PUSS214204

v. 1.0

TimeMate

Software Top Level Design Document

Group 2

Responsible: System Group

Authors: Developer Group

2021-02-25

Contents

1 Document History

Version	Date	Responsible	Description
0.1	2021-02-11	UG	Document created.
0.2	2021-02-17	UG	Ready for informal review.
0.3	2021-02-18	UG	Corrected grammar and more after informal review
0.4	2021-02-25	UG	Made corrections after formal review.
1.0	2021-03-01	UG	Corrected a header and a diagram. Reached baseline.

2 Introduction

This document describes the design of TimeMate [1.1]. The system is a further development of the system BaseBlockSystem [1]. TimeMate hosts various functions related to the creating of time reports, getting a view of reported time and more.

3 Reference Documents

1. Software Requirements Specification: TimeMate, v. 1.1, Doc. number: PUSS214201
2. Software Top Level Design Document: BaseBlockSystem, v 1.0, Doc. number: PUSS12002

4 Terminology

- **Java Server Page (JSP):** Server-side technology that enables the creation of dynamic views.
- **Servlets:** Java programs that run on the server side.
- **Java Beans:** A Java class that only contains set/get methods with private attributes. Moreover, a Java Bean must implement the Serializable interface.

5 Architecture

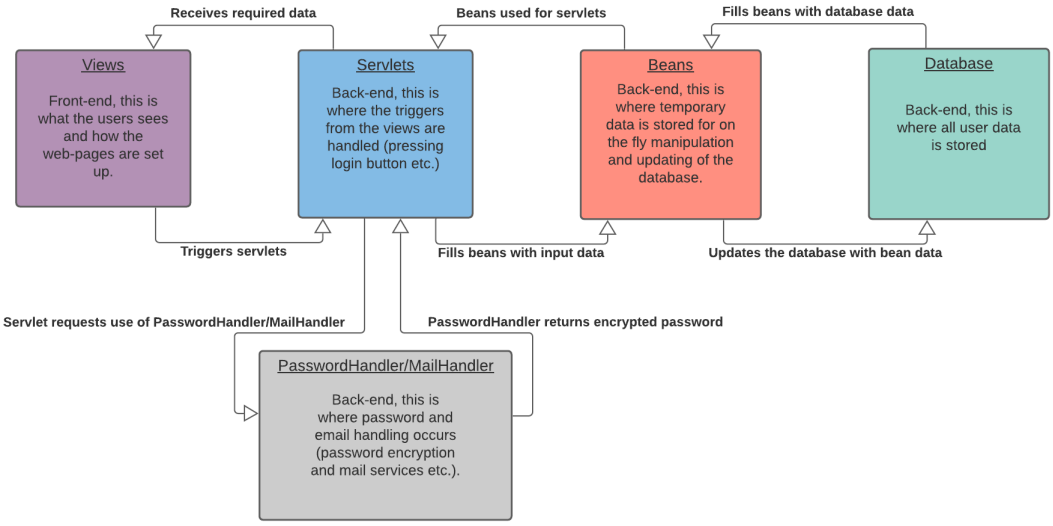

Figure 1: Diagram showing the overal architecture of the system

6 Class Overview

This system is developed using Apache Tomcat, wherein the controllers (Java Servlets) manipulates and sends data to the views (JavaServer Pages).

6.1 Java Server Pages

Below are the pages used for TimeMate and what their functions are.

6.1.1 login.jsp

This view is shown when a user that is not logged in tries to access the system. This is the only page in the TimeMate system a user can get access to without having an account. The page consists of two fields, one for username and one for password, as well as an option to request a new password in case the user has forgotten their password. If the user chooses to reset their password, a pop up is shown where the user can enter their e-mail which the new password will be sent to.

6.1.2 index.jsp

This view is shown when a user has logged in. From here, the user can access the menu which is dynamically updated depending on what group the user belong to. The index page contains an overview of TimeMate.

6.1.3 newreport.jsp

This view shows an empty time report. The user can fill in the time report and submit it.

6.1.4 editreport.jsp

This view is used by users that want to edit their previously reported time reports. When being here, the user can select a submitted report to edit.

6.1.5 summaryreport.jsp

This view represents a summary of a user's previously reported time reports. The user can select a time report to show detailed information about it.

6.1.6 signreport.jsp

This view is used by the project leaders for signing and unsigning time reports.

6.1.7 usermanagement.jsp

This view shows the user management page. The view is used by the administrator and the project leaders. The view is used for assigning roles to project members.

6.1.8 administration.jsp

This view shows the administration page. Only the administrator has access to this view. The view is used for administration purposes such as adding and removing project members.

6.1.9 changepassword.jsp

This view is used by all users to change their password. The page contains three fields, one for the user's current password, and two for the new password (in order to confirm it).

6.2 Java Beans

The files below are the Java Beans that are used to send information from the server to the client view. When created, beans are stored in the current session and discarded when the session ends.

6.2.1 UserBean

This class contains user information required to execute queries and updates related to the logged in user.

6.2.2 TimeReportBean

This class contains a list of users and their time reports which is required to render the report.jsp view.

