Part A Project Proposal

Project Name: BruinNotes

Team Name: DJ-JAYS

Team Members: Jinwoo Baik (705378164), Dylan Phe (505834475), Smayra Ramesh (905206685), Aiqi You (405557435), Jack Zhao (405377811), Yunfan Zhong (605313390)

Github: https://github.com/dylanphe/cs130project

1. Motivation

For students, having good notes to work and study off of is an extremely important part of being successful in classes. However, there is currently no centralized location for students to compare notes with each other, find notes from their TAs and professors, or share study guides easily. Although Bruinlearn allows professors to post their lesson notes online, students often have to sift through many unrelated videos, supplementary materials, and assignments to find the notes that they are looking for. Additionally, not all professors consistently post their notes, and Bruinlearn does not support students sharing their notes with other students in the class. This means that students who miss class due to sickness or an emergency have to reach out to the professor, or have friends in the class they can ask for notes.

BruinNotes will solve all the above issues and more by providing a website that acts as a centralized platform for UCLA students to share, find, and work on notes with their classmates. After creating an account, a user can search for and add UCLA courses, and have a place to request and share notes. Users will also be able to like notes that they find helpful, and also leave comments or questions underneath them. On top of it being a space of sharing, it will also serve as an archive of digital notes for all UCLA courses. Ultimately, BruinNotes aims to provide a simple but featureful experience for its users, and to give students the tools they need to be successful. Sharing is caring.

2. Feature Description and Requirements

2.1 Proposed Features

After creating an account on BruinNotes, users can search for and add their classes to the site. Then, they are able to share, like and dislike, comment on, and request notes. Each feature adds a unique and vital function to BruinNotes.

2.1.1 Account System

Each user of BruinNotes is associated with an account. An account is required to access classes and notes on BruinNotes. A first-time user can create an account with a UCLA email address, full name, username, and password. User accounts are associated with their notes, likes, and comments.

2.1.2 Courses

Notes are located within each course page. In BruinNotes, a course can be searched by the abbreviated course catalog number such as “MATH 131A”. Since a course may be offered across several terms and professors, each course consists of a list of course offerings distinguished by terms and professors. Users can then select a particular class offering to view its notes. Because each professor has their own teaching style, and different quarters can have different content, viewing notes in their own class offering ensures students get accurate information about their class.

If a class is not found on the search page, users can then create the class with the abbreviated catalog name. Similarly, if a class offering from a particular professor and quarter is not listed, users can add that quarter to the class page.

2.1.3 Sharing and Requesting Notes

A major feature of BurinNotes is note-sharing. Users share their notes by providing a hyperlink to the notes, such as a link to a Google Doc, a file located in a cloud drive, etc. They will also come up with a name for their link at this time. Additionally, users would need to select an author type, indicating whether they are a student, TA, or

professor. This allows the course page to be organized in such a way that can be easily understood and navigated. For example, students who are searching for discussion notes on week 3 can look for the notes published by a TA and having “week 3” in the title.

Students may sometimes need a specific set of notes that haven’t been uploaded, and other times they might forget to upload a promised note during a hectic week. This is where the “Request Note” feature comes in. Users can leave a request message, and after the requested notes are uploaded, the message can then be resolved. A usage scenario for this feature is if a student misses the discussion on solving midterm practice problems, and there are no notes available yet. The student can then use the request feature to get these notes from their classmates on BruinNotes. When the other students see the request message, they can upload the notes immediately.

2.1.4 Likes, Dislikes, Comments and Reports

Unlike shared cloud storage products such as Google Drive, BruinNotes enables users to interact with notes through likes, dislikes, and comments. These interactions are shown under the note link. Like and dislike counts help students find good notes among several notes. Comments can be made for questions about the notes, clarifications, and expressing gratitude for a helpful note. A usage scenario is if a student uploads their notes that were taken when they sat in the back of the classroom. Later, a comment points out there is an error in an equation in the note that would lead to serious mistakes in an upcoming homework assignment.

As all users are permitted to upload notes and indicate their roles in the class, unwanted mistakes may occur. The report feature enables users to notify potentially incorrect or suspicious information. Upon receiving a report, the BruinNotes team will review the classes and notes in question and act on them if necessary.

2.2 Use Case Diagram

The case diagram on the next page shows the functionalities that a user can do and how one can interact with others.

