hatchio

[March 09, 2021]

SW Engineering CSC648/848 Spring 2021

Section 02 | Team 03

Milestone 02

Team Information

POSITION	NAME	EMAIL
Team Lead / Github Master	Roland Lee	mlee38@mail.sfsu.edu
Front End Lead	Jose Gonzalez	jgonzalez34@mail.sfsu.edu
Back End Lead	Lyra Solomon	lsolomon3@mail.sfsu.edu
Database Manager	Aaron Singh	asingh26@mail.sfsu.edu

History Table

Milestones	Date Submitted	Date Revised
01	2/23/2021	3/2/2021
02	3/9/2021	

Table of Contents

Functional Requirements | Prioritized **2**

UI Mockups and Storyboards (high level only) **3**

Use Case #1 | Professor 3

Use Case #2 | Professor 5

Use Case #3 | Company 8

Use Case #4 | Student 12

Use Case #5 | Student 18

High level Architecture, Database Organization **22**

High Level UML Diagrams **23**

UML Component and Deployment Diagrams 24

Key Risks **25-29**

Project Management **29**

Functional Requirements | Prioritized

Priority 1 - Core Features

1. Talent Acquisition(TA) User, Employee Resource Groups(ERG), NPO(Non-profit Organizations) shall be able to find talented new graduates from the system providing an interface to query and filter profiles and other necessary information.
2. Student users shall be able to search for professors for a review.
3. Student users shall be able to find prospective employers from the system providing an interface to query and filter profiles and positions.
4. The system shall provide a web-based messaging for users to communicate with each other and with moderators.
5. Professors shall be able to rate students on a scale from 1-5 on multiple categories and enter recommendations.
6. Moderators and recent hires shall be able to rate employers on a scale of 1-5 on multiple categories as well.
7. The system shall allow students to connect to employers with ease of access by an intuitive interface.

Priority 2 - Trust Features

1. Users shall agree to the terms of agreements then be granted access to use all the services.
2. The system shall prompt the users periodically about any updates to the terms and agreements they signed initially.
3. The system shall send verification emails when registering for an (any) account.
4. Students shall be able to upload resumes and videos, demographic information, experience, and education information for their profile.
5. Moderators shall have the functionality to aggressively remove all scams.
6. The system shall deliver feedback to the admin help desk when error reports are created by visitors.

Priority - Support & Niceties

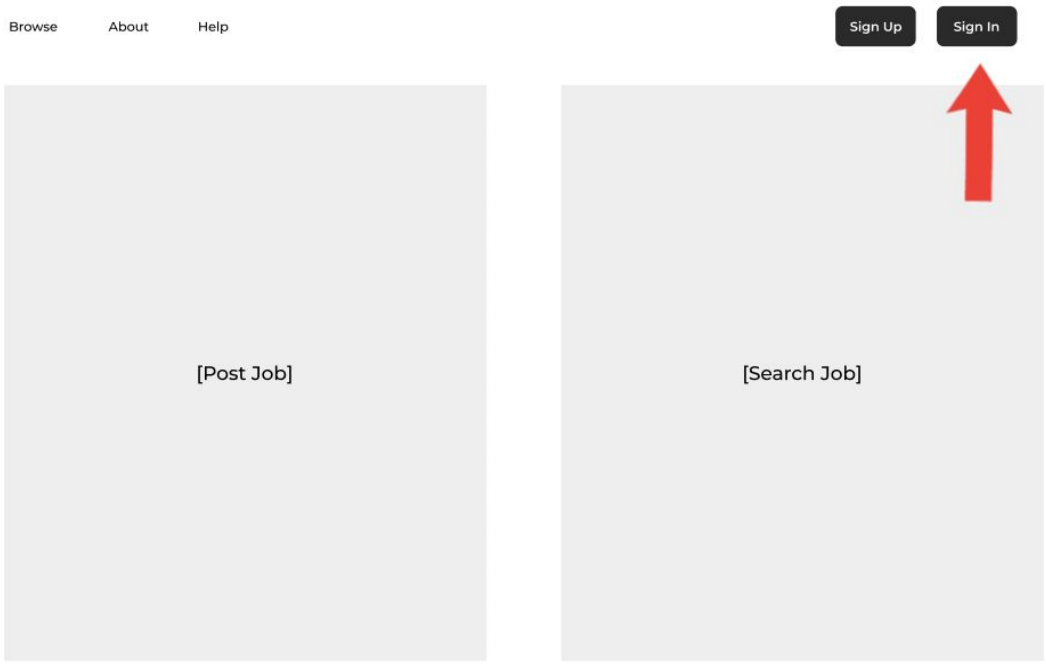
1. Administrators shall be capable of triggering the matching alerts to the companies and alerting students to get ready for interviews.
2. The developers shall take the system offline under a cyber threat attack.
3. The system shall provide multiple viewports for users to select from in order to meet everyone’s accessibility preferences.
4. The system shall periodically update back up database in order to restore information in the event of fatal crash.
5. The system shall provide a dashboard to all user types with customizable layout to personalize their page.
6. The developers shall make sure any bugs reported will be addressed within twenty-four hours.

