Final Report: Rate My Teammate (ver.CWRU)

Haihan Jiang, Liyuan Huang, Nora Tang, Walter Nam *

November 2021

Abstract

Rate My Teammate (version CWRU) is a web-application for Case Western Reserve University students who wish to find partners for course projects or competitions. We are aware CWRU students who want to collaborate on projects or competitions tend to find collaborative teammates who have expected skills or capability. We also conducted thorough competitor analysis on strengths and weaknesses of similar websites to facilitate our design procedures. A popular website among college students called "Rate My Professor" helps students pick the classes with the best professors based on user reviews. It contains a user-friendly interface, but does not easily filter departments by specific criteria. Therefore, the main functions of this project focus on an intuitive user interface, rating and reviewing teammates, and displaying necessary information of the student while preserving their privacy. Thus, the major functionalities include Signup, Login, Search by CaseID/Name/Majors which serves as a filter, Add New Students, Add New Reviews, Submit Review Corrections, Display Searching Results, and Display Rating Summary. To encourage data heterogeneity of the website, some constraints, such as students can only add new reviews if they already logged in or signed up, are enforced. The project is developed with SpringBoot as back-end, React.js as front-end, MySQL as database management. When fully developed, CWRU students will be able to use their CaseIDs to register and utilize the above mentioned functionalities.

*All are from Department of Computer Science.

1 Introduction

Colleges offer many project-based courses. Some of these projects need multiple students to collaborate on a project. Therefore, some students may encounter partners who are not suitable for their collaborations. For example, they may not have a matched tech stack or they do not have enough time for collaborations. These factors may all contribute to an unsuccessful collaboration. To address this problem, we want to build a website application that can help Case Western Reserve University students to filter information on other students with clear rating criterias. We analyze other competitors in the market, RateMyProfessor and RateMyDorm, to learn lessons about the design. In the later sections of this report, we will show our competitor analysis and functionalities to show the strength of our product.

2 Competitor Analysis

Based on the user's needs, we can see that students who want to collaborate on projects or competitions tend to hope to find their teammates effectively and efficiently. Therefore, we try to find three functionalities(Information Filter, Rating & Review, Information's Display) in our competitors: Rate My Professor and Rate My Dorm to analyze their weakness and strengths for concluding our core functionalities.

2.1 Main Function1: Information Filter

1. **Rate My Professor:**

   Strength:

   - The search bar is placed at a center place, and users can use the search function, which is a core function for Rate my Professor.
   - It can filter by two key features: School and Professors, or combine two key features together.

   Weakness:

   - It cannot filter with courses (students may want to compare the rating between different professors who taught the same course).

- It cannot directly choose from students who take the same course.

2. **Rate My Dorm:**

   Strength:

   - It has a clear and distinctive feature filter for quick locating information.
   - Its sorting function enables users access to targeted objects in a short time.

   Weakness:

   - There is no clear tag in reviews, so users will find it hard to extract information from reviews.
   - Types of Filters are very general and not very intuitive for users.

2.2 Main Function 2: Rating and Review

1. **Rate My Professor:**

   Strength:

   - It has appropriate rating criteria for users to clearly see the instructor's capability of teaching.
   - It has specific review tags for users to efficiently select the instructor's teaching style.

   Weakness:

   - There is no way to verify the user, many fake accounts have been made to generate fake information about professors.

2. **Rate My Dorm:**

   Strength:

   - It has several essential rating (bedroom, bathroom,...) breakdown.
   - It displays useful reviewer info.

   Weakness:

   - It does not have review guidelines.

2.3 Main Function3: Information's Display

1. **Rate My Professor:**

   Strength:

   - It has clear yes/no for essential questions.
   - It has clear and useful tag display (like top tags).

   Weakness:

   - It cannot update with the student's improvement

2. **Rate My Dorm:**
   Strength:

   - It has clear class breakdown.

   Weakness:

   - It does not include some important information (such as location, price).

3 Design Goal

The design goal of our project is shown in Table 3.

4 Implementation

4.1 Backend

4.1.1 Database

In database implementation, we planned to store user data in a server. The database should contain 8 entities and 7 relationships among them. In our design, we implemented on-to-one and one-to-many relationships based on our design goals. The ER diagram of database is designed as as Figure 4.1.3.

Objective	Success Criteria	Status
Signup	The user should be able to sign up by inputting required information.	complete
Login	The existent user should be able to login with the correct caseID and password.	complete
Add new student	The logged-in user should be able to see and click on the add new student button at the bottom of the search result page. The user then should be able to add new student by inputting required information. The user who does not login should not be able to see the button and should not be able to access the page.	complete
Add new review	The logged-in user should be able to see and click on the add new review button at the bottom of the student page. The user then should be able to add new review by inputting required information. The user could review the same student only once. The user who does not login should not be able to see the button and should not be able to access the page.	complete
Submit correction	The logged-in user should be able to see and click on the submit correction button at the bottom left of each review shown on the student page. The submit correction page should be able to display information of the previous review. The user then should be able to submit correction by inputting required information. The user could submit correction only for his/her own review. The user who does not login should not be able to see the button and should not be able to access the page.	complete
Search main page	The user should be able to input the text in the search bar. And the user should be able to interact with the search button.	complete
Search result	The user should be able to see the search result of input information of the search bar.	complete
Student page	The user should be able to see the reviews of certain student.	complete
Navigation Bar	The user who does not login should be able to see the signup/login button on the top right of each page. The logged-in user should be able to see the logout button on the top right of each page. The user should be able to see the rate my teammate icon on the top left of each page except the main search page.	complete

Table 1: The design goal and success criteria of the project.

4.1.2 Data Manipulation

For data manipulation, we used maven to insert depenencies we need. In this project, we used JDBC river,Hibernate, and Spring Data JPA. After all setups are done, wee are able to build entites classes an encapsulate them with clear annotations, which are used to map between database and our maven project. Then we created JPA repositories to use some inbuilt methods to implement CRUD(create, read, update, and delete) operations. Also, we added some JPQL queries mapping to SQL queries in our database. In this way, we could manipulate our data easily.