6.2.3 TimeReportManagementBean

This class contains a list of signed time reports which is required to render signReport.jsp.

6.2.4 UserManagementBean

This class contains a list of users (excluding project leaders) and their roles which is required to render the userManagement.jsp view.

6.3 Servlets

The servlets below are the controllers for the system.

6.3.1 LoginServlet

Communication link between the view login.jsp and the bean UserBean.

6.3.2 PasswordChangerServlet

Servlet used when a user changes their password.

6.3.3 TimeReportServlet

Communication link between the views viewReport.jsp, newReport.jsp and summaryReport.jsp and the bean ReportBean.

6.3.4 TimeReportManagementServlet

Communication link between the views editReport.jsp and signReport.jsp and the bean TimeReportManagementBean

6.3.5 UserManagementServlet

Communication link between the view userManagement.jsp and the bean UserManagementBean.

6.4 Other classes

The classes below are never accessed by the user in any way, but are instead helper classes to the servlets.

6.4.1 MailHandler

Class used to send e-mails to users upon account creation and password changes.

6.4.2 PasswordHandler

Class used to encrypt and generate new passwords.

6.4.3 Database

Class used to establish connection between Tomcat and the database as well as executing all queries to the database.

7 Database

In this project, there is one database that consists of the nine relations shown in Figure 1. Every relation has one primary key each, which are the following: *userName* and *reportID*.

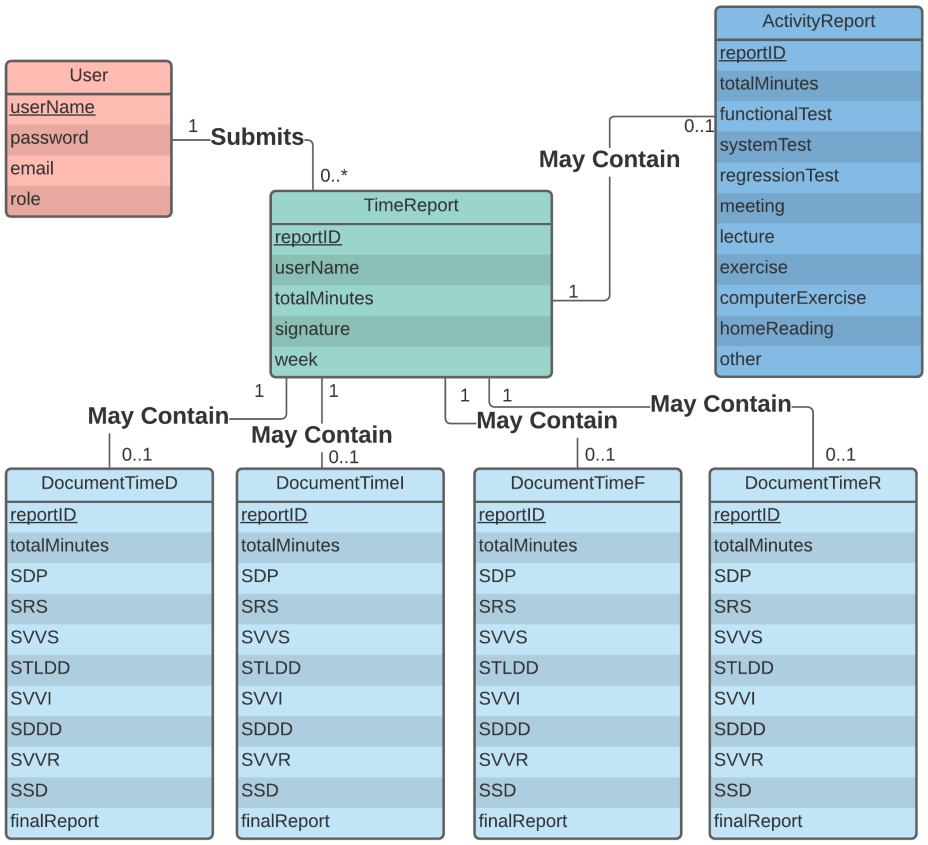

Figure 2: The database consists of the following relations: *User*, *TimeReport*, *ActivityReport* and *DocumentTimeD/I/F/R*.

7.1 User Relation

The User relation contains all registered users together with user specific information such as password and email. A user belonging to the project may be assigned a role under the role attribute.

The userName attribute is used as the primary key, making every username unique. The appearance of the table can be viewed in Figure 3.

```
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| userName | int          | NO   | PRI | NULL    |       |
| password | varchar(50)  | NO   |     | NULL    |       |
| email    | varchar(100) | YES  |     | NULL    |       |
| role     | varchar(30)  | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
```

Figure 3: A view of the User relation and its attributes.