UI Mockups and Storyboards (high level only)

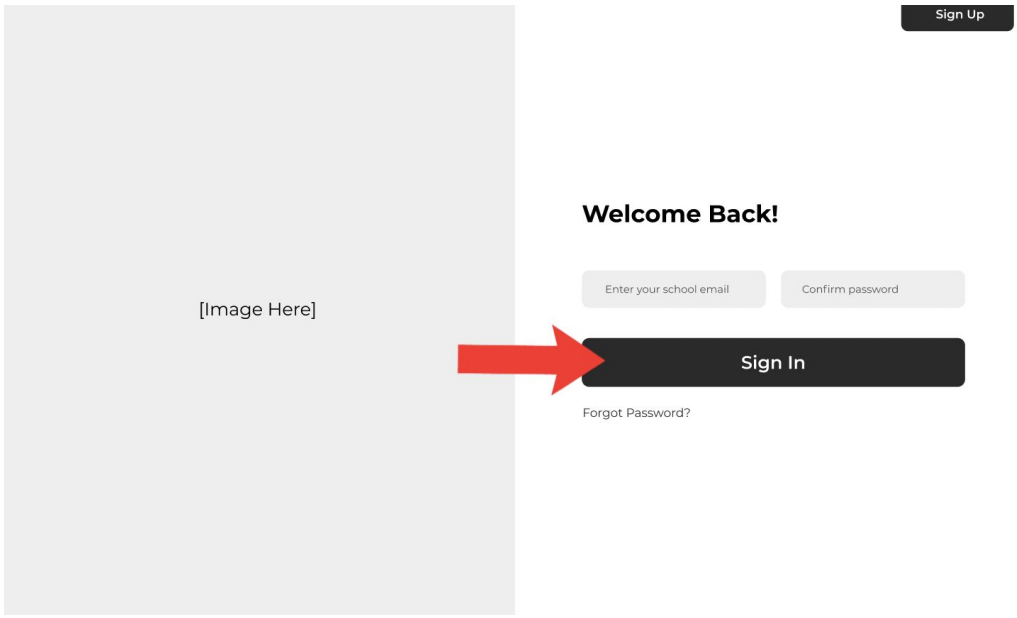
Use Case #1 | Professor

Professor: Recommending & Rating Students