4.1.3 Controllers

For controllers, we implement logic code in different controllers for logic control. We called different Service instances to implement our functionalities: they include Sign up, Sign in, Update Review, Update Profile, Review Correction, and Profile Correction. At the same time, we provided RESTful API to collaborate with front-end subteam.

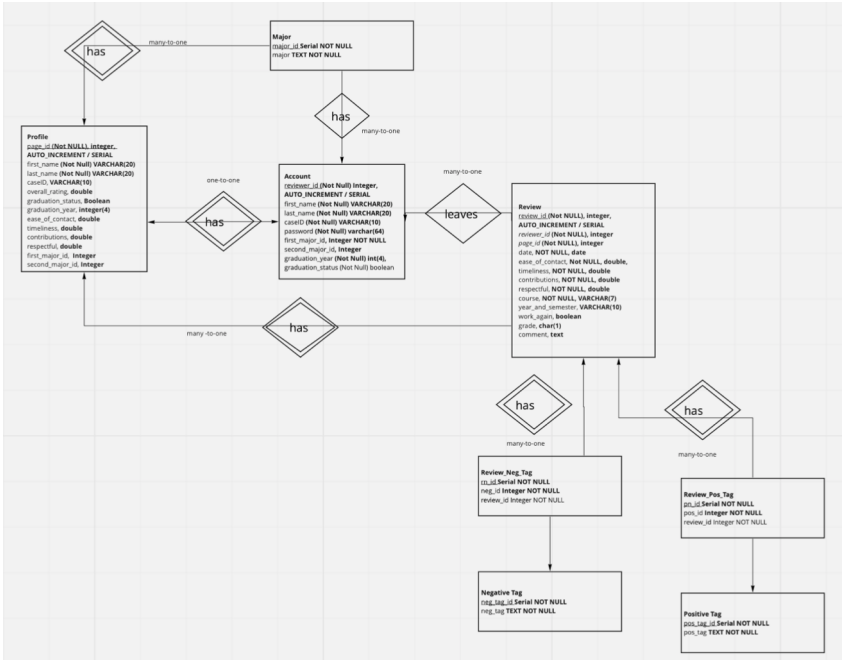

Figure 1: The ER-diagram design of the database.

4.2 Front-end

We noticed that some pages share the same components. For example, signup page and add new student page both have first name, last name, caseID, and graduation year text fields. To facilitate our design, we first broke down the components of each page and the map is shown in Figure 4.2. Then the

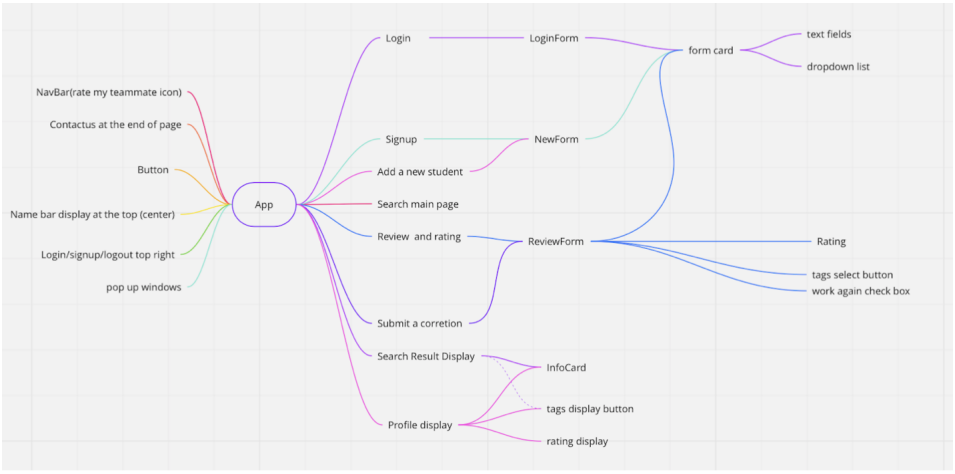

Figure 2: The component breakdown of the front-end.

front-end is built upon the above map as shown in Figure 4.2.

In the *auth-store* folder, we stored the status of the user (login or logout) because for some pages such as add new student, and add new review, only logged-in users are able to utilize these functionalities. The status will also be automatically reset, i.e., users will be automatically logged out, after one hour.

In the *Components* folder, we have several element folders for the use of different pages. *FormElement* contains necessary components of our form pages, including caseID, password, majors, etc. *PageElement* contains elements designed for main page and navbar including search bar, search button, etc. *RatingElement* contains essential parts of rating forms including rating of several metrics, comment text field, tags, etc. *StudentElement* includes components of profile page, including add new review button, reviews of the student, etc.

Here, we do not intend to go through the detailed design of each page, but we would like to present two examples to provide an overview in the

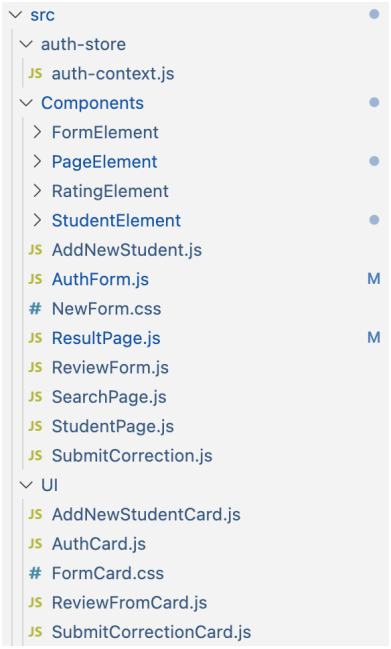

Figure 3: The structure of the front-end.

following parts [1].