A User table can be created using the following MySQL command:

```
mysql> CREATE TABLE User (
    -> userName Integer NOT NULL,
    -> password varChar(50) NOT NULL,
    -> email varChar(100),
    -> role varChar(30),
    -> PRIMARY KEY(userName)
    -> ON UPDATE CASCADE ON DELETE SET NULL
    -> );
```

7.2 TimeReport Relation

The time report to be filled out by each individual user on a weekly basis contains 55 columns, aiming to help the user specify the time spent on various activities. Four sections - *D (Development)*, *I (Informal review)*, *F (Formal review)*, *R (Rework, improvement or correction)*. However, there may be weeks where no time is spent on, e.g., formal reviews. In that case, a large number of columns would be wasted, as there will be no meaningful data added to the columns related to that section of the time report. The appearance of the table can be viewed in Figure 4.

```
+--------------+------+------+-----+---------+-------+
| Field        | Type | Null | Key | Default | Extra |
+--------------+------+------+-----+---------+-------+
| reportID     | int  | NO   | PRI | NULL    |       |
| userName     | int  | NO   | MUL | NULL    |       |
| totalMinutes | int  | YES  |     | NULL    |       |
| signature    | int  | YES  | MUL | NULL    |       |
| week         | int  | NO   |     | NULL    |       |
+--------------+------+------+-----+---------+-------+
```

Figure 4: A view of the TimeReport relation and its attributes.

A TimeReport table can be created using the following MySQL command:

```
mysql> CREATE TABLE TimeReport (
    -> reportID Integer NOT NULL,
    -> userName Integer NOT NULL,
    -> totalMinutes Integer,
    -> signature Integer,
    -> week Integer NOT NULL,
    -> PRIMARY KEY(reportID),
    -> FOREIGN KEY(userName) REFERENCES User(userName)
    -> ON UPDATE CASCADE ON DELETE CASCADE,
    -> FOREIGN KEY(signature) REFERENCES User(userName)
    -> ON UPDATE CASCADE ON DELETE SET NULL
    -> );
```

To avoid data redundancy, the time reports are being separated into several different relations. The reason for this is the fact that every single submitted time report is going to be represented in TimeReports.

7.3 ActivityReport Relation

ActivityReport is the relation concerning the general activity types. As can be seen in Figure 1, these include the unique primary key: reportID and other attributes such as homeReading, meeting and various tests. The appearance of the table can be viewed in Figure 5.

Field	Type	Null	Key	Default	Extra
reportID	int	NO	PRI	NULL	
totalMinutes	int	YES		NULL	
functionalTest	int	YES		NULL	
systemTest	int	YES		NULL	
regressionTest	int	YES		NULL	
meeting	int	YES		NULL	
lecture	int	YES		NULL	
exercise	int	YES		NULL	
computerExercise	int	YES		NULL	
homeReading	int	YES		NULL	
other	int	YES		NULL	

Figure 5: A view of the ActivityReport relation and its attributes.

An ActivityReport table can be created using the following MySQL command:

```
mysql> CREATE TABLE ActivityReport (
    -> reportID Integer NOT NULL,
    -> totalMinutes Integer,
    -> functionalTest Integer,
    -> systemTest Integer,
    -> regressionTest Integer,
    -> meeting Integer,
    -> lecture Integer,
    -> exercise Integer,
    -> computerExercise Integer,
    -> homeReading Integer,
    -> other Integer,
    -> PRIMARY KEY(reportID),
    -> FOREIGN KEY(reportID) REFERENCES TimeReport(reportID)
    -> ON UPDATE CASCADE ON DELETE CASCADE
    -> );
```

7.4 DocumentTimeD/I/F/R Relation

As mentioned earlier, there are four different activity types in the time documentation: *Development and documentation (D)*, *Informal review (I)*, *Formal review (F)* and *Rework, improvement or correction (R)*. All of these relations contain the same attributes mainly concerning the various documents. Besides these attributes, there is one attribute concerning the totalMinutes. The purpose of this attribute is to oversee the amount of minutes spent on each activity type. Since each document continuously is in a different stage, it was concluded to include different phases as activity types to ensure that the time reports are comprehensible. All of these tables can be created

by using the same SQL statement. The appearance of each of these tables can be viewed in Figure 6.

```
+--------------+------+------+-----+---------+-------+
| Field        | Type | Null | Key | Default | Extra |
+--------------+------+------+-----+---------+-------+
| reportID     | int  | NO   | PRI | NULL    |       |
| totalMinutes | int  | YES  |     | NULL    |       |
| SDP          | int  | YES  |     | NULL    |       |
| SRS          | int  | YES  |     | NULL    |       |
| SVVS         | int  | YES  |     | NULL    |       |
| STLDD        | int  | YES  |     | NULL    |       |
| SVVI         | int  | YES  |     | NULL    |       |
| SDDD         | int  | YES  |     | NULL    |       |
| SVVR         | int  | YES  |     | NULL    |       |
| SSD          | int  | YES  |     | NULL    |       |
| finalReport  | int  | YES  |     | NULL    |       |
+--------------+------+------+-----+---------+-------+
```

Figure 6: A view of a DocumentTime relation and its attributes.

