Project Report Web-applications

Albin Becevic

March 2022

https://github.com/albinbecevic/Webapplicatons

1 Introduction

In order to simplify the communication between students and teachers schools utilize the use of a so called learning management platforms. A learning management platform is a webb-based software application which provides a course environment for teacher to distribute course material to students in order to conduct learning. The goal of the project was to develop a webb-based learning management system in Java EE. The platform is supposed to be directed towards high school level education with purpose to ease the communication between students and teachers.

2 List of Use Cases

2.1 User

- A user can create an account
- A user can choose between being a student or a teacher
- A user should be able to log in to their account.

2.2 Student

- A student should be able to join a course
- A student should be able to see which courses it has joined
- A student should be able to see the course syllabus
- A student should be able to view a course calendar
- A student should be able to view assignments

2.3 Teacher

- A teacher should be able to create a course
- A teacher should be able to create a syllabus for the course.
- A teacher should be able to create assignments for the course.
- A teacher should be able to give a code to the students in order to join the course.

3 User Manual

This section will describe how to use the app from a user's point of view. Depending on what kind of account the user created, there will be different functionalities in store for the users. However, the first thing a user will be introduced to is the following login page:

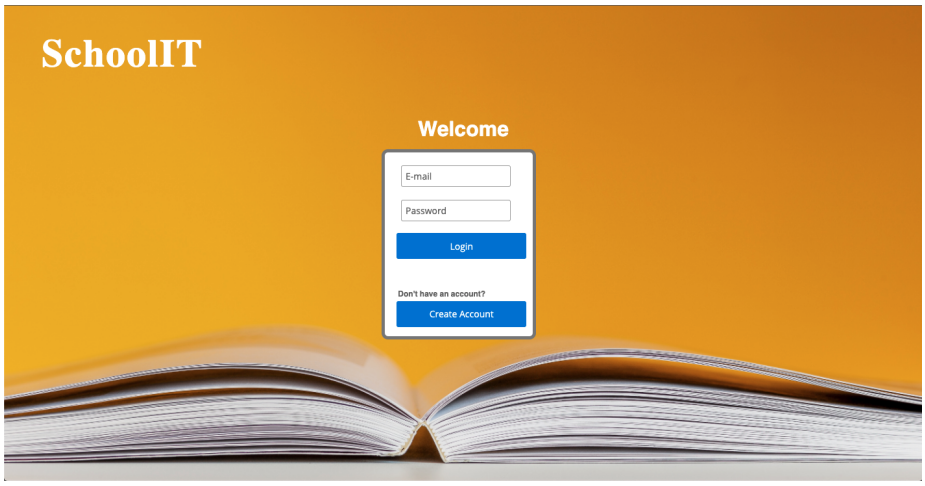

Figure 1: The main page of the web-application

3.1 Login And Creating Account

The first thing a user will be introduces to is the login page. If the user tries to log in without an account a error message will be displayed. Therefore, the user has to create an account first in order to use the application. The user can easily create an account by clicking on the "Create Account" button. After clicking the button the user will be directed to the account selection page and will have to what kind of account will be created, student or teacher account. After selecting what type of account to be created the user will then be directed to the create account page (CreateStudentAccount.xhtml or CreateTeacherAccount.xhtml). Where the user can provide the necessary information inorder to create account. After this process the user will then be directed to back to the login page where it now can login to their account.

3.2 Teacher

A teachers main role is to be able to create a course and be able to distribute the course material to its students.

3.2.1 Teacher Dashboard

By loggin on the teacher account, the teacher will first be introduced to its dashboard. The dashboard is where the teacher can view all its courses and also choose to create a new course. If the teacher decides that it wants to create a course, then it can click on the "Create Course" panel located on the dashboard. After clicking on the panel, the dashboard section will be updated to an create course section where the teacher can provide relevant information for the course.

3.2.2 Creating a Course

The create course section is divided into three parts, where the first part displays the generated course code which the teacher can give to students in order for them to join the course. The teacher can also type the name of the course. The submit button will persist the course in the database. Next section handles the syllabus for the course. Here the teacher can create a course syllabus and also edit (delete a module) if necessary. The teacher selects date, lecture description, recommended reading and also recommended exercises. After submitting a module it will be displayed in the data table below. If the teacher wants to create an (bigger) assignment it can do so in the next section. In the "create assignment section the teacher can type information about the assignment and instructions. After submitting the assignment it will be displayed in a data table below, where the teacher can choose to delete if necessary. After this process the teacher can go back to its dashboard and the dashboard will be updated with a new panel displayed on the dashboard with the newly created course name.

3.2.3 Teacher Course Page

If the teacher wants to inspect the course, it can do so by clicking on the panel on the dashboard with the desired course name. By doing so, the dashboard will then update the view to display the course information such as, course code and name, syllabus and the created assignments. In this page the teacher can copy the course code and give it to students. Also if the teacher wish to add or remove/edit assignments it can do so in the course page as well.

3.3 Student

The role of the student is to be able to join courses and view the information provided by the teacher.

