UniConnect Coding Project Final Report

Prepared by Group 4
Syed Mehdi, Syed Shaban, Dimitar Gjorgievski, Oscar Franco
for use in CS 440
at the
University of Illinois Chicago

May 2024

Table of Contents

List of Figures

List of Tables

No tables in this Document

I Project Description

Universities bring together students from all over the world with various backgrounds, education systems, thought processes, and experiences. In-person campus activities and lecture halls made it extremely easy for students to mingle and learn from each other. But in the modern world, most of the connections are established on the internet. As a university-specific social networking application, UniConnect will provide a platform for students to make more informal connections with their peers to hopefully build friendships and connections for life during and beyond academia.

1 Project Overview

UniConnect aims to bridge the gap between the physical and digital realms of university socialization by providing a comprehensive social networking platform tailored specifically for UIC students. The platform facilitates informal connections among students, fostering friendships and professional networks that extend beyond the confines of academia. By leveraging modern web technologies, UniConnect offers students a user-friendly interface where they can interact, share experiences, and engage in both casual and formal discussions. Users are able to express themselves and share their thoughts with their peers, as well as have valuable discussions pertaining to their studies. Students can follow each other, building networks of connections based on shared interests, academic pursuits, or personal affiliations. By facilitating both casual interactions and formal discussions, UniConnect enriches the university experience for UIC students.

2 Project Domain

This project is a full-stack social media web application. It utilizes technologies to insert, retrieve, modify and delete information pertaining to users. The idea behind it is to facilitate an environment for students where they, as the user, can comfortably navigate it. As this project looks to enhance the accessibility to the social domain to UIC students, establishing proper means of communication is of great importance. Since this is a UIC specific service, students will be able to create their account using their school's credentials. Also, students can express themselves by creating posts, which will be available to the peers that follow them, and for that we need to be successfully storing those posts and displaying them to the user, as well as having a historical record of each post. Establishing connections is also a crucial aspect of this project. The students can be followed by a student, and themselves follow another student, those groups may intersect but they also may not. Students are also able to manually look up some students if they wish to establish a connection or look at their content. This is done through the functionality of the search bar, which takes the student's input and looks through the database for possible matches. Visiting another profile, gives the user the possibility of evaluating if the want to establish a connection with another user or simply reading the specific content pertaining to a single user. Another space for students to interact with each other is through a discussion board, which unlike the personal feed of a student, contains the input of every user. This is to facilitate a space where the users can discuss more formal topics

with each other regarding their studies. In conclusion, this is a project that creates online spaces for UIC students to engage in casual and formal discussions.

3 Relationship to Other Documents

This document relates to the UniConnect Project Report, prepared by Anita Padman, Rafiya Awan, Ollie Rotkis and Ameesha Saxena in the Spring of 2021. This project takes the idea of the project report, but does not follow the group's template for implementation. It also does not implement some functionalities described in the development project report. Functionalities like the messaging system were not established in our coding project due to time constraints. Also, the intended system design and other design patterns from the development project were not considered in the coding project. This decision was made by the coding project group, due to a different vision of the project structure.

4 Naming Conventions and Definitions

The naming conventions and definitions of this coding project follow the ones specified in UniConnect Project Report.

II Project Deliverables

This section describes the approach and functionality of each release as well as the differences in our app from the original idea that was conceptualized in Spring 2021 by Group 7.

1 First Release

February 23rd, 2024

For our first release, we demonstrated our React App with Spring Boot in the back-end and MySQL as the database, following the MVC Software Architecture. The app had a working login and sign up for both Students and Organizations, as well as a homepage that laid out our plans for the upcoming releases.

Deciding the architecture took some time, and even set us back. We started with plain HTML, CSS, and JavaScript, but realized that it would not be enough, so we had to migrate all of our code from HTML/CSS/JS into a framework, which is React.js, as well as have a proper back-end, which ended up utilizing the Spring Boot framework as well as MySQL as the database.

The main achievement for this release was the communication between the front-end, done in React.js, and our backend, using the Spring Boot framework in Java, and its relation to the database in MySQL. This communication helped us have a fully functional login system and the ability to create accounts and store them in the database. In this release, we spent a lot of time ensuring the authenticity of this application.

