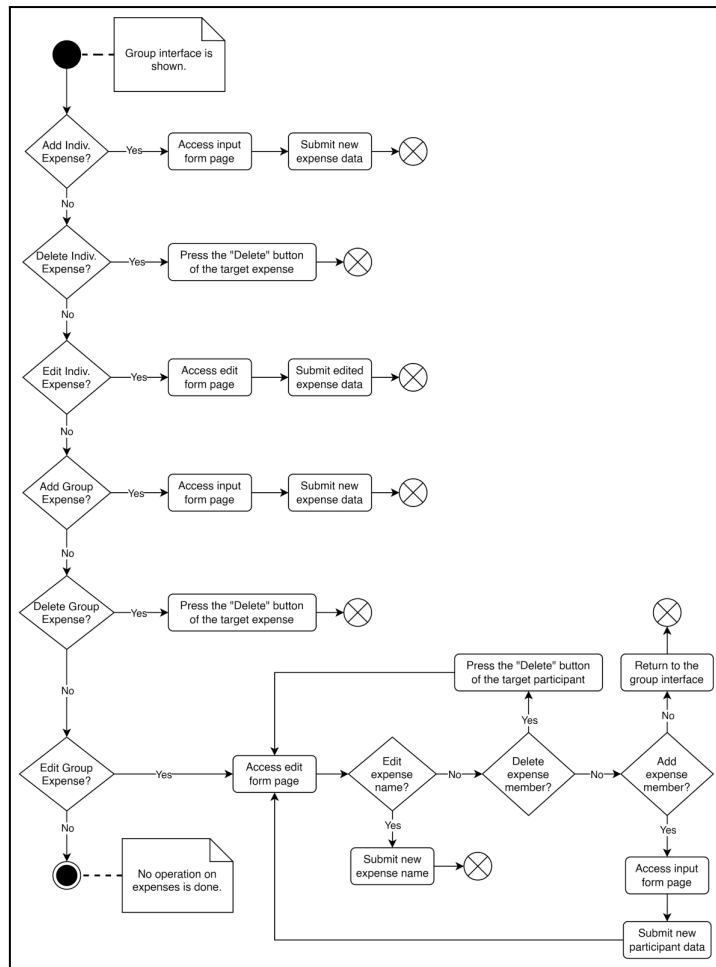
The workflow diagram for the groups starts when on the Home page, where the UI of the list of the user's groups is shown. The user can choose to perform add/edit/delete operations on the group.

5. Expenses Diagrams



Expenses Class Diagram

The class diagram for the expenses management part of the software is composed of two parts: the classes for the individual expenses feature and the ones for the group expenses feature. The classes for each of those features follow a certain pattern (because of the Spring framework). Each feature has Controller, Service, Repository and Entity classes. In Spring, the developer declares the Repository as an interface used by the framework to generate the class. In short, the Controller handles the HTTP request by calling the Service, the Service performs the bulk of processing and uses the Repository to make database related operations. An Entity class represents a table in the database where its operations are available through its associated Repository. Additionally, the group expenses feature has two classes for the DTO concept of Spring (the Request classes).

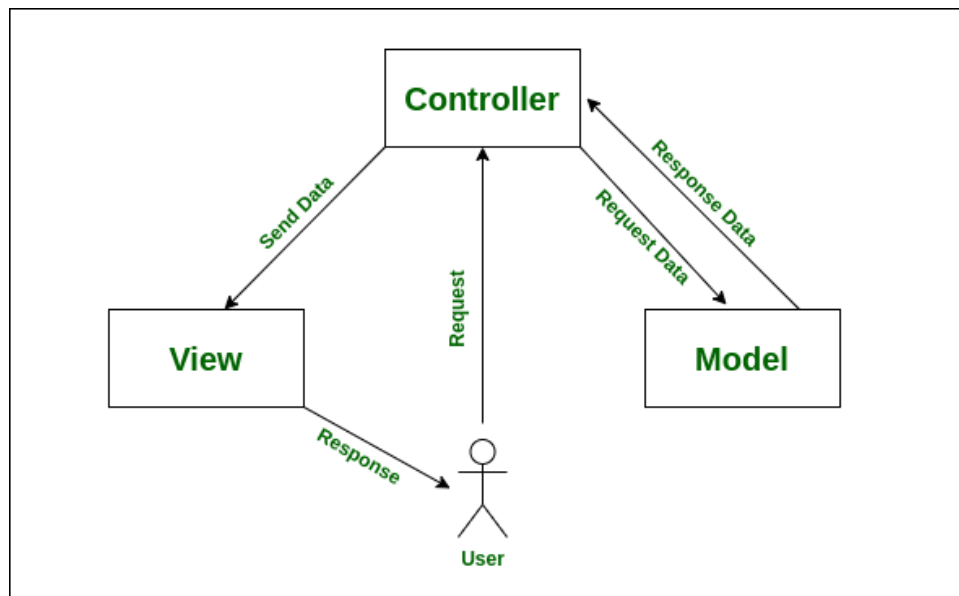


Expenses Workflow Diagram

The workflow diagram for the expenses management part of the software starts when the UI of an individual group is shown. When the group UI is shown, the user can choose to perform add/edit/delete operations on the individual or group expenses lists. At the end of any operation (when any "end of flow" symbol is reached), the group interface is shown again (returning to the starting point of the diagram).

Design Patterns Used

- **MVC**



The Spring Web model-view-controller (MVC) framework is designed around a **DispatcherServlet** that dispatches requests to handlers, with configurable handler mappings, view resolution, locale, and theme resolution as well as support for uploading files. The default handler is based on the `@RestController` and `@RequestMapping` annotations, which is what we have used in the development of the application.

- **Singleton**

- **Spring Beans**

Spring restricts a singleton to one object per Spring IoC container. In practice, this means Spring will only create one bean for each type per application context.

By default, Spring creates all Beans as singletons.

When we define a bean with the singleton scope, the container creates a single instance of that bean.

All requests for that bean name will return the same object. Any modifications to the object will be reflected in all references to the bean.

- **Dependency Injection**

- **Autowired**

Dependency injection is a pattern we can use to implement Inversion of Control (IoC), where the control being inverted is setting an object's dependencies.

Connecting objects with other objects, or “injecting” objects into other objects, is done by an assembler rather than by the objects themselves.

We used the Field-Based Dependency Injection by taking advantage of the already-implemented annotation in the Spring framework, “**@Autowired**”. Wiring allows the Spring container to automatically resolve dependencies between collaborating beans by inspecting the beans that have been defined (“`@Repository`”, “`@Service`”, “`@Controller`”).

Work Contribution

- Culea Ionel-Alexandru -> 35%
 - Expenses Feature
 - Expenses Class Diagram
 - Expenses Workflow Diagram
- Lazăr Mihai -> 35%
 - Groups Feature
 - Groups Class Diagram
 - Groups Workflow Diagram
- Şofei Andrei-Adrian -> 30%
 - Authentication Feature
 - Authentication Class Diagram
 - Authentication Workflow Diagram