4.2.1 AuthForm

Rather than designing two different pages for login and signup, it would be much more efficient to combine the two forms into one by storing the user's certain action as a trigger of the switch. As shown in Figure 4.2.1, *isLogin* will store the trigger of the user, if it's false, then the page will display necessary text fields for signup, otherwise it will only display caseID and password.

```
{!isLogin ? <GraudationYear onSaveGradYearData={saveGradYearDataHandler} />
{!isLogin ? <Majors onSaveMajorData={saveFirstMajorDataHandler} /> : ''}
{!isLogin ? <Majors onSaveMajorData={saveSecondMajorDataHandler} />: ''}
```

Figure 4: Code snapshot of the authorization form.

[1]For more details, please refer to 9.1

4.2.2 ReviewForm

One of the most important functionality of the review form is to provide users selection of tags. Figure 4.2.2 shows the negative tag rendering in the review form. Users could select and de-select the tags, and the result of interaction will be recorded through *saveNegTagDataHandler* and *removeNegTagDataHandler*. The former one stores the IDs of selected tags as a list. The tricky part happens when users try to de-select some tags. Whenever such action occurs, *removeNegTagDataHandler* will first detect the ID of the de-selected tag, and then filter the saved tag ID list to renew and store the latest tags as shown in Figure 4.2.2.

```
{negTagData && negTagData.map((negTagData) => (
    <NegTag
        onRemoveNegTagData={removeNegTagDataHandler}
        onSaveNegTagData={saveNegTagDataHandler}
        value={negTagData.id}
    >
        {negTagData.negativeTag}</NegTag>
))}
```

Figure 5: Code snapshot of the review form regarding the tag rendering.

```
const removeNegTagDataHandler = (selectedNegTag) => {
    const tagsCopy = [...selectedNegTagList];
    var filtered = tagsCopy.filter(function (value) {
        return value != selectedNegTag;
    });
    setSelectedNegTagList(filtered)
};
```

Figure 6: Code snapshot of the review form regarding the remove tag functions.

5 Overall Functionality

5.1 Search by CaseID/Major/Name

The user could search for students by caseID, major, and name as shown in Figure 7.

Status: Complete

5.2 Display Search Result

The website will display search result of users' input which will include the average ratings, course, graduation year and so on of each student as shown in Figure 8.

Status: Complete

5.3 Login/Signup

Users can sign up and automatically create their own profile by filling in the authorization form as shown in Figure 9. User can also log in using caseID and password as shown in Figure 10.

Status: Complete

5.4 Add a New Student

Users who have logged in can add a student who doesn't have a profile yet to the website as shown in Figure 11.

Status: Complete

5.5 Add a New Review

Users who have logged in can add a new review for a student if they haven't reviewed yet as shown in Figure 12.

Status: Complete

5.6 Submit Review Correction

Users who are the authors of the review can submit correction of the review as update as shown in Figure 13.

Status: Complete

5.7 Display Student Page

The comments and information display is one of the most essential parts of the application. Users are able to see the ratings, past comments, and related

information (such as majors, project courses, graduation year) of the specific student as shown in Figure 14

Status: Complete

6 Discussion

The information displayed demonstrates the integration of both the frontend and backend elements. The backend controllers are logically linked to the database functions and use RESTful API to connect to the frontend. Additionally, with the JDBC driver and Spring Data JPA, entities classes were developed along with annotations to map the SQL database with the backend maven project. As such, the overall functionalities listed above are linked to the frontend react components and are controlled by the logic mappings via JDBC and Spring Data JPA as specified in the Project Object Manager. The coverage reports from the testing of the integrated backend entities and controllers are supplemented below, demonstrating sufficient coverage.

com.cwru.backend.dal.entities

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
Profile		76%		n/a	13	28	14	41	13	28	0	1
Review		76%		n/a	13	28	14	41	13	28	0	1
Account		77%		n/a	8	18	9	26	8	18	0	1
ReviewNegativeTag		78%		n/a	3	8	4	11	3	8	0	1
ReviewPositiveTag		78%		n/a	3	8	4	11	3	8	0	1
NegativeTag		80%		n/a	2	6	3	8	2	6	0	1
Majors		80%		n/a	2	6	3	8	2	6	0	1
PositiveTags		80%		n/a	2	6	3	8	2	6	0	1

7 Work Allocation

- **Nora Tang:**

  1. Participated in the design and implementation of the database.

  2. As a member of front-end sub-team, implemented code in the **/FormElement**, **/RatingElement**, and **/StudentElement** (UI)folders, **AuthForm.js**, **AddNewStudent.js**, **ReviewForm.js**, **StudentPage.js** (UI), **SubmitCorrection.js**, and configured the initial routing for each page.

  3. Wrote **ReviewNegativeTagsController.java, ReviewPositiveTagsController.java, NegativeTagController.java, PositiveTagController.java.**, and **Review/add**, **Review/update** of ReviewController.java of the backend.

4. Participated in writing API document for REST API.

5. Fixed minor bugs in the back-end code.

- **Liyuan Huang:**
  1. Participated in the design and implementation of the database.

  2. As a member of front-end sub-team, implemented result page, navigation bar, website icon, and card wrappers under the /UI folder. Participated in the implementation of search page. Also completed the connection between search, result, student pages with the back-end, and the data parameters passing between search, result, student, review, and submit correction pages.

  3. Created and wrote the API document for REST API.

  4. Fixed minor bugs in the back-end code.

- **Haihan Jiang:**
  1.Maven Configurations for dependencies with JDBC drive and Spring JPA Data.

  2.Completed **Every Model Class** mapping to MySQL Database tables.

  3.Completed **Every Repository Interface** with JPA repositories to ensure basic CRUD(Create,Read,Update,Delete) operation and customized Criteria Search.

  4.Completed **Every Service Interface and Service Implementation Class**.

  5.Completed **Account,Major,Profile,Review Controllers** in backend to guarantee SignUp, Login, Search, Update Profile, Review Correction functionalities could work peoperly.

- **Walter Nam:**
  1. Participated in the UI design and functionality of the web application.