Figure X shows how the objects were designed in the back-end using Java, with primary focus on the Student object and its functionality first in this release and the following releases. **Figure X** shows the simple architecture this application uses

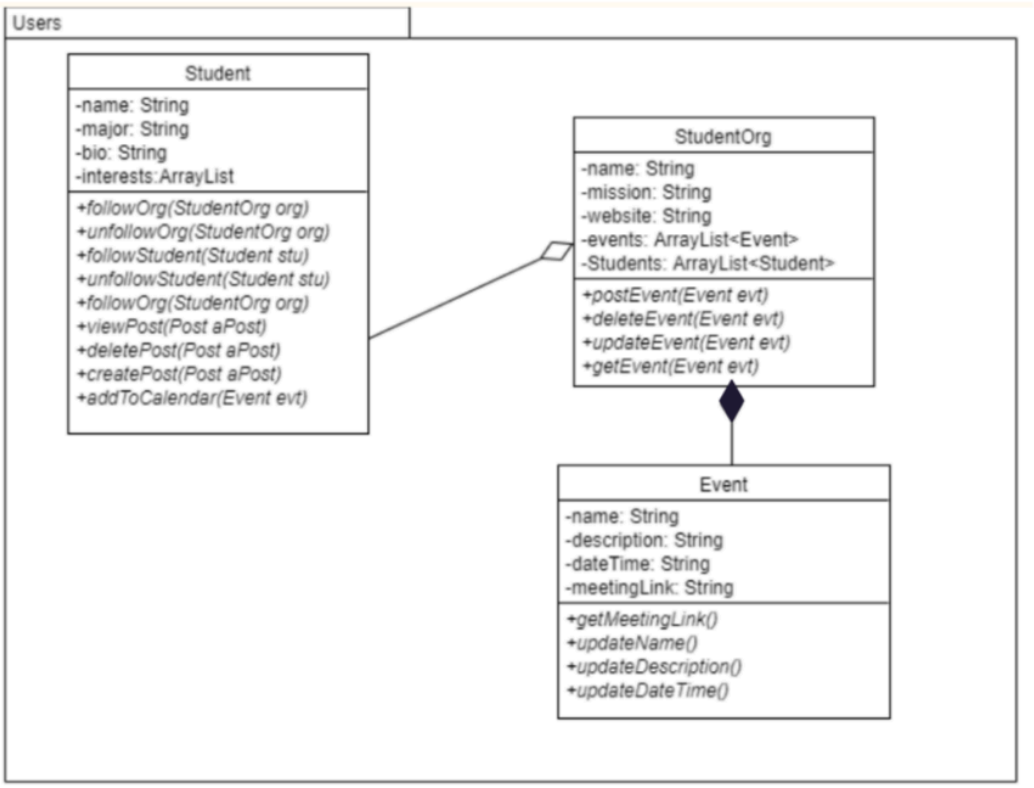

Figure 1: Student and Student Org UML Diagram

Source: UniConnect Project Report Spring '21

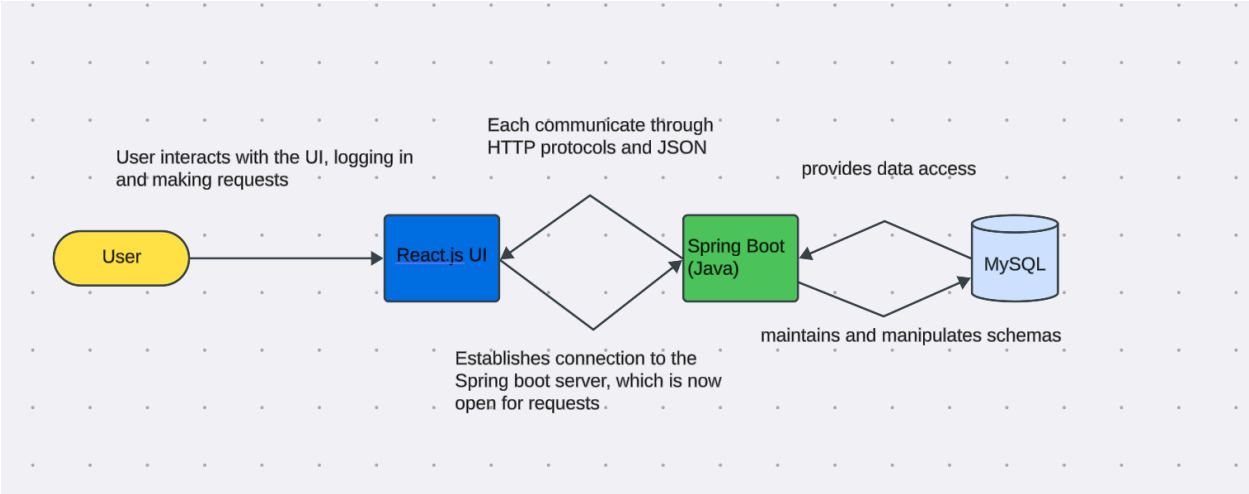

Figure 2: System Design of First Release

2 Second Release

March 29th, 2024

For our second release, we added more functionality in the front-end for the user to do, as well as expand the ability of the back-end.

Firstly, we deployed our database using AWS onto the cloud. This allowed us to all share the same centralized database while developing the project, and was a necessary step to have a functioning and authentic application.

Then, we added front-end features for the Discussion Board aspect of the application. The user was able to add threads as well as reply to threads which relate to the classes the user is currently taking.

We also added a profile page, where the user can view their own details and edit them, as well as view their own posts.

The main functionality added in both front-end and back-end was the ability to create posts. This allows users to share their thoughts or any important information to their followers. The posts were displayed on the homepage timeline in order of most recently posted, mimicking Twitter.

One major setback we had during this release was that we could not add functionality for pictures. Our aim was to leverage the Imgur API for hosting all pictures uploaded, but communication with 2 servers at once was complicated, and we decided to scratch this feature after some time.