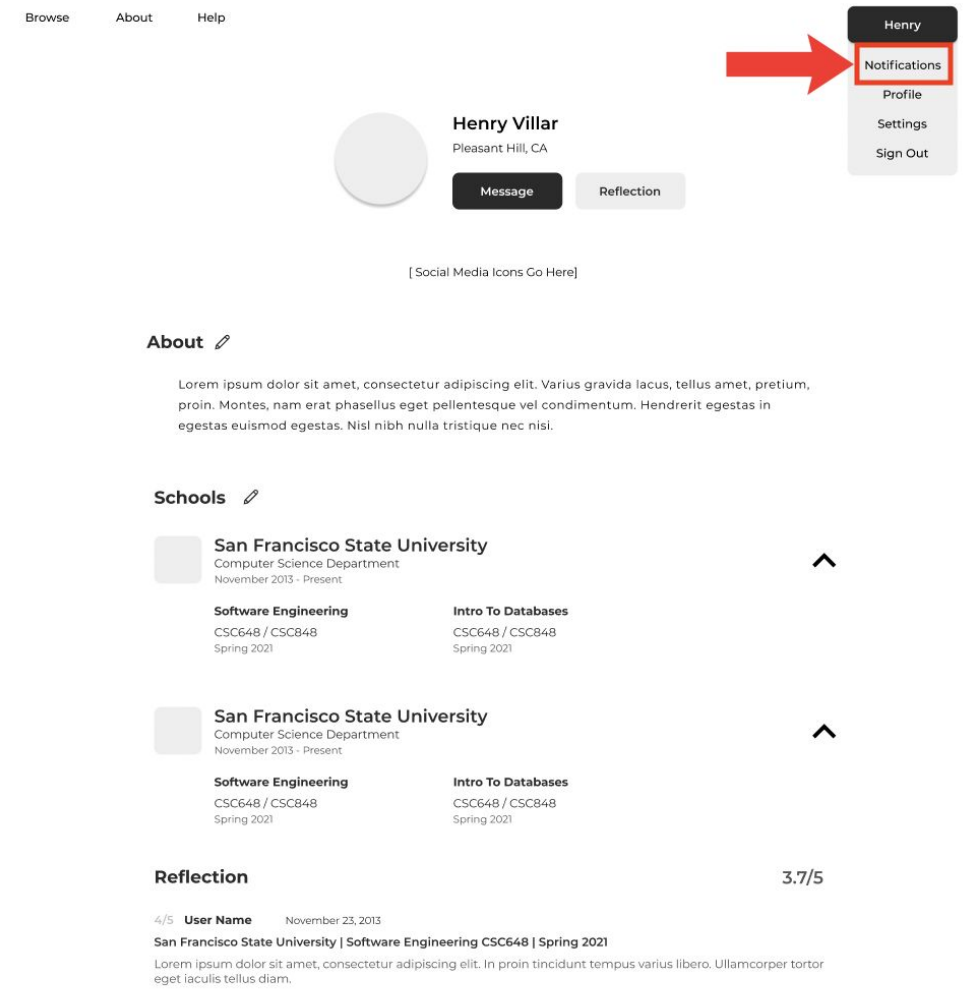
1. Landing Page



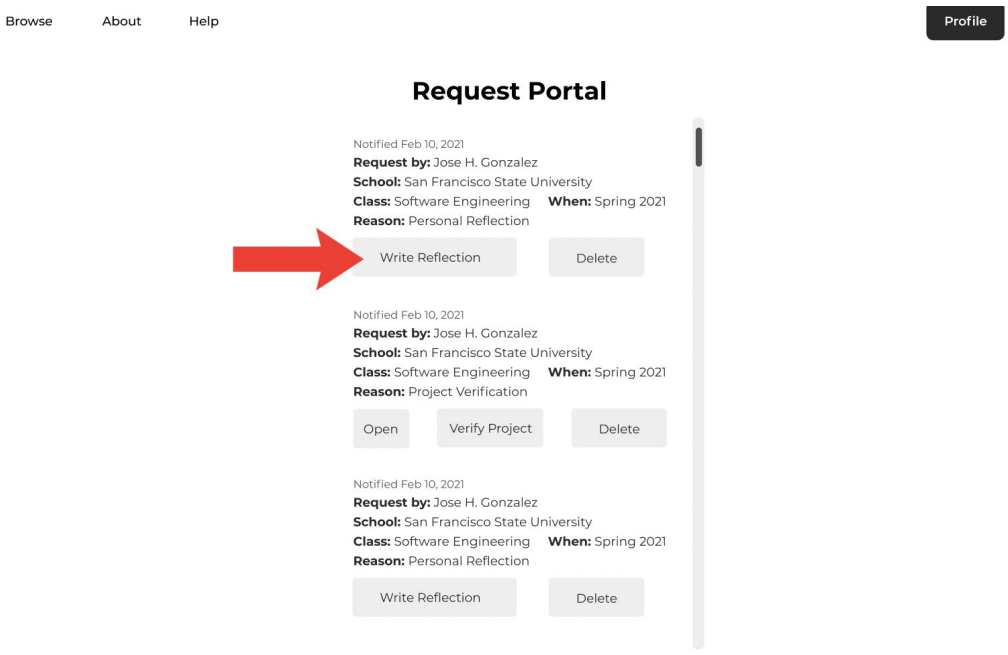
2. Login Page



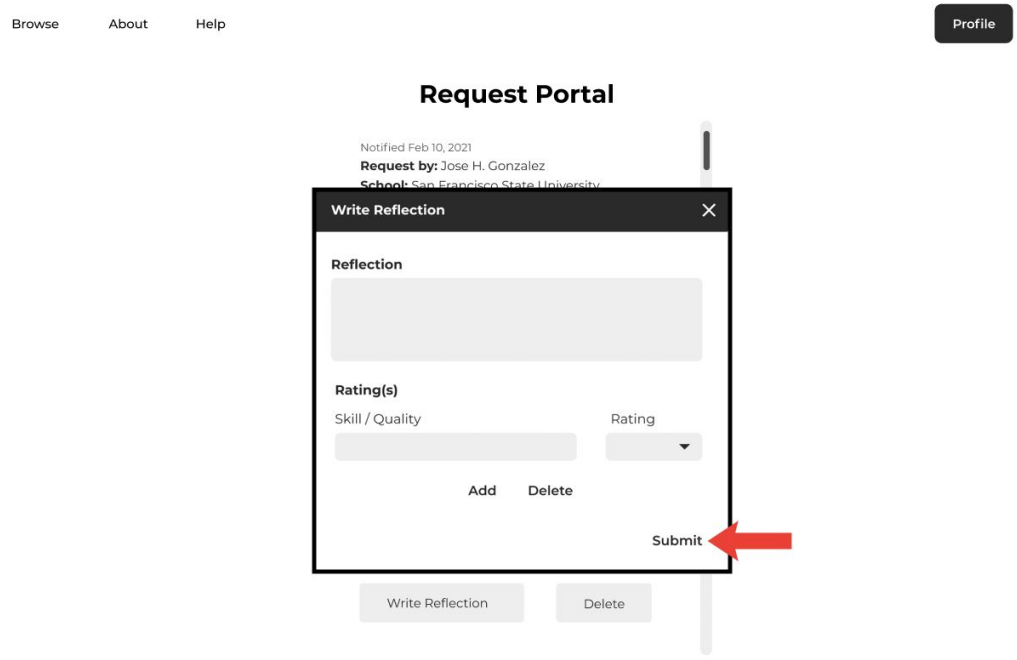