  2. Participated in the design and development of the database, including the ER diagram and sample queries.

  3. Set up the template code for the backend maven project, ensured the correct project structuring and hierarchy.

4. Generated coverage report using JaCoCo on build to ensure code entities were properly functioning.

8 Conclusion

The final product of the Rate My Teammate web application meets most of our expectations in the proposal report. With our cooperation and hard work through this semester, we implemented features that include Sign up, Log in, search by Major/CaseID/name, Add a new student, add a new review in perspective of rating, class information, and comments, and submit a correction for the review. In general, our web application exhibits a high level of usefulness – help Case students to have a general idea about classmates' personalities and capabilities, and find teammates who have expected skills to complete projects or competitions.

9 Appendix

9.1 GitHub Repository

Please refer to this link for more implementation details.

9.2 Front-end Snapshots

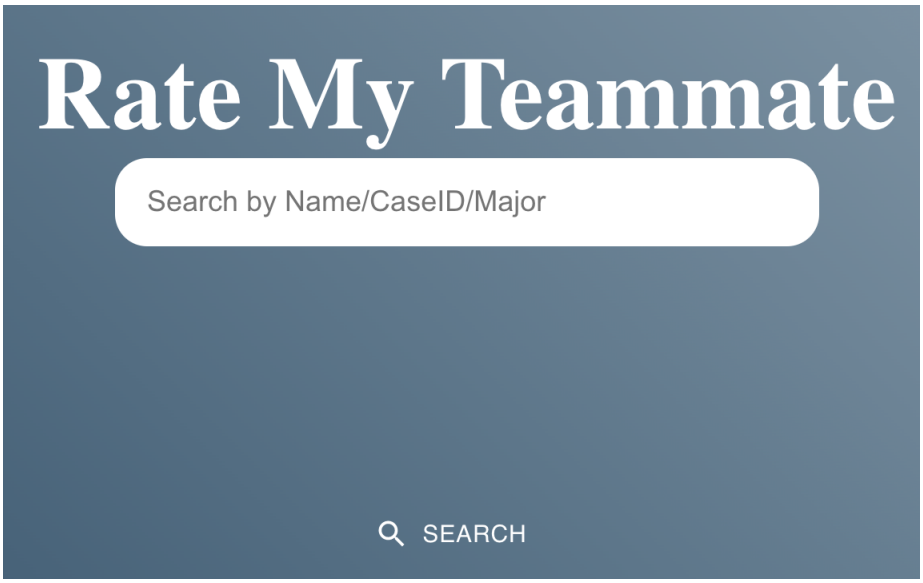

Figure 7: Search Page

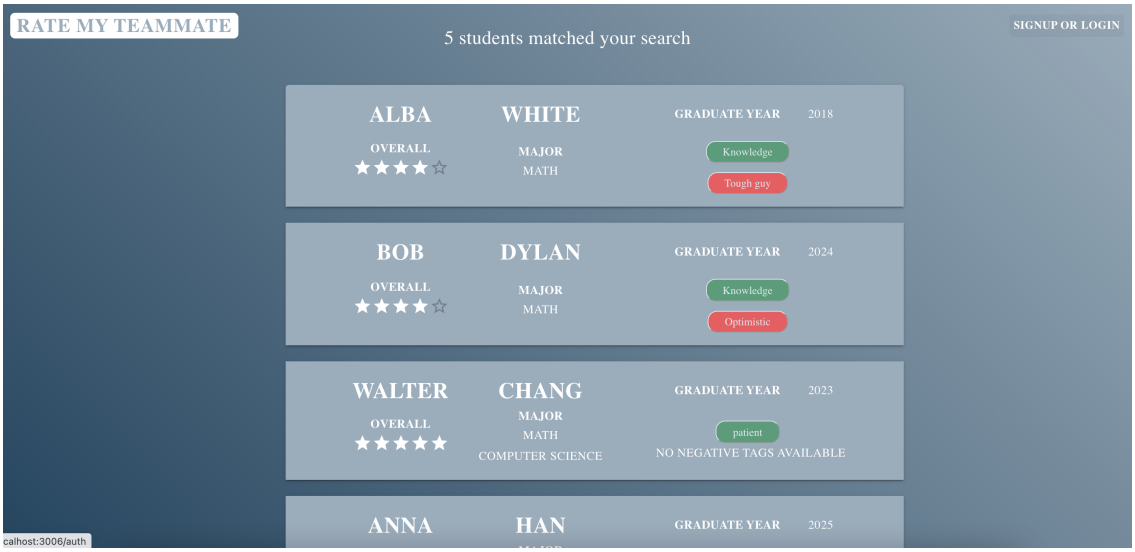

Figure 8: Result Page of finding students whose major is “Math”