3.3.1 Student Dashboard

After the user logs in as a student it will be introduced to a similar dashboard as the teacher with a slight difference. Instead of a "create course" panel there is a "Join Course" panel. Clicking on the panel will create a pop-up window which asks for a course code. If the provided course code from the student exists, then

the dashboard will be updated and a new panel with the course name will be shown.

3.3.2 Student Course Page

If the student clicks on a course panel it will be redirected to the course page for students. The student will be introduced to a tab-menu with the different menu items "Home", "Calendar", "Syllabus", "Assignments" and "System". Currently, the only menu items with a functionality is the "Calendar", "Syllabus" and "Assignments". If the student clicks on syllabus then the view will be updated to a data table that shows the course syllabus. If the student wish to see the modules in the data table in a calendar format. The student can click on the "Calendar" menu item and the view will be updated to a calendar that shows the modules in the syllabus in a calendar. Lastly, the student can click on the "Assignment" menu item and it will display the assignments created by the teacher in a data table.

4 Design

This section will explain the different technologies, Libraries and frameworks used in the application.

4.1 Technologies for Environment

Netbeans was used as IDE for the development of the application. Apache Derby was utilized as a database for the application. The Java EE application server used for the application was Payara server.

4.2 Libraries and Frameworks

This section will introduce the different libraries and frameworks used in the application, also a small justification as to why.

4.2.1 JSF - Java Server Faces

JSF was used as the "frontend" framework. Reason for that is that JSF comes with some perks. JSF is built on top of the Java Servlet API which means that you no longer have to spend time on implementing servlets, JSF will handle it for you. It also does alot of things for you such as validate data and handle the components on the page in its life cycle.

4.2.2 PrimeFaces

Together with JSF the component library PrimeFaces was used in order to use more complex components in the application.

4.2.3 Querydsl

Inorder to write more simple, and familiar sql queries the library Querydsl was used instead of the standard Criteria API in Java EE.

4.3 Java EE Technologies

This section will introduce the different Java EE technologies used within the application.

4.3.1 JPA - Java Persistence API

To structure the database and store data JPA was used. JPA applies ORM technology in order to store data between incompatible systems. Java objects can't directly be stored in a relational database as Apache Derby, with ORM this is no longer a problem.

4.3.2 EJB Technology

The Java EE EJB technology was used in order to specify Enterprise Java Beans so that the container could handle them properly. In the application it was mostly the DAOs (in the persistence layer) that was annotated with the EJB annotation.

4.3.3 Java Security API

For secure login and authentication of the user the Java Security API was used. Through the security context the correct authenticated user could be retrieved within the application.

4.3.4 Context and Dependency Injection

Context and Dependency injection was used in order to instantiated objects within the application.

5 Application flow

This section will describe a simple use case within the application together with a UML sequence diagram provided at the bottom. The use case in consideration is when a user wants to create a teacher account.

When a user arrives to the web-app it's first introduces to a login page. If the user tries to login without typing anything in the input fields or having created an account JSF will invoke an error message. When the user decides that it wants to create an account and click on the "Create Account" button, JSF will redirect the user to a so called "SelectTypePage". The SelectTypePage only consists of two panels with three buttons, "Back", "Teacher" or "Student".

Depending on what the user selects, JSF will redirect the user to the proper page. When the user click on the "Teacher" button JSF updates the view to the "CreateTeacherPage". Here the user can now provide the necessary information in the input fields in order to create an Teacher Account. When the fields are filled and the user click on the "Create Account" button a request is then sent to JSF on the server, which performs its JSF lifecycle and invokes the application, calling the CreateTeacherBackingBean, the backing bean. If the account doesn't already exist then the backing bean calls its Data Access Object TeacherDAO to persist the newly created Teacher object.

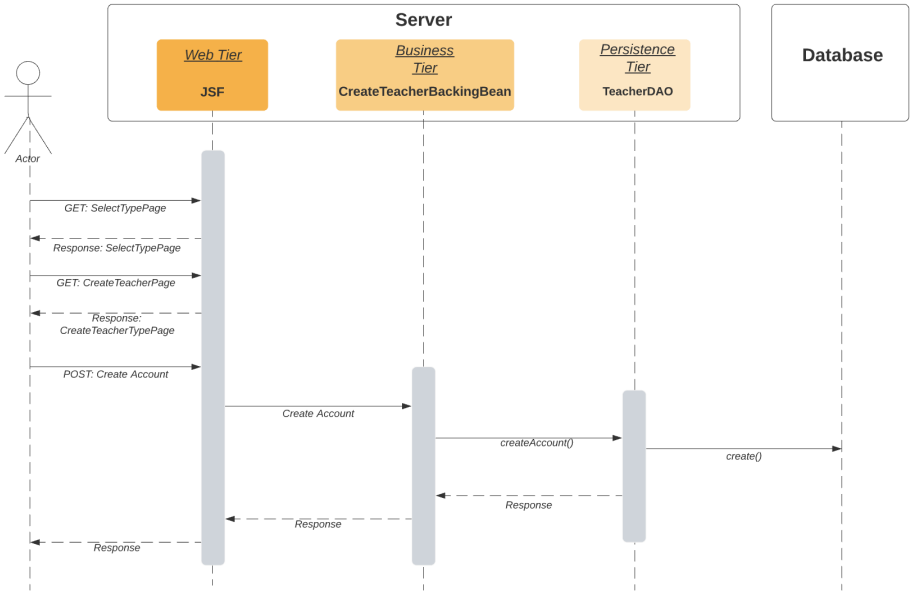

Figure 2: The UML sequence diagram for creating a teacher account

6 Package Diagram

The following picture shows the package diagram of the layered application.