3. Professor Profile Page (redirect)



4. Notifications Page



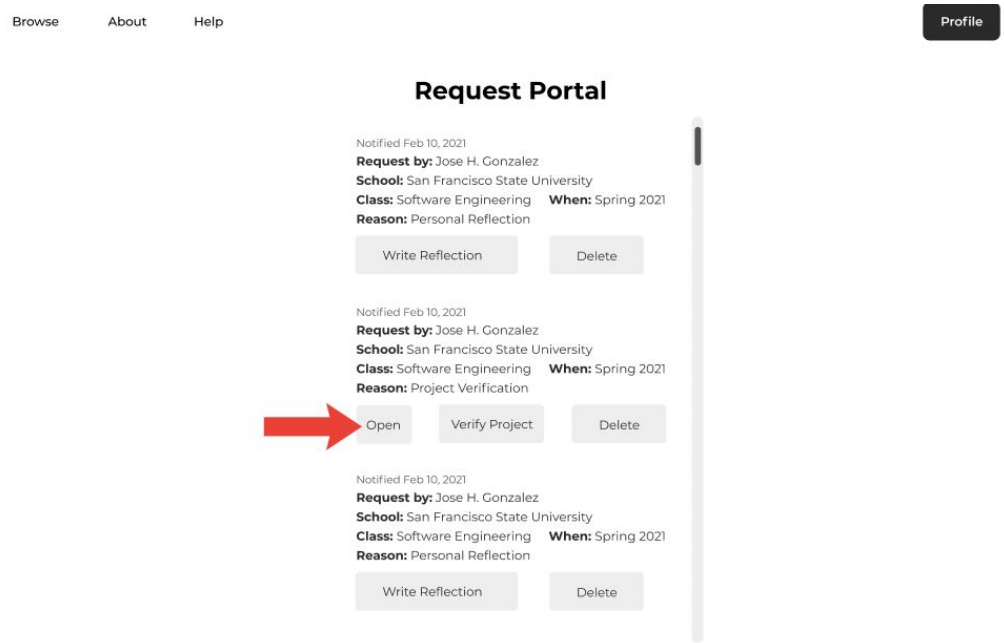
5. Notifications Page - Student Recommendation Popup