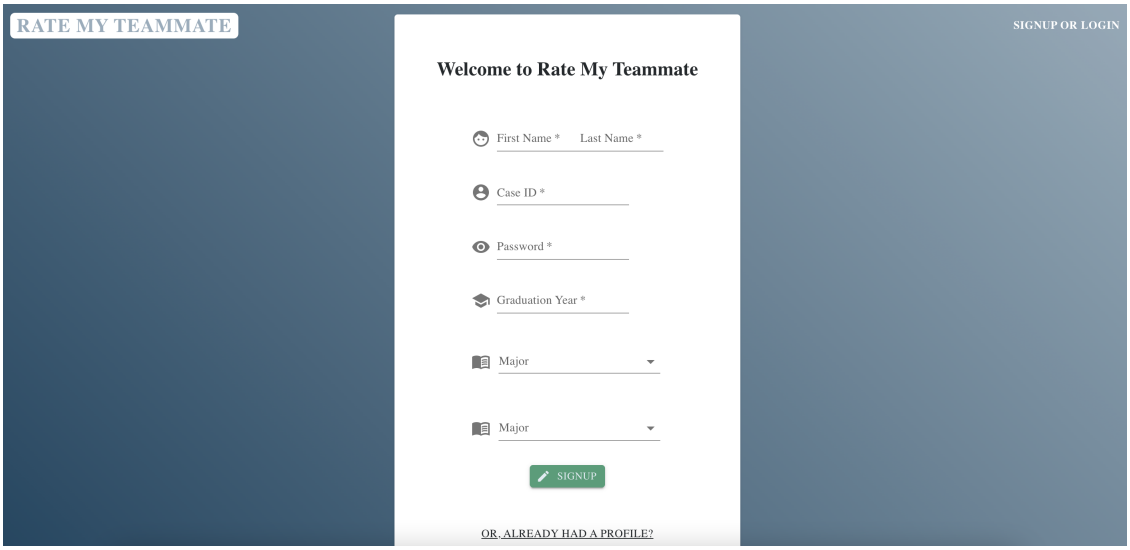

Figure 9: Sign up page

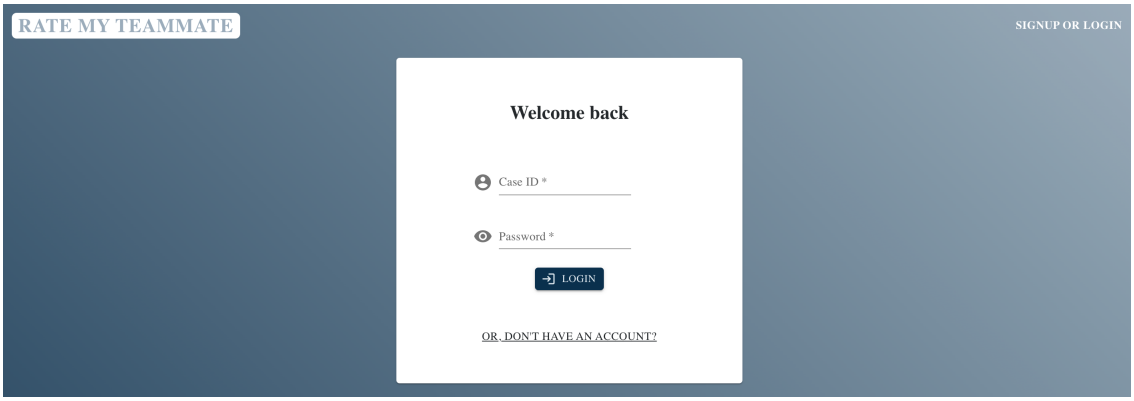

Figure 10: Login page

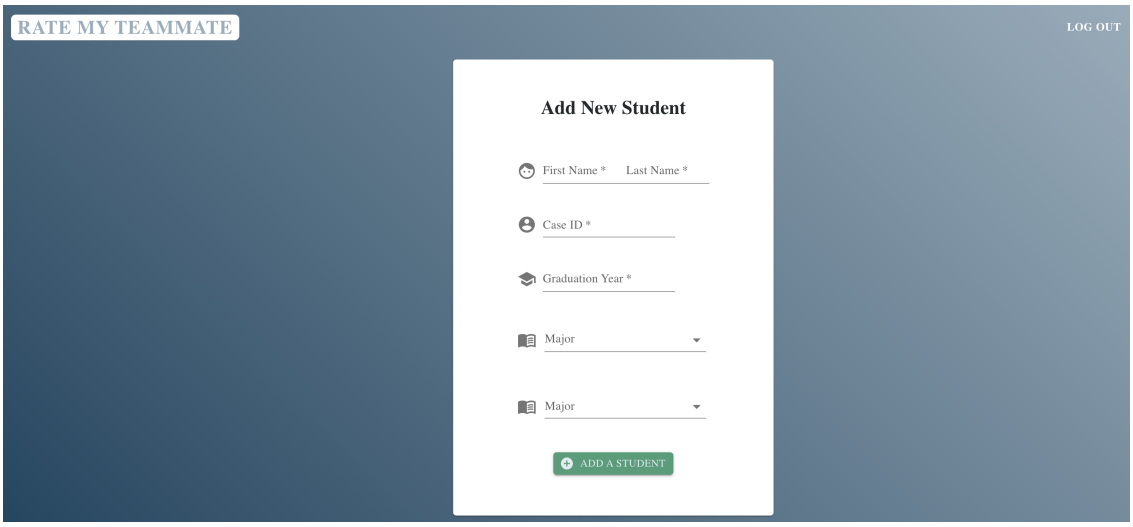

Figure 11: Add new student page

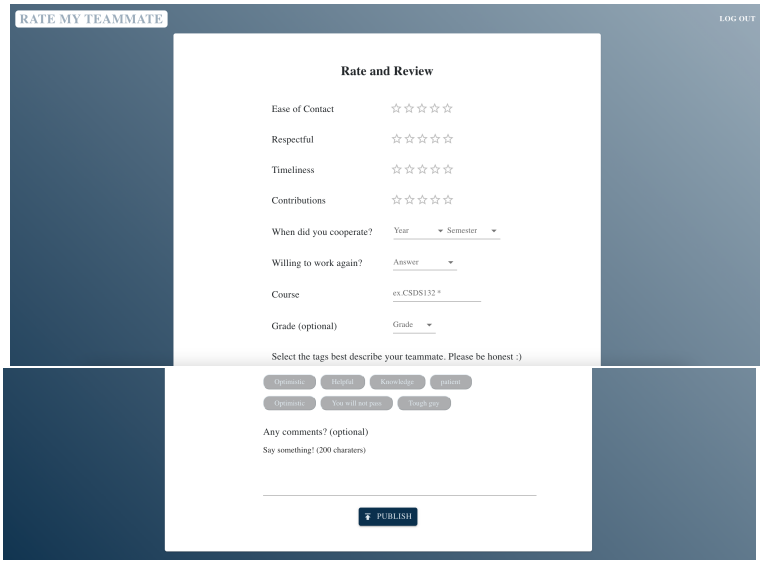

Figure 12: Add new review page

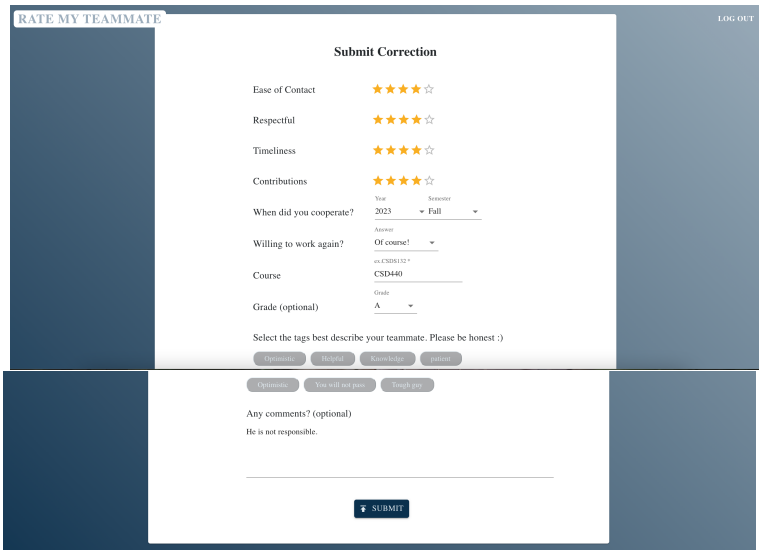

Figure 13: Submit correction for a certain review

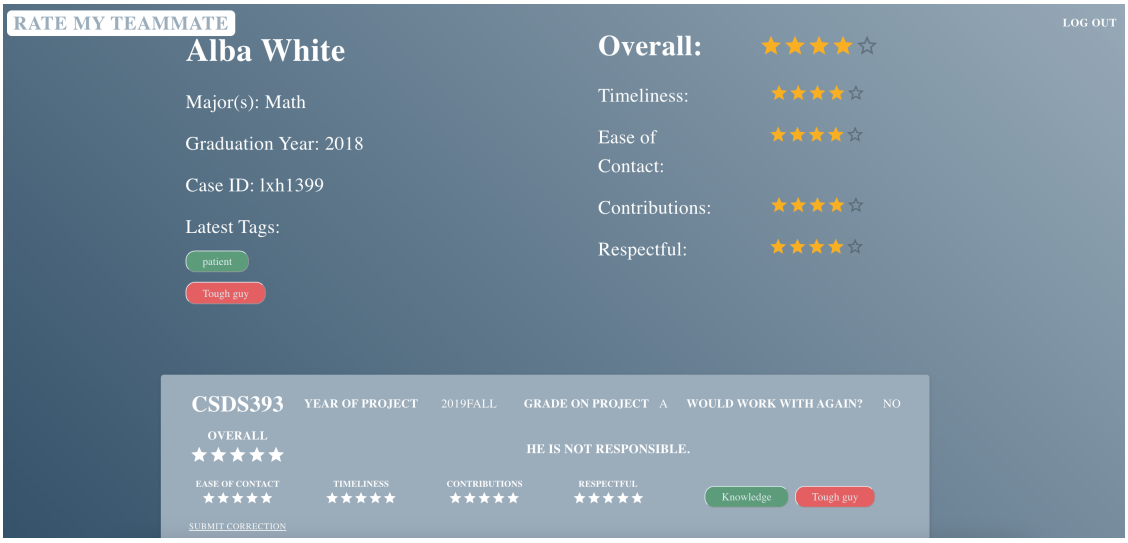

Figure 14: Sample student information page

Rate My Teammate (ver.CWRU)
Individual Report

Liyuan Huang

November 2021

1 Project Responsibilities

As a group, we implement the web application by breaking it down to the database, front-end, and back-end. The database part was implemented by all of the team members. Due to time considerations, we split our team into front-end and back-end sub-teams.

As a member of the front-end sub-team, I individually completed the result page and form wrappers. To interact smoothly with the back-end, I also finished the connections between several pages and the back-end database.

Here are my major contributions:

1. Participated in the design and implementation of the database.

2. Wrote most of the code in the **/UI** folder which work as form wrapper. Completed the several .js files in the **/Page Element** folder, such as **NavBar.js**, **RateIcon.js**, **searchResult.js**, and **LoginButton.js, SignupButton.js, AddStudentButtons.js**. Additionally, I was responsible for the implementation of **resultPage.js**, participated in the implementation of **SearchPage.js**, and configured the **data parameters passing** between search, result, student, review, and submit correction pages.

3. Created and wrote the **API document** for the connection between front-end and back-end.

4. Fixed minor bugs in the back-end code.

Finally, I appreciated Nora and Haihan's cooperation and help on front-end implementation and React-Spring Boot connection issues.