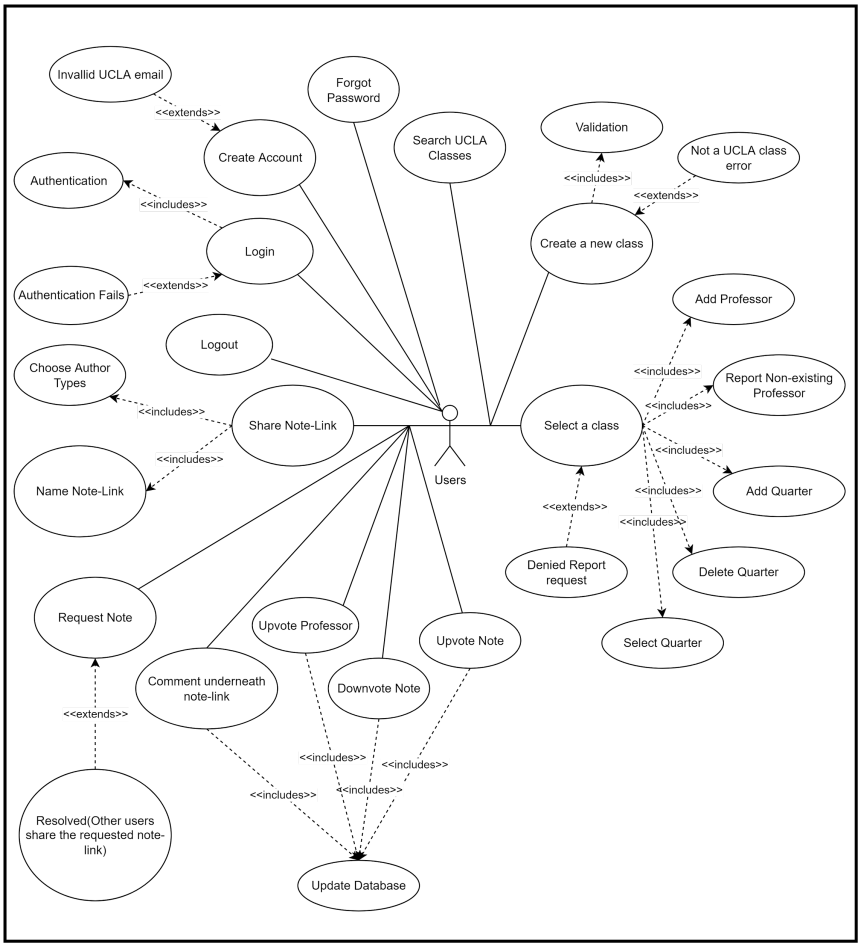


Figure 1. Use Case Diagram of BruinNotes

2.2 Mockups

Below is a detailed description of user interaction scenarios. The mockups are wireframes showing the planned layout of each page.

1. First, create an account or log in with a UCLA email address, full name, email and password.

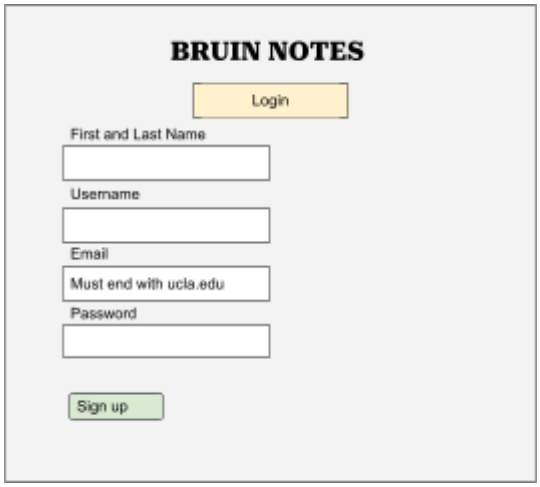


Figure 2: Sign Up Page of Bruinnotes

2. Users are then directed to the home page, where they can either search for an existing class or add a new one if needed.