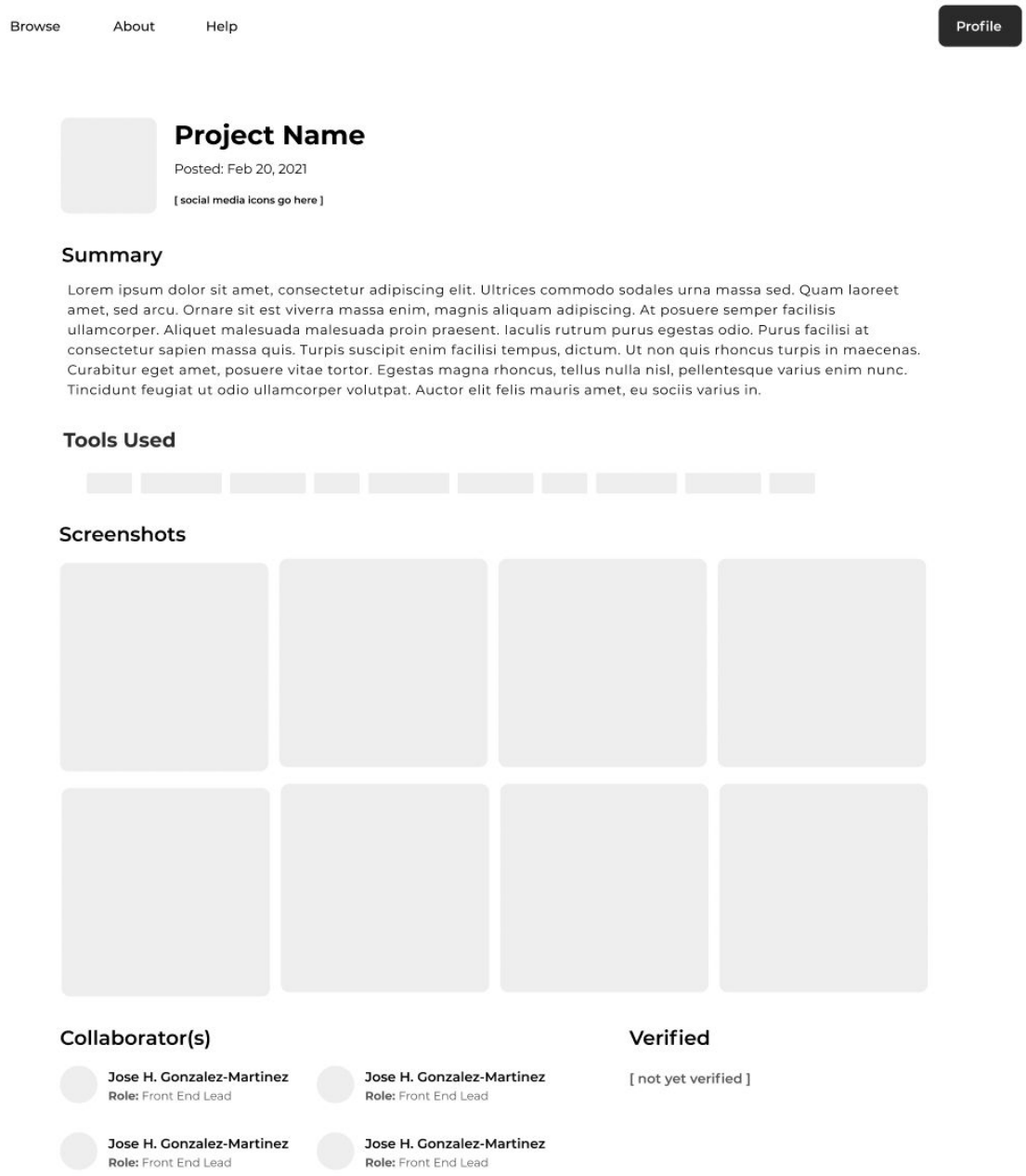
Use Case #2 | Professor

Professor: Validates Project information after