A DocumentTime table can be created using the following MySQL command:

```
mysql> CREATE TABLE TimeDevelopments (
    -> reportID Integer NOT NULL,
    -> totalMinutes Integer,
    -> SDP Integer,
    -> SRS Integer,
    -> SVVS Integer,
    -> STLDD Integer,
    -> SVVI Integer,
    -> SDDD Integer,
    -> SVVR Integer,
    -> SSD Integer,
    -> finalReport Integer,
    -> PRIMARY KEY(reportID),
    -> FOREIGN KEY(reportID) REFERENCES TimeReport(reportID)
    -> ON UPDATE CASCADE ON DELETE CASCADE );
```

8 Description of Classes

This section describes all classes and their public methods. The classes along with their methods and attributes can be viewed in Figure 7.

8.1 Database

8.1.1 addUser()

Adds a user to the database.

8.1.2 removeUser()

Removes a user and all related time reports from the database.

8.1.3 retrieveTimereports()

Retrieves a list of all time reports from the database.

8.1.4 viewTimereport()

Retrieves a specific time report from the database.

8.1.5 signTimereports()

Changes all timereports according to user input.

8.1.6 viewUsers()

Retrieves a list of all users from the database.

8.1.7 setProjectLeader()

Changes a users role to project leader [1] in the database.

8.1.8 newTimereport()

Adds a new time report to the database.

8.1.9 updateTimereport()

Changes an existing unsigned time report according to user input.

8.1.10 deleteTimereport()

Removes an unsigned time report from the database.

8.1.11 getSummary()

Retrieves a summary of all reported time.

8.1.12 updateRole()

Changes a users role according to user input. Cannot change a users role to PG [1].

8.1.13 checkLogin()

Checks if the login credentials are correct.

8.1.14 getPassword()

Retrieves the users current password.

8.1.15 setPassword()

Checks if the new password is valid by using getPassword(). If it's valid, it replaces the old password. If not, the password remains unchanged.

8.1.16 getEmail()

Retrieves the users email.

8.2 UserBean

8.2.1 populateBean()

Sets all attributes in the bean according to user input.

8.2.2 getUserName()

Gets the username contained in the bean.

8.2.3 getEmail()

Gets the email contained in the bean.

8.2.4 getRole()

Gets the role contained in the bean.

8.2.5 setEmail()

Sets the email attribute in the bean.

8.2.6 setRole()

Sets the role attribute in the bean.

8.3 UserManagementBean

8.3.1 populateBean()

Sets all attributes in the bean according to the database.

8.3.2 getUserName()

Gets the username contained in the bean.

8.3.3 getUserList()

Gets the userlist contained in the bean.

8.3.4 getRole()

Gets the role contained in the bean.

8.4 TimeReportBean

8.4.1 populateBean()

Sets all attributes in the bean according to user input.

8.4.2 getUsername()

Gets the username contained in the bean.

8.4.3 getWeek()

Gets the week of the time report contained in the bean.

8.4.4 getReportValues()

Gets the values of the time report contained in the bean.

8.5 TimeReportManagementBean

8.5.1 populateBean()

Sets all attributes in the bean according to user input.

8.5.2 timeReportList()

Gets a list of all time reports contained in the bean.

8.5.3 getUserName()

Gets the username contained in the bean.

8.6 PasswordHandler

8.6.1 hashPassword()

Returns the hash of a given password.

8.6.2 generatePassword()

Generates a new random password according to requirement 6.2.1 in [1].

8.7 MailHandler

8.7.1 sendPasswordMail()

Sends an email containing a password generated from PasswordHandler.