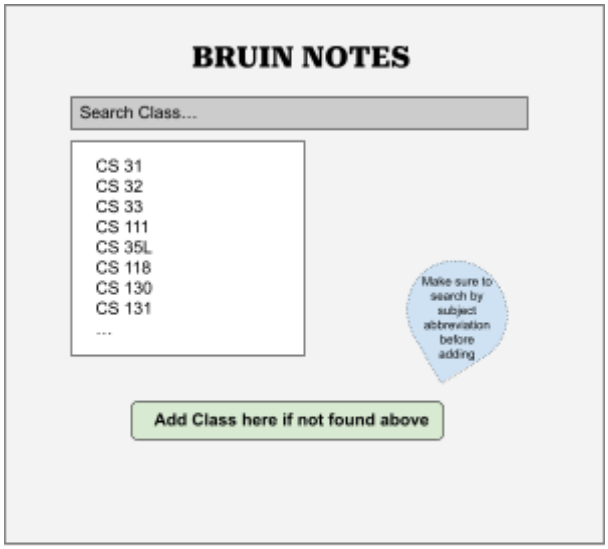


Figure 3: Welcome page of Bruinnotes

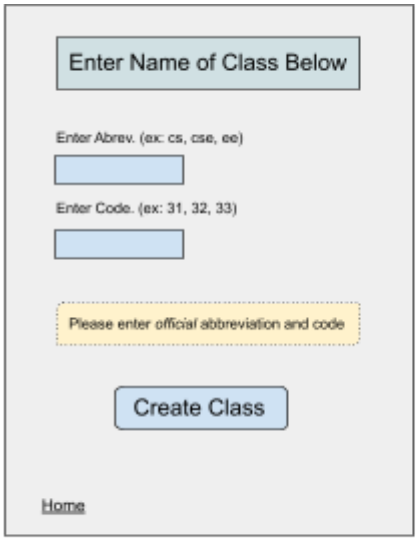


Figure 4: Add Class page of Bruinnotes

A usage scenario for this feature is if a student is taking a new class, such as a fiat lux that has not been offered in previous quarters. If the class has not been added to BruinNote yet, a student can add it themselves. They would be able to input the name of the class, which will then be added to the official class list. Any future students looking for the same class will be able to simply search for it, and find the class in the existing list.

3. If a class needs to be added, the user adds the class into the "Add Class" page using the official abbreviation and class code.
4. From there, the user will be routed back to the home screen to choose the correct class from the updated drop down of classes.
5. Once a class is clicked, the user is taken to the main class page, which is shown in Figure 5 below. Here, they can select an existing professor and quarter or add a new one if needed.

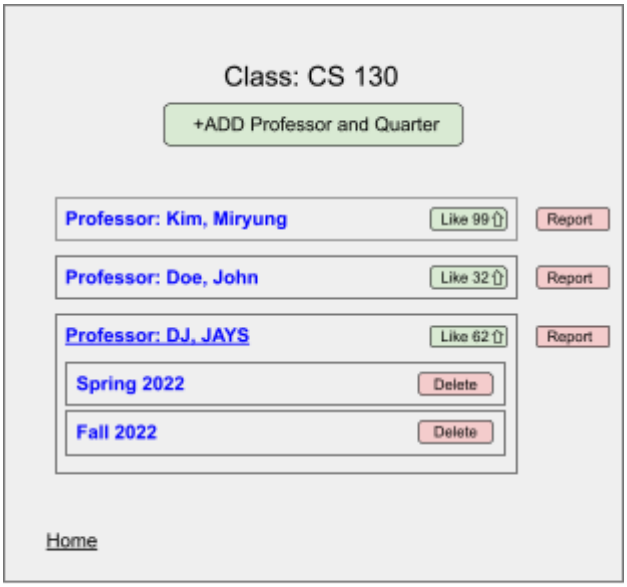


Figure 5: Course page of Bruinnotes

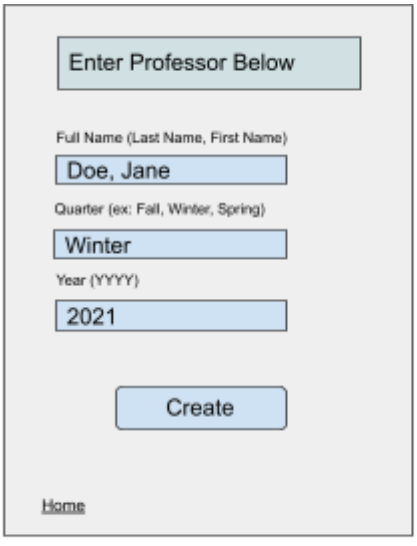


Figure 6: Add Professor and Quarter

A usage scenario for adding a new term is at the beginning of a new quarter, students or professors can add a new term to the existing class. They can put in the professor for the class, as well as the current term. The new quarter will be added to the class, and students can then share notes there for the rest of the quarter.