2 Achieving Objectives

Compared to the Java application I have done in the CSDS 395 class, this project is much more complicated and harder in the perspective of workload and implementation difficult level. In the previous project, I was only responsible for designing, implementing Java GUI, and allowing it to interact with Spring Boot back-end. Therefore, it is my first time coding in the JavaScript language and React platform. Fortunately, with the help of online education websites, such as Youtube and StackOverflow, I became more familiar with JavaScript, CSS, React, and Rest API. By integrating the knowledge learned from online resources, I finished the front-end part step by step and completed the major proposals with my teammates.

3 Issues

The data parameters passing between different pages is the major problem I encountered in this project. Because of the lack of deep understanding of React, I stuck and wasted several days on not working data parameters passing method. By searching the state-of-art data transferring methods online and applying them in our project, I finally found an applicable method and solved the data parameters passing problem successfully.

Besides, during the process of fetching data from the back end, I realized we need to consider more thoroughly in writing the initial API document. The lack of thoroughness in writing the API document results the incomplete and useless implementations of some of the REST API.

4 Lessons Learned

Through this project, I have a deeper understanding of JavaScript language, REST API, and React, Spring Boot platforms.

I also realized the importance of considering comprehensive in the initial stage of design. The thorough considerations will avoid many unnecessary efforts later.

Finally, I learned that communications between team members are pretty significant. The tasks should be assigned to each group member accordingly in the initial stage.