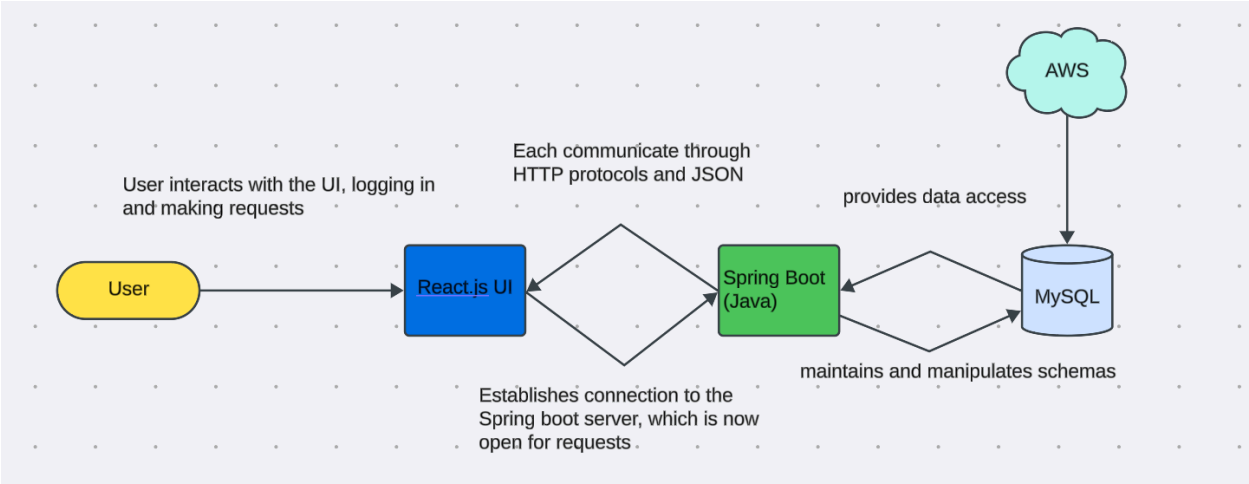

Figure 3: Updated System Design

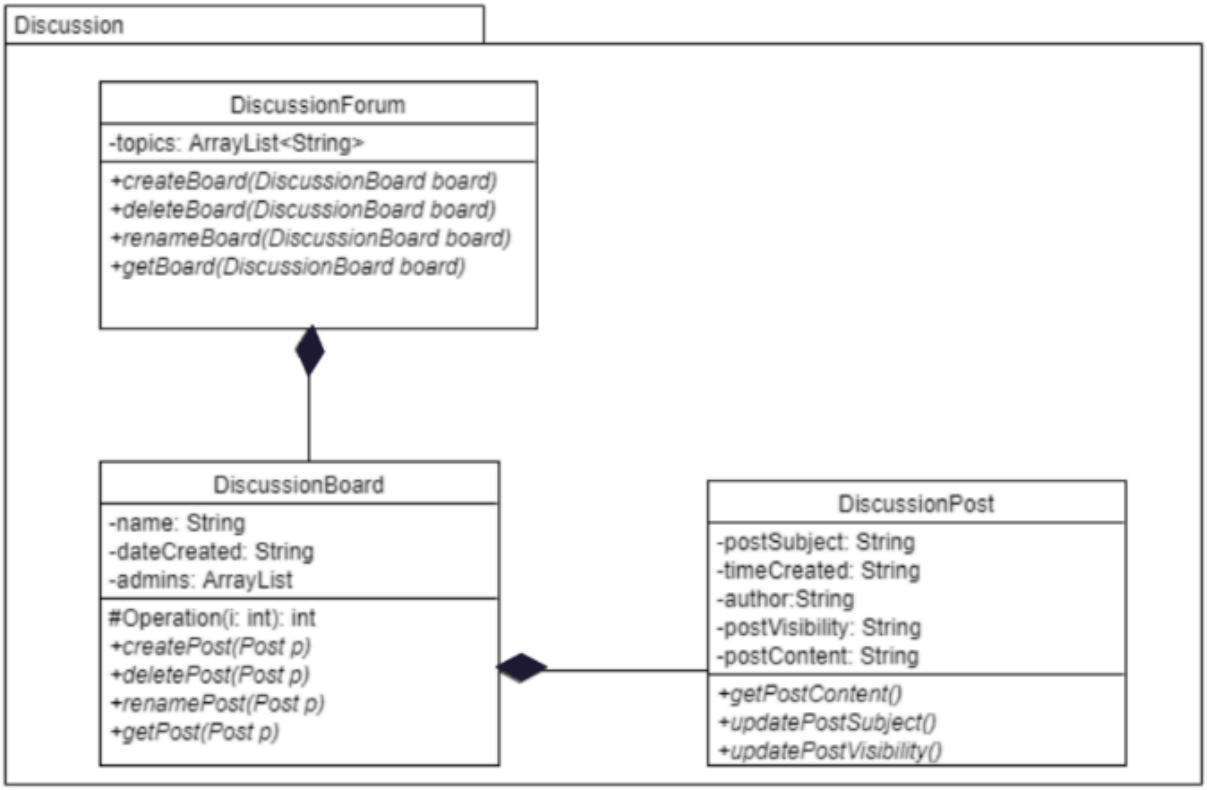

Figure 4: Discussion Board UML

Source: UniConnect Project Report Spring ‘21

3 Comparison with Original Project Design Document

Our prototype follows the initial idea of UniConnect, trying to be a social media app to unite students and organizations into one centralized application monitored and maintained by the respective university, however, due to some creative differences as well as a lack of experience and capability from our part, the application went a different route than originally planned by Group 7 from Spring 2021.

In **Figure 1**, we see that StudentOrgs inherit Student, and instead, we decided to make Student and Organization separate entities because they would be storing a lot of different things. However, in hindsight, the original design would have made the app even more functional in lesser time because we have functionality for mainly the Student entity, and to make the same functionality for the Organization entity, we would need to do redundant work, so we decided to focus on other aspects of the application.

The messaging system was intended to be 1/3rd of the application, however we scratched that entire idea due to the time limit we had for this project.

The discussion board is also a major aspect of the project, but due to time constraints, that feature has remained static but still functional and stateful.