6. If a quarter needs to be added, users can go to the "Add Professor and Quarter" screen. Here, they will specify the professor's name, term, and year.
7. Then, the user is routed back to an updated term selection page, where they can select their newly added quarter.
8. Once the correct term is selected, the user is routed to the class's page.
    a. Here, users can share, vote on, and comment on notes.
    b. To view a set of notes, a user can click on the link and open the notes in a new tab.
9. To share notes, users can go to the "Name & Upload Link" page (Figure 8). Here, they can add a hyperlink, a name for their notes, and their author type.
    a. Specifying author type (professor, student or TA) will create a differentiation in the "officiality" of the notes.

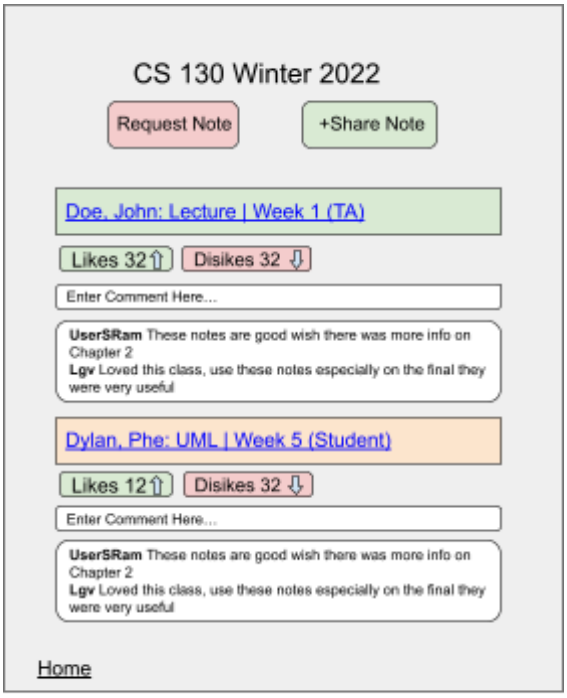


Figure 7: Course note page of Bruinnotes

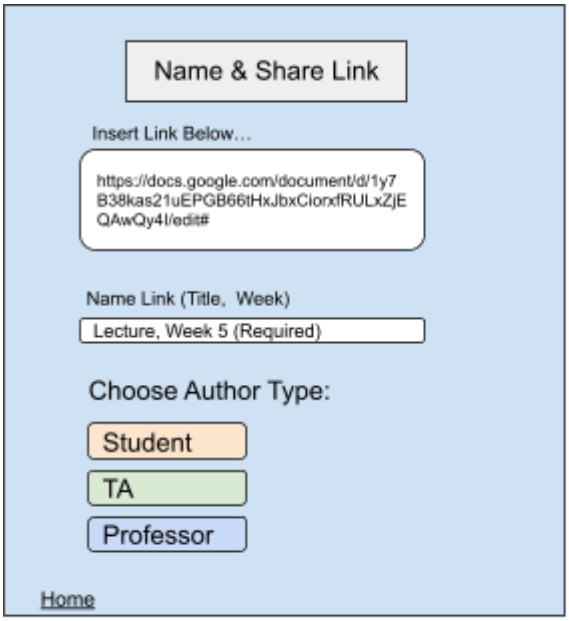


Figure 8: Share note page of Bruinnotes

A usage scenario for this feature is when a student has some notes they want to share or compare with other students. They can upload it using this page, so other students can see it. They are able to paste in a link to their notes, name the link, and indicate that they are a student.

10. If a user needs a specific set of notes, they can click on the Request Note button and fill out the request form. The request will be resolved once other users share the requested notes.