9 Class Diagram

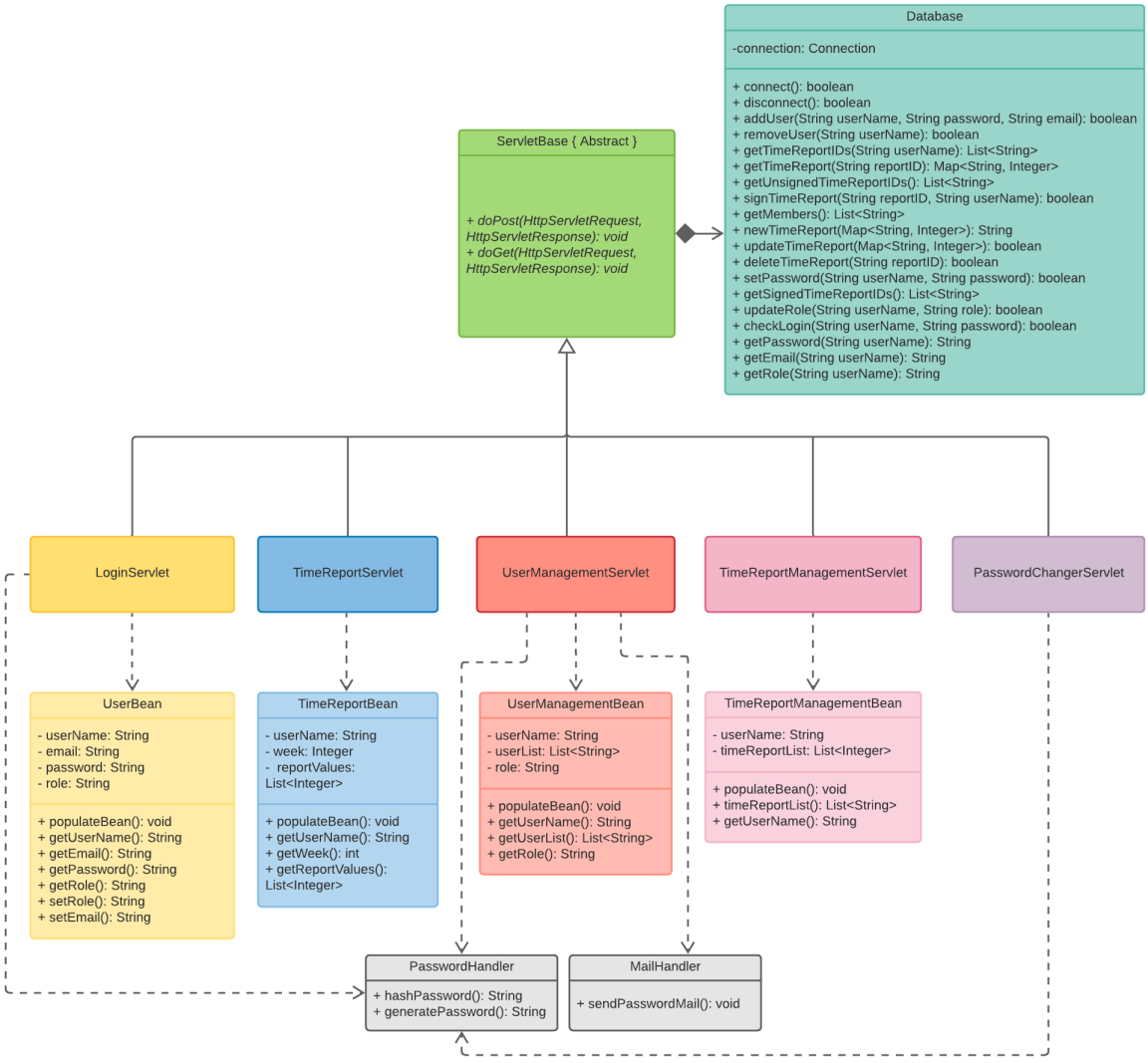

Figure 7: Diagram containing an overview of all classes.

10 Sequence Diagrams

10.1 LoginServlet

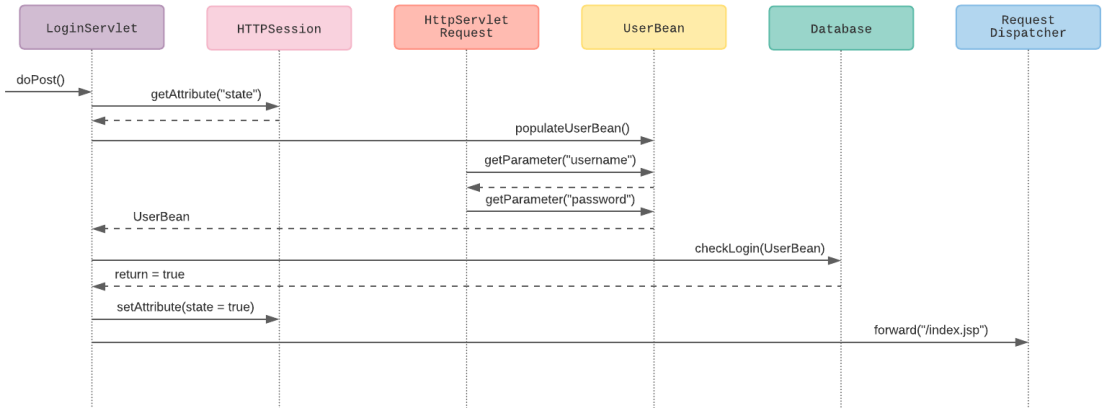

Figure 8: Sequence diagram for a successful login.

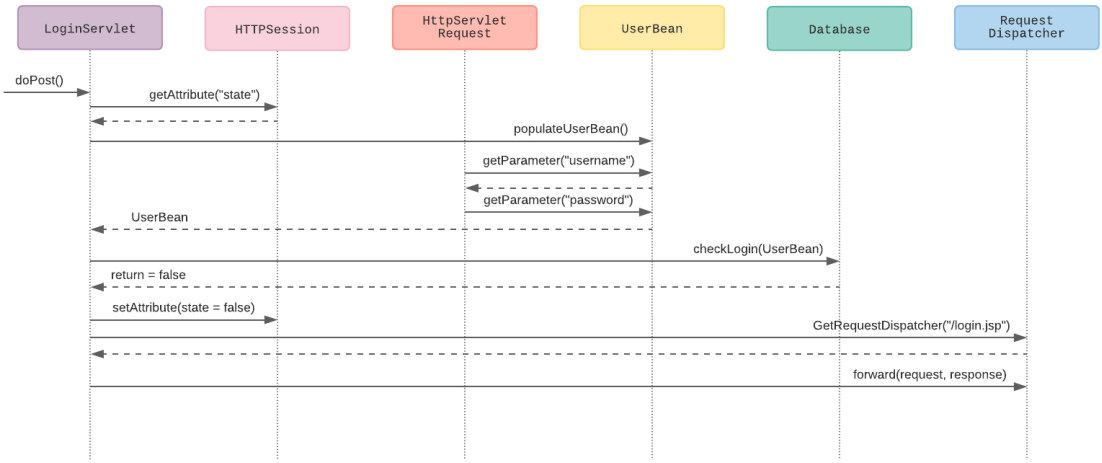

Figure 9: Sequence diagram for a failed attempt to login.

