EE417 Group Assignment - Documentation

Group E

April 14, 2024

Introduction

This documentation provides a comprehensive guide to the development and deployment of the DCU Smart Dashboard, a dynamic platform designed to streamline the management and accessibility of university rooms. Within these pages, readers will find detailed information spanning from the front-end development, to back-end architecture, deployment and testing.

The sections are organized to walk you through the website's architecture, both front-end and back-end development practices, database integration, cloud deployment strategies, and rigorous testing procedures that ensure robust performance. Additionally, insights into the team's structure and individual responsibilities will be shared, highlighting the collaborative efforts that contributed to the project's success.

How to run:

As is the project has not undergone cloud deployment, so the version to be corrected is local only. Below are the steps to run the project and have it accessible at http://localhost:8080/

- Import project folder into the Eclipse IDE
- Right click on the pom.xml file, and select run as -¿ maven build to build target folder
- When prompted, enter 'package' into goals and apply
- Run `springJwt-main\src\main\java\com\GroupE\Assignment4\SpringJwtApplication.java`
- Local website is now running
- For full access, login with username 'brice13' and password 'password'

Website Overview

The RoomSense platform offers a comprehensive suite of tools for the DCU community, streamlining various aspects of university life into a cohesive dashboard experience. It features a robust interface for room management, course organisation, calendar, and user account handling. The modular design emphasizes user-friendly interactions with a focus on intuitive navigation and real-time updates.

Front-end Development

User Interface

The front-end of the RoomSense platform is crafted with a modern aesthetic, delivering a responsive design that adapts seamlessly across devices. Utilizing HTML, CSS, and JavaScript, it integrates interactive elements that provide immediate feedback to user actions. The design employs a consistent dark theme that enhances readability and reduces eye strain in various lighting conditions.

Authentication Flow

Through `login.html` and `register.html` within the `/connect` directory, the system facilitates secure user authentication. Employing local storage, JWT tokens ensure secure sessions across the platform, maintaining a user's logged-in state across different pages.

Dynamic Content Rendering

JavaScript is employed to fetch and display data dynamically. This is evident in the room selection feature, where users can choose a room from the sidebar, and relevant details are immediately presented using visually engaging gauges for metrics like temperature and occupancy.

Navigation

The navigation is designed to offer direct access to all major sections of the dashboard from the static sidebar, with links to `home`, `courses`, `calendar`, and `rooms`, fostering an environment of efficient user flow.

Page Implementations

Home Page

The `home.html` serves as the landing page, providing a welcoming interface and a brief overview of the dashboard's capabilities. It features dynamic content such as current campus news or user-specific academic information.

Room Management Pages

- **Add Room (`addRoom.html`):** Allows administrators to add new rooms to the system. This form includes fields for room characteristics such as size, temperature, and occupancy.
- **Edit Room (`editRoom.html`):** Used by administrators to update room details. Similar to the add room page, but pre-filled with existing data.

User and Authentication Pages

- **Login Page (`login.html`):** Provides user login functionality with JWT token handling for session management.
- **Registration Page (`register.html`):** Allows new users to register, feeding their information into the system for authentication and management.

Academic and Event Management

- **Calendar (`calendar.html`):** Displays academic and event calendars, offering users the ability to track important dates and events.
- **Courses Page (`courses.html`):** Lists available courses and includes detailed descriptions, room assignments, and scheduling information.

Administration and Settings

- **User Management (`users.html`):** An interface for administrators to manage user accounts, including roles and permissions.
- **Room Settings (`rooms.html`):** Provides a detailed view of all rooms and their current status, with options to edit as needed.

Style Consistency

All pages follow a unified style guide defined in `style.css`, which ensures a consistent look and feel throughout the platform. This includes a dark color scheme, responsive design elements, and interactive components like buttons and links that enhance user engagement.

Back-end Logic

User

Creation of Get and Post request in the `UserController`.
Get request to access `/admin/users` page. It shows the list of user in the database and allow to modify the information and apply the modification.
Also, allow to delete a user from the database.
All of those interactions are locked behind an authentication, meaning it needs the `jwt` token to be accessible.

Home

A `HomeController` allows to redirect to `home.html` (a static page) upon connection to `localhost:8080`.

Rooms

Creation of Get and Post request in the `RoomController`.
Possibility to create a new room from `/admin/addRoom` and to edit a room using `/admin/editRoom/{id}`.
Ability to see every rooms information from the `/rooms` page. Post method allow to dynamically get a specific room data and is rendered.
The page is dynamically changing depending on the level of authorization a user has to properly show the `addRoom` and `editRoom` button.

Database Connectivity

Implementation of AWS DB for the project involves configuring the `application.yml` file with the database credentials. This setup ensures that the application can interact with a remote MySQL database hosted on AWS. The database uses the update strategy for Hibernate's DDL auto configuration to avoid dropping and recreating tables on every application restart, facilitating a persistent data management environment.

Repository and Service Layers

The system architecture includes a robust repository layer, where each entity has its dedicated interface that extends Spring Data JPA. This layer abstracts the database interactions and provides an elegant way to perform CRUD operations. The service layer implements business logic and database transaction management, ensuring data consistency and integrity throughout the application.