However, the follower/following system, the search feature, profile creation and editing, viewing other profiles, and making posts is something that we tried to follow as close as possible to the documentation.

Overall, we started off with the original idea, and then made adjustments to how we see fit and also compartmentalized due to our abilities and time constraints. Our intention was to be as close to a real social media app as possible, and for that we ended up sacrificing functionality, for better or for worse.

III Testing

1 Items to be Tested

Front-end:
- Login and Sign Up Forms
- Discussion Board Functionality
- Profile Page
- Post Creation

Back-end:
- Database Connection
- User Authentication
- Thread Creation
- Post Creation
- AWS Deployment
- Error Handling

2 Test Specifications

ID# 1 - Form Validation and Profile Creation

Description: Test Validation of login, sign up forms, and account creation.

Items covered by this test: Front-end form validation, and a profile will be created and stored in the database

Requirements addressed by this test: N/A

Environmental needs: N/A

Intercase Dependencies: N/A

Test Procedures: Enter invalid inputs into login and sign up forms and verify error messages.

Input Specification: Invalid inputs for email, password, etc.

Output Specifications: Error messages displayed for invalid inputs. Lead to homepage.

Pass/Fail Criteria: Forms should validate inputs and display appropriate error messages. Profile should be created.

ID# 2 - Thread Functionality

Description: Test the ability to create new threads and reply to ones made.

Items covered: Front-end thread creation.

Requirements addressed: N/A

Environmental needs: N/A

Intercase Dependencies: N/A

Test Procedures: Create a new thread and verify its display as well as respond to others.

Input Specification: Thread title and content.

Output Specifications: Newly created thread displayed on the discussion board.

Pass/Fail Criteria: Newly created threads should appear correctly.

ID# 3 - View and Edit Profile

Description: Test viewing and editing user profile details.

Items covered: Front-end profile page functionality.

Requirements addressed: N/A

Environmental needs: N/A

Intercase Dependencies: N/A

Test Procedures: View and edit profile details and verify changes.

Input Specification: User profile details.

Output Specifications: Profile details should be displayed and editable.

Pass/Fail Criteria: Profile details should be editable and changes should be saved.

ID# 4 - Creating Posts

Description: Test the ability to create posts.

Items covered: Front-end post creation.

Requirements addressed: N/A

Environmental needs: N/A

Intercase Dependencies: N/A

Test Procedures: Create a new post and verify its display.

Input Specification: Post content.

Output Specifications: Newly created post displayed on the homepage timeline.

Pass/Fail Criteria: Newly created posts should appear on the timeline in the correct order.

3 Test Results

ID# 1 - Form Validation and Profile Creation

Date(s) of Execution: 01/30/2024

Staff conducting tests: Syed Shaban, Syed Mehdi

Expected Results: Error messages displayed for invalid inputs.

Actual Results: Website identified invalid inputs and gave error message.

Test Status: Pass

ID# 2 - Thread Functionality

Date(s) of Execution: 04/04/2024

Staff conducting tests: Oscar Franco

Expected Results: Newly created thread displayed on the discussion board.

Actual Results: New Threads are displayed on the discussion board.

Test Status: Fail, Threads are not connected to the database so will not be saved after leaving the page making this functionality useless for now.

ID# 3 - View and Edit Profile

Date(s) of Execution: 03/20/2024

Staff conducting tests: Syed Shaban, Dimitar Gjorgievski

Expected Results: Profile details should be editable and changes should be saved.

Actual Results: Profile reflects changes such as: Name changes, Post made, and Follower and Following count.

Test Status: Pass

ID# 4 - Creating Posts

Date(s) of Execution: 03/28/2024

Staff conducting tests: Syed Shaban, Syed Mehdi, Oscar Franco, Dimitar Gjorgievski

Expected Results: Newly created post displayed on the homepage timeline.

Actual Results: Every user's feed is unique to them showing posts from their following only and giving the correct timestamp.

Test Status: Pass

4 Regression Testing

NONE

IV Inspection

Our inspection process was separated into components of pages in React, where each person would analyze each of the components that were not made by themselves to scan

for errors or look for possible refactoring to increase reusability, readability, or efficiency in any other way.