10.2 UserManagementServlet

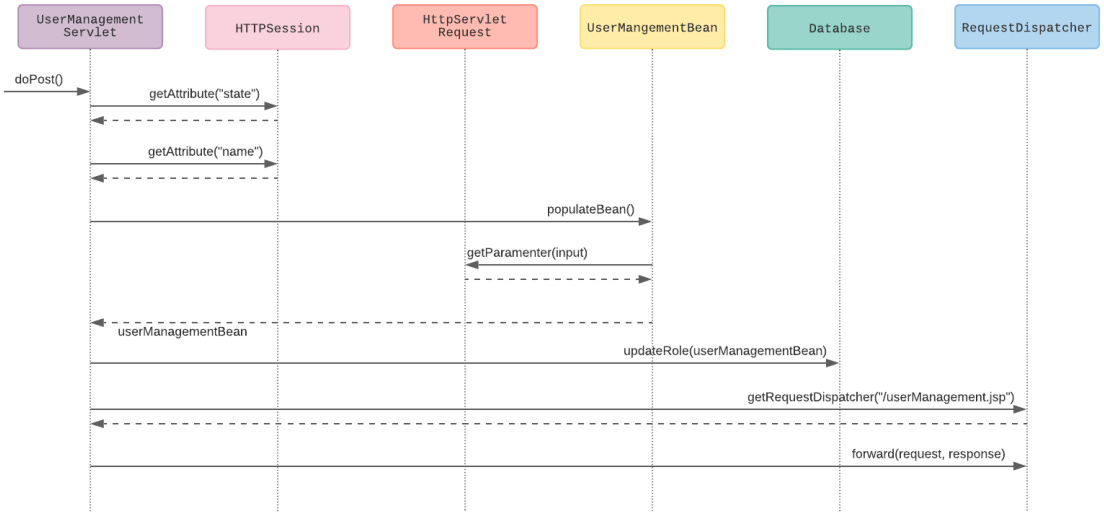

Figure 10: Sequence diagram for the process of updating the role of a project member.

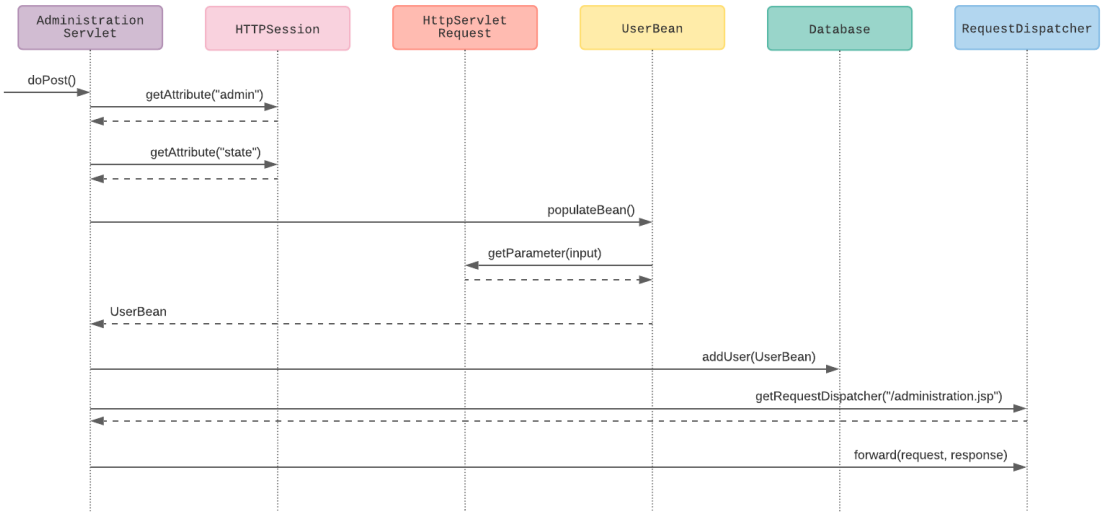

Figure 11: Sequence diagram for adding a user by administrator.

10.3 PasswordChangerServlet

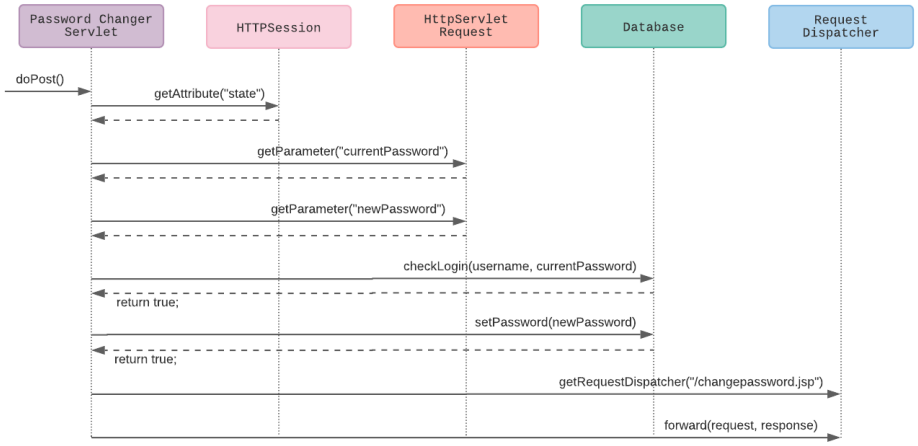

Figure 12: Sequence diagram for a successful password change.