Security and JWT Integration

The security configuration within the application is robust, utilizing JWT for authentication and authorization. This mechanism ensures that only authenticated and authorized users can access specific endpoints and functionalities within the application, especially in the `/admin` section.

Error Handling

The application employs a centralized error handling mechanism to manage and respond to exceptions uniformly. An `ErrorController` and `ErrorHandler` provide user-friendly error responses that help in diagnosing issues during operation.

Testing

Throughout the development lifecycle of RoomSense, rigorous testing was conducted to ensure the robustness and reliability of the application. Frontend components were meticulously tested for cross-browser compatibility and responsiveness, ensuring a consistent user experience across various devices. Using a combination of unit tests and integration tests, the backend services were validated against a suite of test cases that simulated real-world usage scenarios. Security tests, particularly for the JWT-based authentication system, were performed to safeguard against common web vulnerabilities. The collective testing efforts contributed significantly to the stability and security of RoomSense.

Team Roles and Responsibilities

All team members are responsible for:

- Documentation for their specific area
- Attendance in group meetings

Patriks

- Project and team management + GitHub repository management
- Front-end JS integration with backend - testing endpoints.
- Data visualization using Plotly.js
- Compiled groups documentation into this report

Adam

- Front-end design and content on static pages ,calendar.html,
- Backup alternative login.html (not used)

Brice

- Server-side design and logic initially - setup all controllers etc
- Database design and integration

Adrien

- Back-end development/working with frontend, fullstack

- Database testing and validation

David

- No evidence of contribution through GitHub commits

Theodore

- No evidence of contribution through GitHub commits

Yuan

- No evidence of contribution through GitHub commits

Kaushal

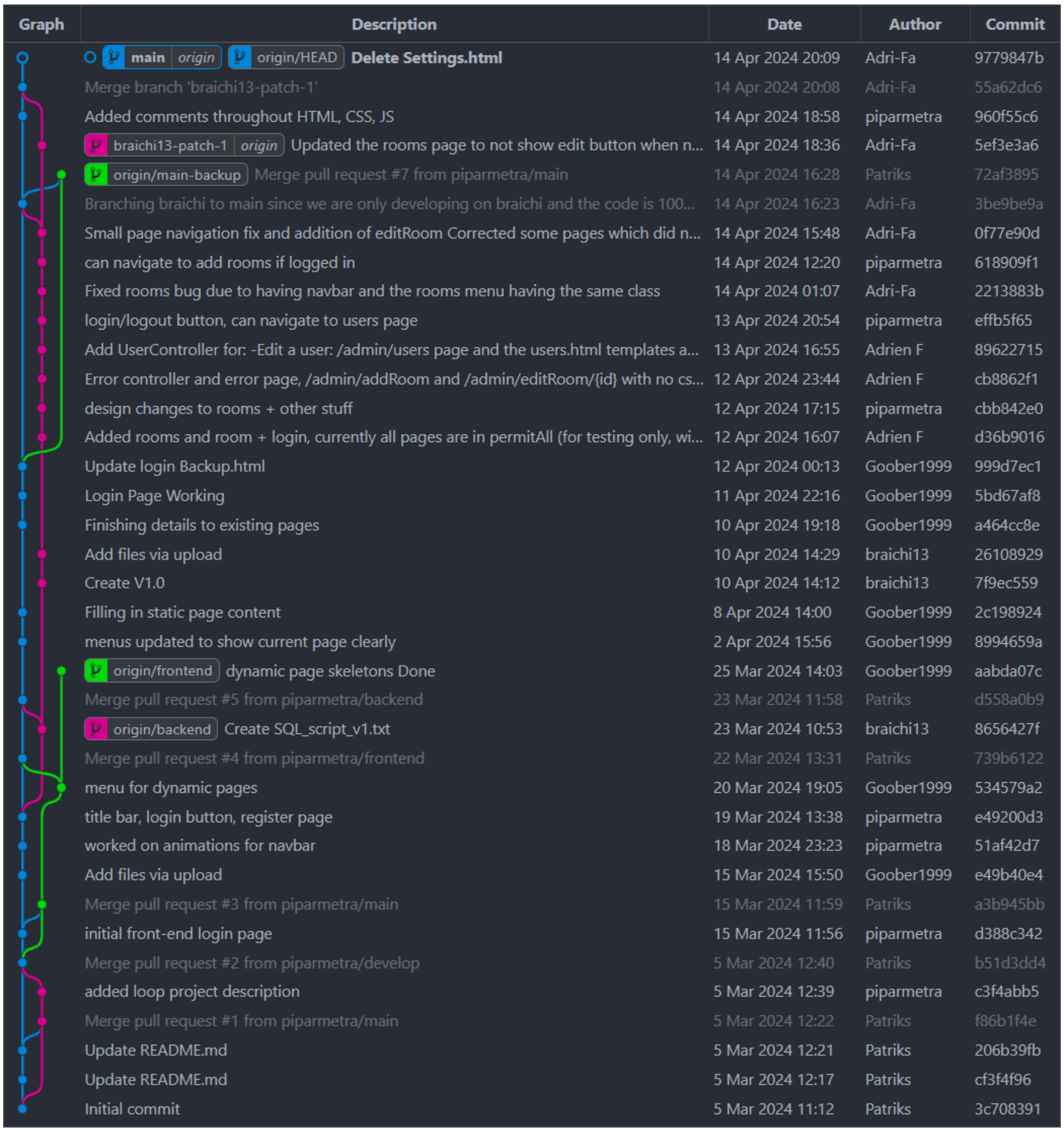
- No evidence of contribution through GitHub commits