1 Items to be Inspected

Main Pages of the Website -

- DiscussionPage
- FriendsPage
- HomePage
- LoginPage
- ProfilePage
- SignUpPage
- VisitProfile

2 Inspection Procedures

Checklist:

- ☐ Possible opportunities for creating more components to allow for code reusability
- ☐ The UI should be visually appealing and contain the correct information for the component
- ☐ Compiles and runs well (with full functionality)
- ☐ Each button or event on each page doesn't cause errors(test for edge cases where possible)
- ☐ Simplification opportunities (for readability purposes)
- ☐ General Error-Checking

3 Inspection Results

VisitProfile inspection performed by Syed Shaban (04/26/2024 3:40pm), Oscar Franco (Sometime in early April), and Syed Mehdi (04/25/2024 3:00pm):

- All buttons worked as intended MOST of the time.
    - There was an error that was showing up when somebody would attempt to follow or unfollow somebody else, randomly, still unsure about what causes this issue, or how to recreate it..
- Decided that the "+friend" button should be changed to a "follow/unfollow" button
- Needed to make sure that people's posts were the only one's showing up when visiting somebody else's posts.
- No errors other than what was mentioned earlier.

HomePage inspection performed by Syed Mehdi (03/01/2024 2:15pm), Oscar Franco (03/20/2024 3:00pm), and Dimitar Gjorgievski (04/3/2024 3:00pm):

- All buttons worked as intended.

- Decided that HeaderNavBar should be created out of the NavBar on the top of the screen as a whole new component that can be used across pages.
- Decided that Posts should be their own component
- Decided that HomePage should have a MiniCalendar widget to fill up sidebars that can lead to a LargeCalendar by clicking on it.
- Removed unnecessary buttons (Friends, which was replaced with Followers/Following, Messages, Friend Recommendations, Notifications)
- Added a way to log out as a part of the HeaderNavBar, instead of the Homepage itself, so that it can show up on the HeaderNavBar across pages.

LoginPage inspection performed by Syed Mehdi (03/01/2024 2:00pm), Oscar Franco (03/01/2024 2:00pm), and Dimitar Gjorgievski (03/01/2024 2:00pm):

- All buttons and events were handled as intended.
- Decided that the background image should have lighter opacity.
- All logic was efficiently done, and seemed to be very readable.
- All appropriate error-checking was done (for the email and password)
- Decided to add a ‘show password’ checkbox.

SignUpFirst inspection performed by Oscar Franco (04/26/2024 2:45pm), Syed Shaban (04/26/2024 2:45pm), and Dimitar Gjorgievski (04/26/2024 2:45pm):

- There was an error with the tab highlighting at the bottom of the page that shows the stages of where you are in the process of signing up, that wasn’t properly functioning. This was fixed on the spot.
- Decided that the loading of the profile picture was unnecessary, considering we never figured out how to store images on the SQL databases, so there was no use of having that, but still kept it.
- All of the buttons worked properly.

ProfilePage inspection performed by Syed Mehdi (04/10/2024 2:00pm), Oscar Franco (04/24/2024 2:00pm), and Dimitar Gjorgievski (04/24/2024 2:00pm):

- Should be optimized to reuse the Post component made from the HomePage component.
- Should remove all unnecessary/unimplemented buttons (such as ‘About’).
- Should include an ‘Edit Profile’ button to edit one’s profile, linking to another page that allows somebody to edit attributes of their profile such as their name, profile picture, or banner picture.
    - Edit Profile was later inspected, and all buttons did not work as intended, as the backend was not implemented to be able to save images to our SQL database, so the buttons that allowed the user to choose a picture to replace their banner/profile picture with were rendered useless, as they didn’t change anything after being chosen.
- Should add follower/following count, as well as follow button.
- All buttons and events were handled as intended (except for one error that occasionally shows up when attempting to edit somebody’s name, randomly, once in a while, still unknown what is causing that error).