5 Insights

Overall, it is a pleasure to work with my teammates to implement this Rate My Teammate web application. Although we are not familiar with each other before and have a limited understanding of React and Spring boot platforms, we still cooperate well and deliver it in the manner which meets our expectations. From my perspective, the main objective of web application is useful. Like Rate My Professor, I hope this application can help some students who are looking for responsible teammates. Although there exist some possible improvements in this application, it is our first step to explore the world of software. We will design and implement more useful applications in the future career.

Individual Report:Rate My Teammate (ver.CWRU)

Haihan Jiang

November 2021

1 Project Responsibilities

In this peoject, I am in charge of Backend Design and Implementations. With out team, we also complete discussion for dtatabase and prototype designs.

In the Backend Module,I am responsible for implementing CRUD operations, Criteria Queries,general Requests Handling, and some backend functionalities.

Specifically,

- Maven Configurations for dependencies with JDBC drive and Spring JPA Data.
- Completed **Every Model Class** mapping to MySQL Database tables.
- Completed **Every Repository Interface** with JPA repositories to ensure basic CRUD(Create,Read,Update,Delete) operation and customized Criteria Search.
- Completed **Every Service Interface and Service Implementation Class**.
- Completed **Account,Major,Profile,Review Controllers** in backend to guarantee SignUp, Login, Search, Update Profile, Review Correction functionalities could work peoperly.

  With heldps from Liyuan and Nora, we together make connection between backend and frontend smoothly. They also helped me to understand some logic between different modules.

2 Achieving Objectives

To achieve our goal, I read official documentations and video tutorials on Spring Boot, Maven, MVC deisgn, JPA Data, and Restful API to understand how to architect out Back-end Design. Start from scratch, I completed backend setup, connections, and core functionalities based on our designs. Exceptions and errors made me become more adapted on web application based on Spring Boot framework.

3 Issues

During the implementation, I encountered several issues:

- Annotations: Since Spring Boot needs annontation injection, and sometimes the missing Parameters or annotaions could cause errors, such as @Qualifier,@Autowired. After reading many technical blogs and official documentations, I am able to solve problems.
- Implementation foreign key implementations in Model classes. There are many constrints for Spring Boot to implement foreign keys and Easy to cause Data inconsistency Problems.
- JPA criteriaQueries. When we try to customize our queries, we may face many potential conflicts of Mapping between JPA and MySQL.

4 Lessons Learned

A great takeaway is I obtained from this project is that I provided a fully functional backend aned it is also maintainable application. In this project, I obtained deep understanding and Hands on experience in Spring Boot framework, MySQL, JDBC driver, Hibernate and JPA. Every debugging process made me understand those applications well.

Also, it gave me a lesson that design clearly first was much more efficient than writing code directly. Although the design of website is quite simple compared to websites that deployed in the industry, it made me understand more about software engineering process, which I may not understand well by learning from documentations or some tutorials. The hands on experience made me understand even add a simple feature needs many considerations

and discussions with others. We need understand the system fully first to make sure the changes are wise. It is important to set up core API first, and have sufficient discussions with frontend. These are all keys that can contribute to a successful Application.

5 Insights

This is my last project in CWRU. This collaboration of work gave me pleasure and helped me to improve a lot. We had setted up a clear and reasonable goal at the beginning of this semester. This project pushed me to learn some state-of-art techniques, and gave me opportunities to design a fully functional website. It is valuable for me to gain those practical experiences. Also, out team can communicate fluently to help others to know each other's progress and problems one face. With these inspiration, in the future, I will continue learn new knowledge in software engineering and become a qualified full stack engineer.

Individual Report: Rate My Teammate (ver.CWRU)

Nora Tang

November 2021

1 Project Responsibilities

As a whole team, we together completed the database design, the prototypes of the front-end. As a front-end subteam, we together discussed and broke down the components, and helped each other out whenever there are issues with rendering, or connection.

Individually, I am mainly responsible for the forms and student page UI design of the front-end, marrying the form connection between the front-end and backend, and trigger functions in the database design.

Specifically,

- Created, configured, and wrote instructions of the initial React project.
- Wrote the React.js code in the **/FormElement**, **/RatingElement**, and **/StudentElement** (UI) folders, **AuthForm.js**, **AddNewStudent.js**, **ReviewForm.js**, **StudentPage.js** (UI), **SubmitCorrection.js**, and configured the initial routing for each page.
- In controllers, I wrote all the java code in **ReviewNegativeTagsController.java, ReviewPositiveTagsController.java, NegativeTagController.java, and PositiveTagController.java.**
- In **ReviewController.java**, I implemented the following APIs: **Review/add**, **Review/update**.
- I also debugged for **AccountController.java**, and **ProfileController.java**.

- Other than all the above, I also fixed some minor bugs in the backend.

In addition, thanks to Liyuan Huang's and Haihan Jiang's hard work, they have helped me a lot with front-end design and connection issues.

2 Achieving Objectives

The workload and design of the front-end is far more complicated than my previous projects. During my previous project, I was only required to finish the basic form design and some search result rendering with React.js, Node.js and Express.js, and was mainly responsible for databse design with PostgreSQL. I didn't have enough experience with configuring React with Spring Boot and MySQL. When I realized that my knowledge for the stack development for this project is limited, I immediately set off learning React with Spring Boot as backend from the beginning in order to deliver a front-end above the standards. By going through each section in online courses, I began to be famliar with Javascript, CSS, and Restful APIs with Spring Boot. At the first stage, I followed the instructions of the online course to complete the same application as they did to gain basic knowledge of React. Afterwards, I started our project from scratch, transferring what I learned into my own understanding and writing. Step by step, I was able to deliver the required designs.

Later, when encountering with the Spring Boot connection, I also followed the same procedure as above. In addition, thanks to Haihan's work, I was able to follow his style and wrote the restful APIs for reviews and tags.

3 Issues

During the implementation, I encountered several issues:

- The position of React components are sometimes hard to adjust. Instead of using *pt* to set the distances, *rem* or percentage are better choices.