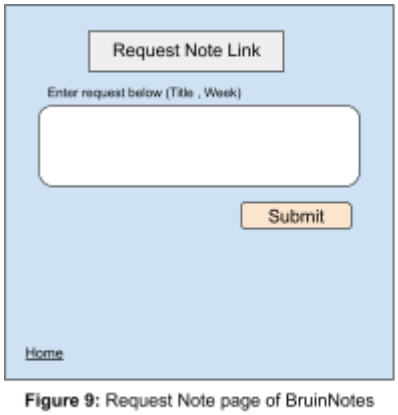


Figure 9: Request Note page of BruinNotes

3. Feasibility

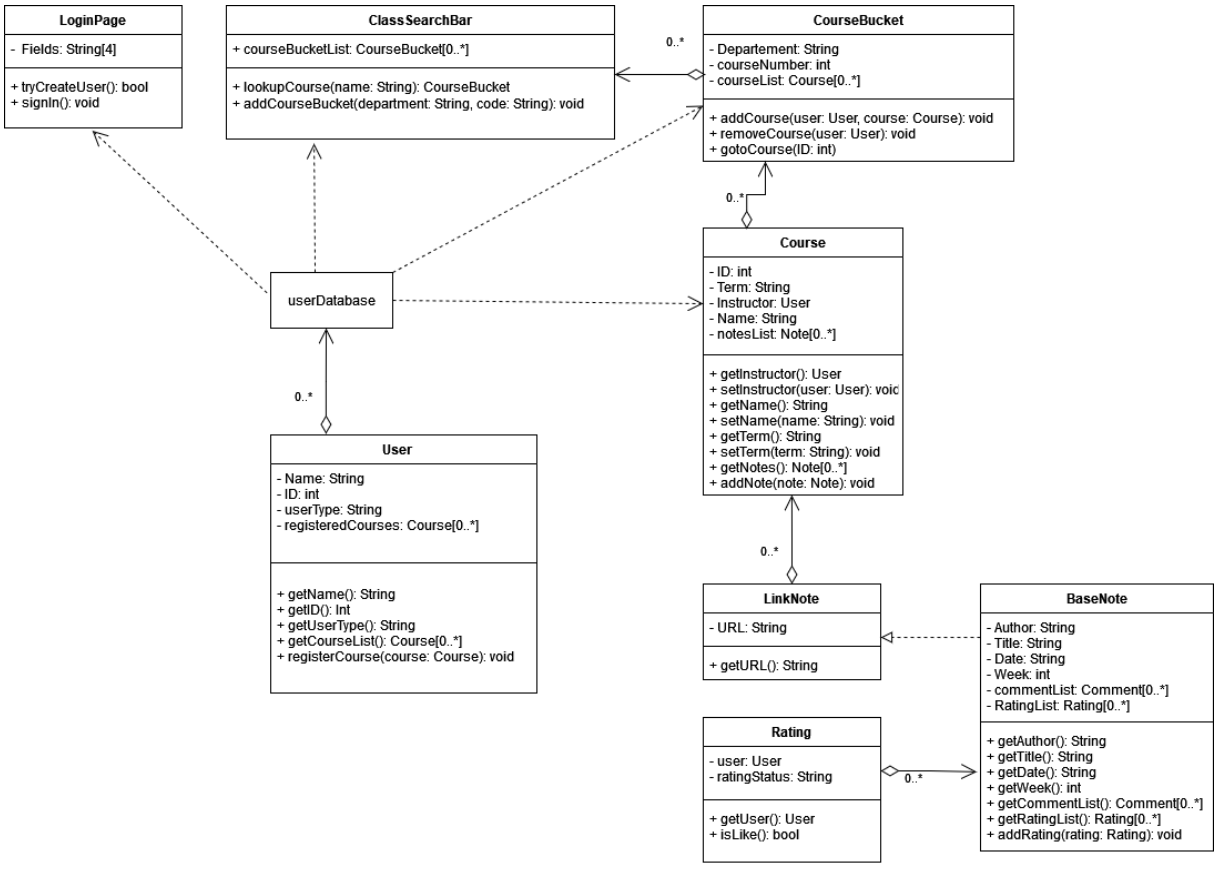
Originally, our team planned on implementing a hub for note taking, storing, and sharing. However, due to the constraints of time and space (if we chose to store all the notes within our own database), we chose to scale down our design to the current plan. Our model for the project has many separate components and pages that can be divided up and worked on in parallel. Additionally, instead of building everything from scratch, we will be utilizing existing reliable tools and frameworks. The resources we chose to use are ones that we either have experience with, or are similar to things we've used before. Finally, we distilled our design down to the most useful features, and made any additional details into stretch goals. All of this will make BruinNotes a feasible project to build in seven weeks.