FriendsPage inspection performed by Syed Mehdi (04/24/2024 2:15pm), Oscar Franco (04/24/2024 2:15pm), and Dimitar Gjorgievski (04/24/2024 2:15pm):

- All buttons worked as intended, no errors
- Decided that the friends page should be changed to the following page (to be placed juxtaposed to the followers page), instead of having traditional facebook friends, closer to the format that Instagram follows.
- Code and logic seemed to perform at optimal runtime with high readability.

DiscussionPage inspection performed by Syed Mehdi (04/26/2024 2:30pm), Syed Shaban (04/26/2024 2:30pm), and Dimitar Gjorgievski (04/26/2024 2:30pm):

- Decided that we should separate the threads into a different component
    - Created Discussion.jsx to fix this issue
- All buttons worked as intended, but didn't connect to SQL, which still needs to be fixed.

V Recommendations and Conclusions

This project was a huge success for us, as it was all of our very first attempts at making a full-stack application, and to get this up and running with the features that we have implemented is an accomplishment for all of our groupmates. The most notable features of our application currently are our account creation/verification processes, post processes, discussion-related processes, and friends-related processes.

Not all of our tests and inspections went perfectly, as you can take from the results above, and we ended up leaving a couple of features that we had planned to implement in this release version. These are our next steps:

- Testing for the discussion page revealed that discussions were not being saved in the SQL database properly at all, leading to a complete removal of it being included in the SQL database at all for the final release. Hence, the absolute next step would be reworking the SQL schema to include more tables to store information about Discussion threads, replies, etc, so that we can pull and push data from our server, instead of just manipulating the page locally.
- As mentioned in the inspection section, the inspection and tests for the EditProfilePage were largely unsuccessful when related to images, because of our problem with saving images to our SQL database. We attempted to upload them to an Imgur server using an API, but this failed, so this would be one of our next steps, as well.
- Adding Messages: A big part of this application was its messaging feature, which allowed friends (followers/people you are following) to directly message each other. We never got to the point where we could start implementing this, as we deprioritized it at the very beginning, because even though we focus on centralizing other platforms like Piazza and Facebook (catered towards UIC students), messaging is something that is possible through phones and other devices that people don't absolutely need for this application's main idea.

VI Project Issues

1 Open Issues

Images:

Considering our project uses an MVC Structure (Spring MVC), using SQL as the Model, which we decided about a week or two into the project, we knew we needed to store most, if not all, of the data that people provide into the website early on, online, which we used AWS to keep online. Something that we knew was integral to most social media platforms was the profile picture, so we decided to start development of integration of an Imgur API early on, in about the the third week of development, all the way until at least the 8th week, coming back to it often, trying our absolute best to get it working, but it never ended up working, leaving us with a huge issue in our overall program, which is that there are no images being uploaded or stored in the database, whatsoever. We had the idea of just storing the web link in the SQL, and being able to upload an image to a server/folder/drive available on Imgur itself through our website, and store the link of it as an attribute for any account, but the uploading never worked, and we ended up scrapping the idea completely before the latest release, and it currently does not work.

Profile Editing:

For some reason, in the editProfile page, when we would change the name of any character, once in a while, after clicking 'Done' to upload our changes, it would cause an issue. One of the reasons was related to the name actually needing to be changed to two separate words (separated by a space) in order to be split appropriately into our database, which has required columns for First and Last name, which were ungracefully causing exceptions, but sometimes it would happen despite having a first and last name anyway. This issue is still prevalent, and difficult to recreate, despite hours of attempts at recreating it, and analyzing.

Follow on VisitProfile:

When visiting somebody else's profile, and attempting to follow or unfollow them, on occasion, there would be an ungracefully handled exception that popped up when clicking the button. This issue was not able to be recreated, and we never found out what caused this issue once in a while.

2 Waiting Room

Messaging System

A significant part of the original idea that was developed in Spring 2021 was a peer-to-peer message system as an informal medium to communicate with one another. In our release, we did not implement a working prototype for messaging

between users, and would like to see it in the future. A simple client-server model can be implemented to host messaging between friends and groups.

Organization Events

In our implementation, we have the ability to create organizations, however, due to errors in the object design we implemented, they have limited functionality compared to the student. The next large step would be for organizations to be able to create and host events that show up on the calendar of users that follow the respective organization. This would also give the calendar some depth as well.