Git graph

We used Git for version control and collaboration. To visualize the commit history and branching structure of our repository, we generated a Git graph using the git graph extension from Visual Studio Code. This graph shows the evolution of the branches throughout this project. It also shows each commit with the beginning of the description and their author.

Also, below is a list that relates GitHub usernames to real names of project members:

- Patriks - GitHub: piparmetra (also Patriks)
- Adrien - GitHub: Adri-Fa (also Adrien F)
- Adam - GitHub: Goober1999
- Brice - GitHub: braichi13



Graph	Description	Date	Author	Commit
	main origin origin/HEAD **Delete Settings.html**	14 Apr 2024 20:09	Adri-Fa	9779847b
	Merge branch 'braichi13-patch-1'	14 Apr 2024 20:08	Adri-Fa	55a62dc6
	Added comments throughout HTML, CSS, JS	14 Apr 2024 18:58	piparmetra	960f55c6
	braichi13-patch-1 origin Updated the rooms page to not show edit button when n...	14 Apr 2024 18:36	Adri-Fa	5ef3e3a6
	origin/main-backup Merge pull request #7 from piparmetra/main	14 Apr 2024 16:28	Patriks	72af3895
	Branching braichi to main since we are only developing on braichi and the code is 100...	14 Apr 2024 16:23	Adri-Fa	3be9be9a
	Small page navigation fix and addition of editRoom Corrected some pages which did n...	14 Apr 2024 15:48	Adri-Fa	0f77e90d
	can navigate to add rooms if logged in	14 Apr 2024 12:20	piparmetra	618909f1
	Fixed rooms bug due to having navbar and the rooms menu having the same class	14 Apr 2024 01:07	Adri-Fa	2213883b
	login/logout button, can navigate to users page	13 Apr 2024 20:54	piparmetra	effb5f65
	Add UserController for: -Edit a user: /admin/users page and the users.html templates a...	13 Apr 2024 16:55	Adrien F	89622715
	Error controller and error page, /admin/addRoom and /admin/editRoom/{id} with no cs...	12 Apr 2024 23:44	Adrien F	cb8862f1
	design changes to rooms + other stuff	12 Apr 2024 17:15	piparmetra	cbb842e0
	Added rooms and room + login, currently all pages are in permitAll (for testing only, wi...	12 Apr 2024 16:07	Adrien F	d36b9016
	Update login Backup.html	12 Apr 2024 00:13	Goober1999	999d7ec1
	Login Page Working	11 Apr 2024 22:16	Goober1999	5bd67af8
	Finishing details to existing pages	10 Apr 2024 19:18	Goober1999	a464cc8e
	Add files via upload	10 Apr 2024 14:29	braichi13	26108929
	Create V1.0	10 Apr 2024 14:12	braichi13	7f9ec559
	Filling in static page content	8 Apr 2024 14:00	Goober1999	2c198924
	menus updated to show current page clearly	2 Apr 2024 15:56	Goober1999	8994659a
	origin/frontend dynamic page skeletons Done	25 Mar 2024 14:03	Goober1999	aabda07c
	Merge pull request #5 from piparmetra/backend	23 Mar 2024 11:58	Patriks	d558a0b9
	origin/backend Create SQL_script_v1.txt	23 Mar 2024 10:53	braichi13	8656427f
	Merge pull request #4 from piparmetra/frontend	22 Mar 2024 13:31	Patriks	739b6122
	menu for dynamic pages	20 Mar 2024 19:05	Goober1999	534579a2
	title bar, login button, register page	19 Mar 2024 13:38	piparmetra	e49200d3
	worked on animations for navbar	18 Mar 2024 23:23	piparmetra	51af42d7
	Add files via upload	15 Mar 2024 15:50	Goober1999	e49b40e4
	Merge pull request #3 from piparmetra/main	15 Mar 2024 11:59	Patriks	a3b945bb
	initial front-end login page	15 Mar 2024 11:56	piparmetra	d388c342
	Merge pull request #2 from piparmetra/develop	5 Mar 2024 12:40	Patriks	b51d3dd4
	added loop project description	5 Mar 2024 12:39	piparmetra	c3f4abb5
	Merge pull request #1 from piparmetra/main	5 Mar 2024 12:22	Patriks	f86b1f4e
	Update README.md	5 Mar 2024 12:21	Patriks	206b39fb
	Update README.md	5 Mar 2024 12:17	Patriks	cf3f4f96
	Initial commit	5 Mar 2024 11:12	Patriks	3c708391

Figure 1: Git graph of the finished project with all commits and authors.