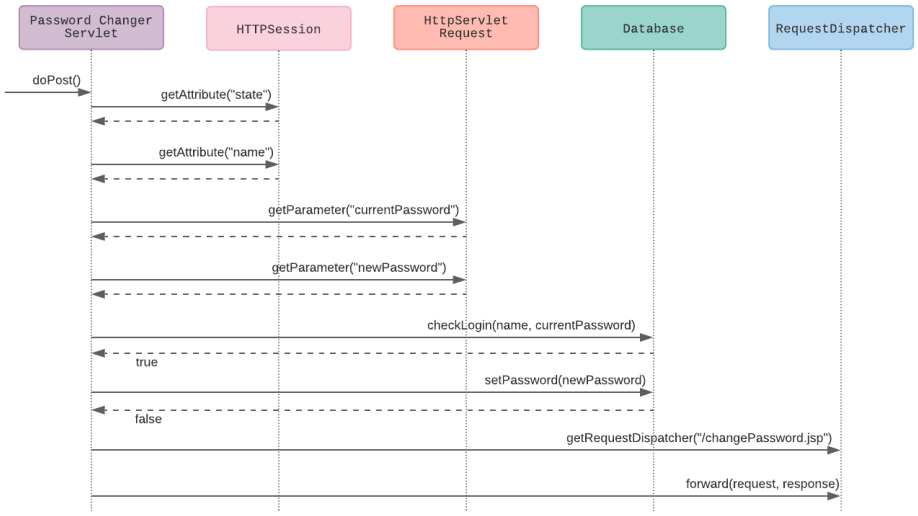

Figure 13: Sequence diagram for a failed attempt to change password.

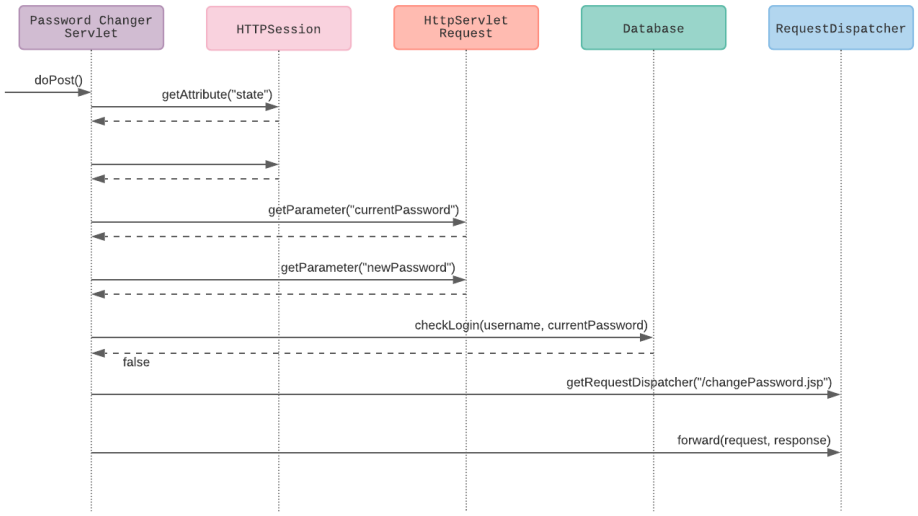

Figure 14: Sequence diagram for a failed attempt to change password due to incorrect user input.