- Authorization status of the users is necessary which I didn't implement at the first place. Later on, I set the expiration time of authorization token to be 1 hour and stored the authorization token in local since

some pages require users' login status, otherwise, users should not be allowed to access these pages.

- During the implementation, I realized some fields of tables in the database should be revised to be more logical, such as the name should be separated into firstname and lastname. These issues have been fixed afterwards.

- Establishing connection was time-consuming. Because the backend team and front-end team were working separately in the first half stage, we did not realize that some fields in backend and frontend were not consistent. Such inconsistency took me a lot of time to re-configure the input and output data.

4 Lessons Learned

- **Communication matters.** The most essential one is that font-end and backend should work coherently from the beginning. And we should first specify our requirements of restful APIs, and set up rules for field names, which could save us a lot of time during integration. The efficient communication in our second half stage has speeded up our progress as well.

- **Familiarity of full stack development.** Creating a website with appealing look and efficient backend is a valued experience for me to become familiar with full stack development and sparkled my interest in further pursing software engineering.

- **Boosting both soft skills and technical skills.** From this project, I learned a project cannot be developed without neither cooperation nor technical skills. A great project needs both excellent skills and efficient communication.

5 Insights

Overall, it is a pleasure to work with my teammates to deliver a website within our expectations. Their excellence cannot be overstated. Although we never worked as a team before, and we didn't know each other's work

styles before this project, it is such a surprise that we could quickly be on the same line. With various backgrounds of our team, some having little experience with the front-end, some having little experience with the back-end, we are able to efficiently allocate our workload to accomodate everyone's needs. In addition, developing a simplest forum-like website from scratch requires more non-trivial and essential designs. It is not as simple as we used to assume. Passing data from parents to children components, handling input data type from form component through connection, setting up routing with considerations of users' security, setting up authorisation and so on are all essential parts to ensure users' experience. However, there's still available imporvement for our data visualisation and UI design. In the course of time, I will learn more tools to design a more user-friendly front-end.

I am more than grateful to be in such an outstanding team. It's the effort and dedication we made together that leads to our achievement.

Individual Report: Rate My Teammate (ver.CWRU)

Walter Nam

November 2021

1 Project Responsibilities

- Participated in the UI design and functionality of the web application.
- Participated in the design and development of the database, including the ER diagram and sample queries.
- Set up maven template code for the backend, added necessary dependencies to pom.xml. Configured for JaCoCo and Apache.
- Assisted in the development of **ReviewController.java** and **ProfileController.java**
- Developed unit tests connecting the backend SpringBoot and the MySQL database.
- Generated and compiled JaCoCo coverage reports, demonstrate sufficient coverage for all entities.

2 Achieving Objectives

This project was the first time I would work with a SpringBoot Web Application merged with React frontend components. In my previous experience and projects, I assisted in the creation of SpringBoot and Django projects connected to basic HTML and CSS. Additionally, I had less experience in developing with maven projects, as my academic Java programming focused on

projects with Gradle and AntBuild. Additionally, SpringBoot syntax changes quite frequently, so even with prior experience from years ago, I found myself frequently looking up issues on StackOverflow and Baeldung. Fortunately, after sufficient self-study, I was able to work past the quirks of the both the framework and VSCode, I was able to test the basic functionality of the main project entities.

3 Issues

Throughout the course of the project, I encountered several issues:

- I was initially using IntelliJ Ultimate during development; however, there were nagging issues in compiling the project from a shared repository. These issues were resolved from switching instead to VSCode.
- Occasionally, I would get code formatting errors due to differing line endings between Unix and Windows systems working concurrently on a shared repository. I resolved these issues by occasionally switching to a Linux system using VirtualBox.
- During testing, the team noted an issue with the date format in both the database and the Review Controller. I noticed that the format had to be converted to epoch, and resolved the issue during testing.
- Testing revealed inconsistencies between the formatting of the SQL database, and the backend entities. During testing, I resolved the issues and was able to establish concurrent database connection while testing the backend.

4 Lessons Learned

Overall, I learned quite a lot during the project, and believe this experience will make me more comfortable as a software engineer in a professional setting. I am indebted to the rest of the team for the hardwork and commitment that was exemplified throughout the course of the project. Aside from the technicals, I will share some vital lessons I learned that will be useful in my future looking ahead.

- **It's not about the number of tools you know:** Even though I had prior experience with SpringBoot, this framework is constantly updating. As such, I found myself in unfamiliar territory with the project specifications during testing. However, I eventually discovered that I only needed to think logically, and utilize my problem solving skills to tackle my tasks at hand, and not necessarily worry about syntax for a fickle language. Syntax comes as second nature through experience. As such, the language you know is just a tool to get the job done.

- **Be Meticulous:**

  Despite my studying of Code Complete, I never realized the full extent of the importance of clean, easy to read code. As a personal example, when writing test queries and unit tests in Java, the inconsistencies resulted in time wasted debugging, and getting the server to run again. I learned at this moment that you never know when you'll ever have to work with a piece code again, and productivity can be impacted even by the smallest inconsistencies with the smallest snippets of code. If your code is messy, whoever works on it next will not be able to add features or maintain it.

- **Networking/Teamwork:**

  Group projects offer the opportunity to learn from the peers around you. Even though we did alot of self study, it turned out that I would learn the most from other teammates during our group meetings throughout the course of the project. As such, I had gained individual software engineering experience, supplemented with valuable communication and collaboration skills.

5 Insights

Overall, it was pleasure working with the team, and the craftsmanship of the finished product cannot be undermined. Even though we had never before worked together, there was a consistent aura of professionalism and mutual responsibility that delivered us an application that sufficiently met our expectations. Despite our collective limited experience with SpringBoot and React, we had managed to develop a rating website with basic functionality

within a very limited time frame. This website is geared with a similar objective to RateMyProfessor and RateMyDorm, and I hope Case students will find this application to be a useful tool in finding group project teammates.