3.1 Novelty

As mentioned in Section 1, we do not believe that there currently is a centralized note sharing platform for UCLA students. We aim to encourage users to have a culture of sharing notes as tools for better education. Although professors can share their notes on Bruinlearn, students are unable to easily share notes with their classmates. If a student wants to share a study guide or compare notes with another student, their only option is to try to find someone they know who is taking the class. Other than uploading and viewing notes, BruinNote also has a social aspect with the ability to like and comment on notes. All of these features combined give students a novel way to interact with their classmates and work together by sharing notes.

3.2 Preliminary Design

Library/ Framework/API	Description	Relevance to Project
React	A front-end JavaScript library for building user interfaces.	This will be used to create components to build the frontend of the website. It can also technically be used for some backend uses as it includes object-oriented programming.
Node.js	A server side JavaScript framework.	Node.js will be used to set up the backend and integrate it with the frontend of our site.
MongoDB	Database that uses JSON-like documents to store data	MongoDB will be used to store user information, and can be linked to Java to communicate to the front-end.
Passport	Middleware for Node.js that assists with authentication and authorization.	Passport will be used to help us authenticate users, and keep them logged in as they navigate BruinNotes.
UCLA Registrar API	An API that provides information on all classes offered in a selected term.	This can potentially be used to scrape all the classes for a current term and add them to BruinNotes. It requires approval from the Registrar, so our project design does not currently incorporate it. If approval is granted in time, this can replace the function of students adding their own classes.



4. Capability

Our team will be able to complete the implementation of this project by combining our diverse strengths in all aspects of full stack development. We have picked frameworks and APIs that we either have extensive experience with, or can be quickly learned due to a strong foundation in a similar tech stack.

Jinwoo Baik worked with backend related work in his summer internship at Amazon, where he has worked on the services and internal device architectures that powers Alexa and related devices. During this internship, he worked mostly with Java and C++. He also has a good amount of experience with Python, although in a more research-oriented context, but it should be enough to implement some of the design details in the backend should it be necessary. He was involved in the design process for this project, and aims to work mostly in the backend side of the project by working on implementing the interactions of all the different components of the project. For the Part A proposal, he worked on the UML class diagram and the presentation slides.

Dylan Phe has experience with full-stack development as both a frontend and backend developer in his past projects. Last spring, he teamed up and successfully implemented a mock version of wordle that provides additional levels of difficulty and a scoreboard. In that project, he utilized HTML, CSS, Typescript, Javascript, and React with NodeJS server and MongoDB for the implementation. He was also heavily involved in the design of both UI/UX and the client-server of the game. For this project, he has proposed the idea and worked together with the team on the design and the report. He will mainly focus on the front-end side of the app. For the Part A proposal, he worked on the UML use case diagrams, editing and adding to most sections, and project design.

Smayra Ramesh has experience in front end development. She's worked on teams as a user experience researcher and tester. In addition to that, has experience in UI design of mobile applications. For her last project, she worked as a product manager for a finance technology product. Her experience in backend has been with Python, C++, and Java Script. For the Part A proposal, she worked on creating all the wireframe mockups.

Aiqi You has experience with the frameworks used for this project. She worked on frontend development for her last project, intensively using React, CSS, and Bootstrap. For her last project, she worked with the database using MongoDB in a python application. In this project, she will work on both frontend and backend development. For the Part A proposal, she worked on the Proposed Features section.

Jack Zhao has experience in full-stack and backend infra development from previous internships. In his most recent internship, he worked on backend infra for large distributed ML training jobs on kubernetes clusters. The tech stack used includes Python, various AWS services, Airflow scheduler, and Kubernetes. In his other internship, he worked on ElasticSearch full-stack development. The tech stack involved Angular framework with Typescript and Java backend. For this project, Jack will mainly focus on backend development. For the Part A proposal, he worked on the presentation slides.

Yunfan Zhong has experience with full stack web development from previous internships and personal projects. Last summer, she used Angular (with TypeScript, HTML, and CSS) and Python to build a UI and to implement a local server for a command line tool at Google. She was also heavily involved in the project planning, and helped design the client-server architecture and the user interface. She also has experience working with SQL, Java, C++, C, React, and MongoDB. For this project, she will do both frontend and backend work, focusing on the database design and account system. For the Part A proposal, she worked on the project design, the Motivation and Feasibility sections, and editing the proposal.