10.4 TimeReportServlet

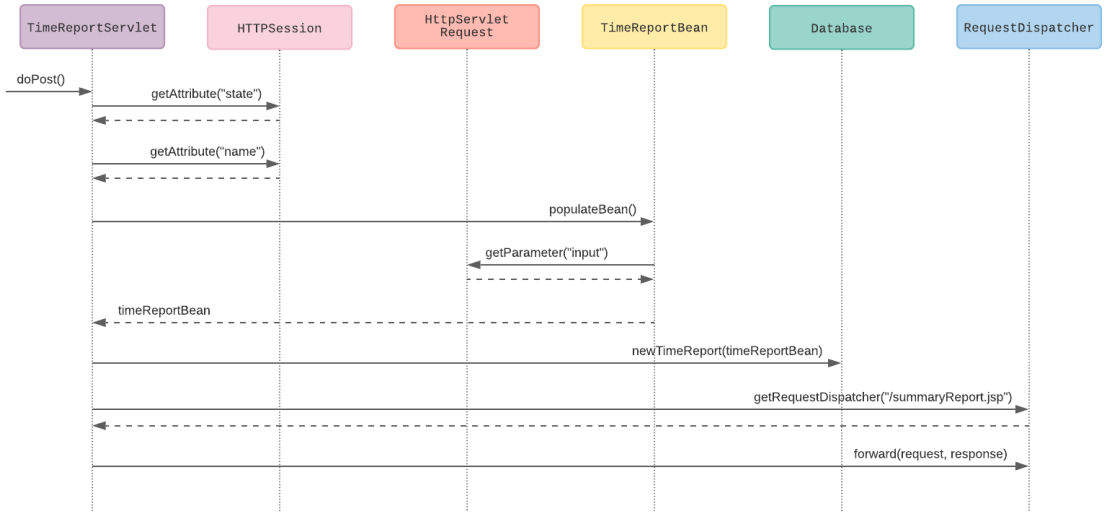

Figure 15: Sequence diagram for adding a new Time Report.

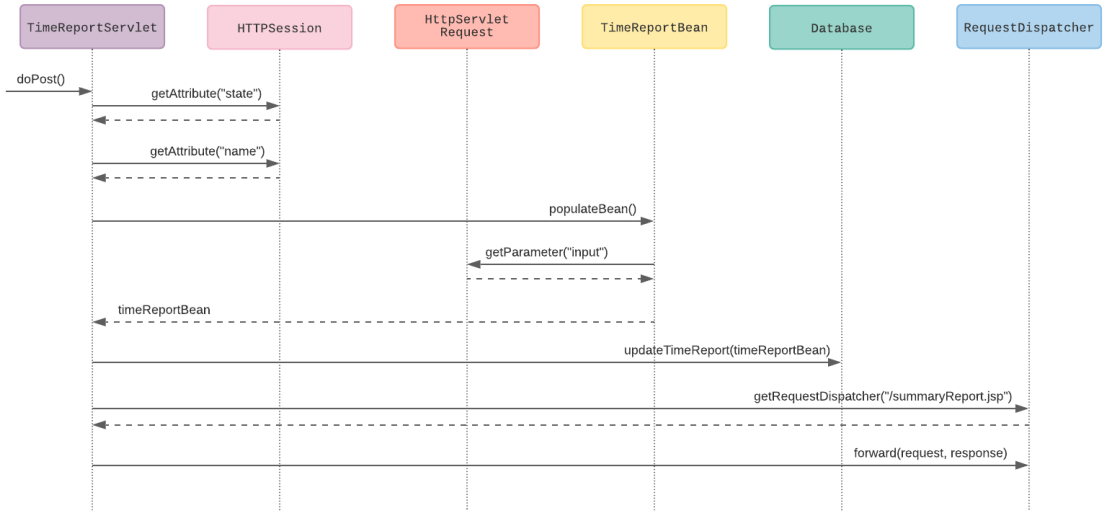

Figure 16: Sequence diagram for a editing a Time Report.

10.5 TimeReportManagementServlet

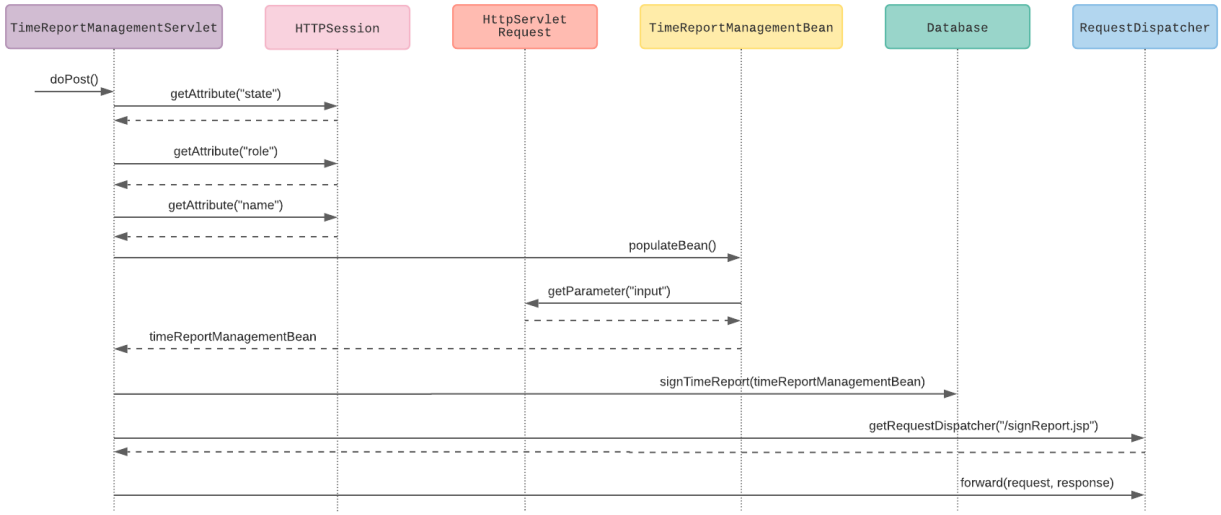

Figure 17: Signing/unsigning a Time Report. This can only be done by the project leader.