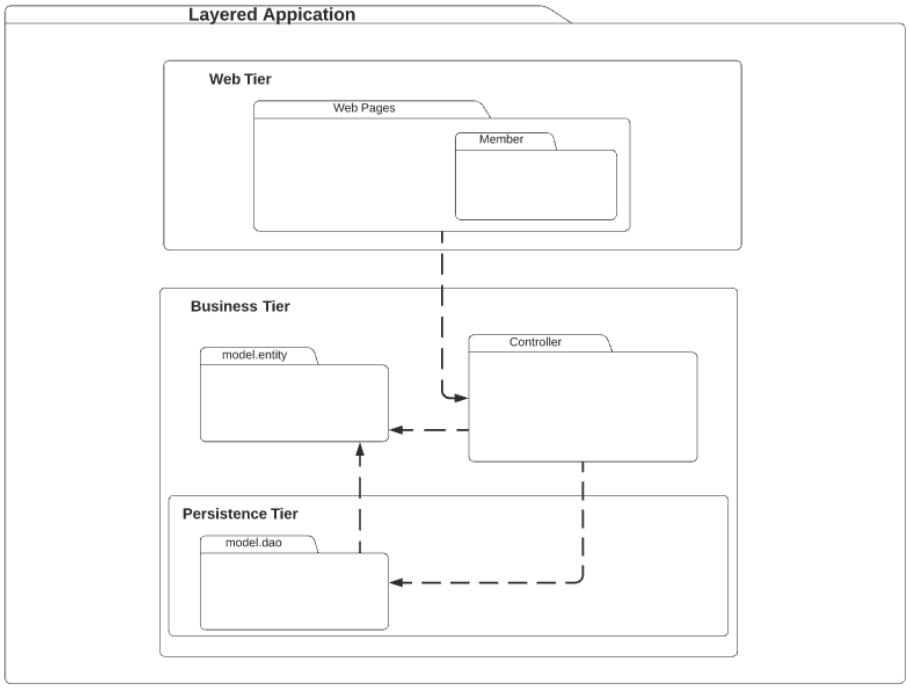

Figure 3: The overall package diagram of the Multilayered web-application

7 Model Diagram

The following picture show the UML diagram of the model.

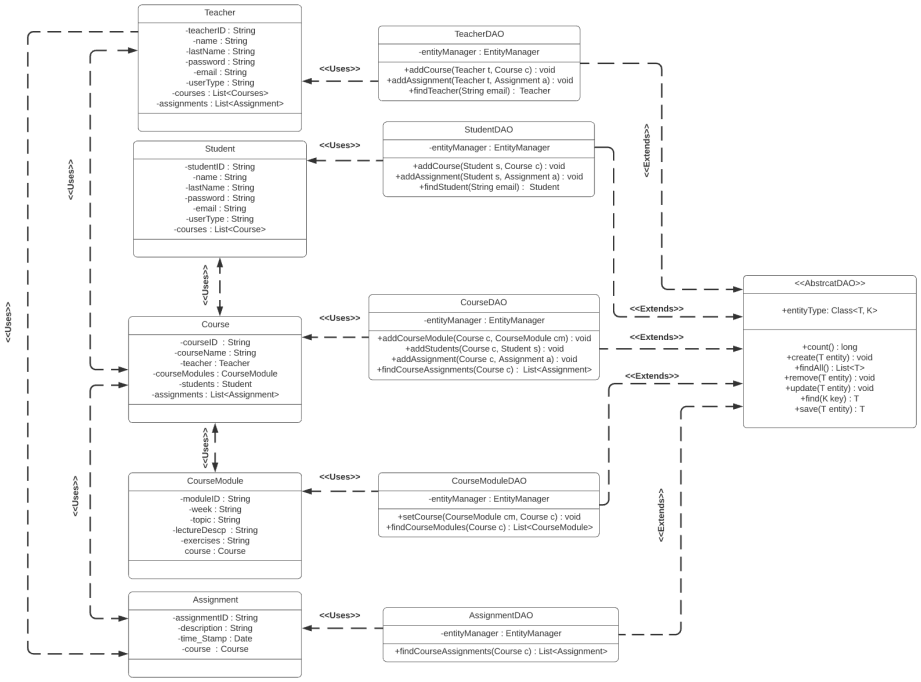

Figure 4: UML diagram of the model.

8 Responsibilities

Albin Becevic was responsible for the entire application.