Discussions

In our final release, we have a discussions page for classes, however, it is merely a prototype/template of what can be when implemented properly. Firstly, the associated classes for each user need to be stored in the database, and the object design for the discussion board also needs to be started so the discussions can be fully functional and responsive.

3 Ideas for Solutions

Images:

Reapproaching the Imgur API would be beneficial to see how we could use it to upload images directly through the website using the EditProfile and account creation pages to upload and store profile banners and profile pictures, as well as images inside of posts and discussion threads. We know that it can be used inside of applications with our tech stack so, it is definitely one of the better ideas when it comes to staying free and efficient.

Messaging System:

Using some sort of server-client system would be beneficial for carrying out this messaging system idea, as it is one of the larger parts of the original idea developed by the original creators of the UniConnect idea, allowing for peer-to-peer messaging, as mentioned in the last section. Since we are using Java for the back-end logic, it wouldn't be very difficult to integrate within our existing application to actually finish all of the core ideas of this project!

Organizations:

Organizations are something that were largely left unexplored in our completion of this project, as we focused on the student-side of the application, but it's implementation would be very similar to the students' (in terms of the back-end), with some slightly different attributes, but most of the functionalities would be very similar, except that they would be linked to certain events, also as mentioned in the last section, that can allow for people to plan things like class assignments, meetings, tests, and plenty of other things to place onto the calendar that's available on the homepage. It would also allow for further exploration of the discussions section of the website, as organizations

could be considered their own “classes” the way classes are treated on other discussion-type websites like Piazza, allowing for more formal conversations and Q&A/forum communication relating to specific organizations or classes to occur.

4 Project Retrospective

Looking back at this project, we have learned a lot of things. We initially started with HTML, CSS, and plain JavaScript as our tech stack, however this changed later on as we determined the scale of this project and the requirements for a backend. We ultimately decided on React.js in the frontend and Spring Boot (Java) in the backend along with MySQL for our database, following a typical MVC architecture.

However, none of us were experienced in this sort of tech stack, let alone how they would communicate with each other, so a steep learning curve set us back in terms of functionality for our website. Dividing our team into frontend and backend helped us get ahead of the curve by allowing individual members to focus on their responsibility alone.

In reflection, we could have picked a simpler tech stack, such as using Node.js or Django as our backend framework, as Java proved to be complex and required an additional IDE to work with. The object design for our backend is also a little complicated and could be simplified to improve code readability and overall efficiency. Focusing on functionality is something that we would have also focused more on, rather than trying to develop a professional application, although the learning experience has been great.

VII Glossary

React.js: React is a free and open-source front-end JavaScript library for building user interfaces based on components.

Piazza: A website where students can have formal discussions about their course with other students, TAs, and Professors.

Spring Boot: A framework for Java that provides an easy way to just “run” and application while keeping production at a professional level. Offered us ways for the front-end to communicate with the database.

Rest APIs: When a client request is made via a RESTful API, it transfers a representation of the state of the resource to the requester or endpoint. This information, or representation, is delivered in one of several formats via HTTP: JSON (Javascript Object Notation), HTML, XLT, Python, PHP, or plain text. JSON is the most generally popular file format to use because, despite its name, it’s language-agnostic, as well as readable by both humans and machines.

MVC: Stands for model-view-controller, defines a software architecture that our application is partitioned into. Model is usually a database, a view is what the user can see and interact with, and the controller controls features on the UI and communicates with the model.

VIII References / Bibliography

[1] Robertson and Robertson, Mastering the Requirements Process.

[2] A. Silberschatz, P. B. Galvin and G. Gagne, Operating System Concepts, Ninth ed., Wiley, 2013.

[3] J. Bell, "Underwater Archaeological Survey Report Template: A Sample Document for Generating Consistent Professional Reports," Underwater Archaeological Society of Chicago, Chicago, 2012.

[4] M. Fowler, UML Distilled, Third Edition, Boston: Pearson Education, 2004.

[5] R.Awan, A.Padman, O.Rotkis, A.Saxena, UniConnect Project Report: CS 440, Spring 2021

IX Index