looking at project details

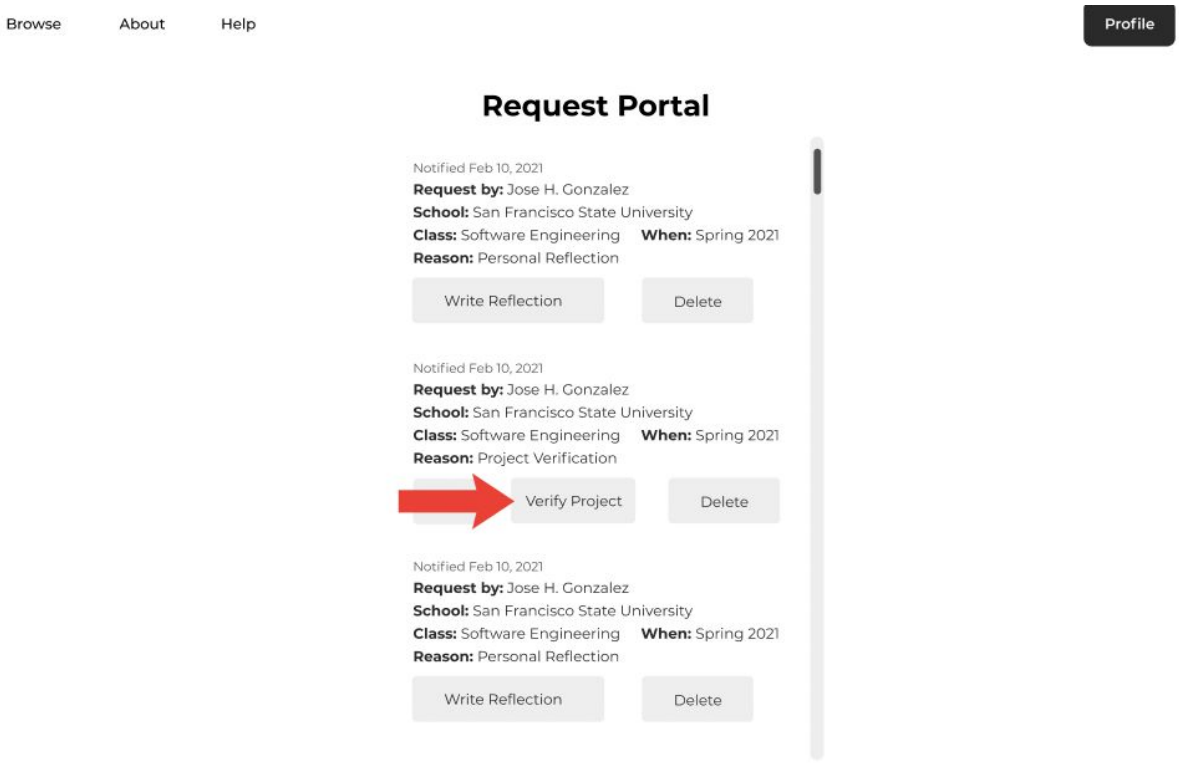
1. Notifications Page - Open Project



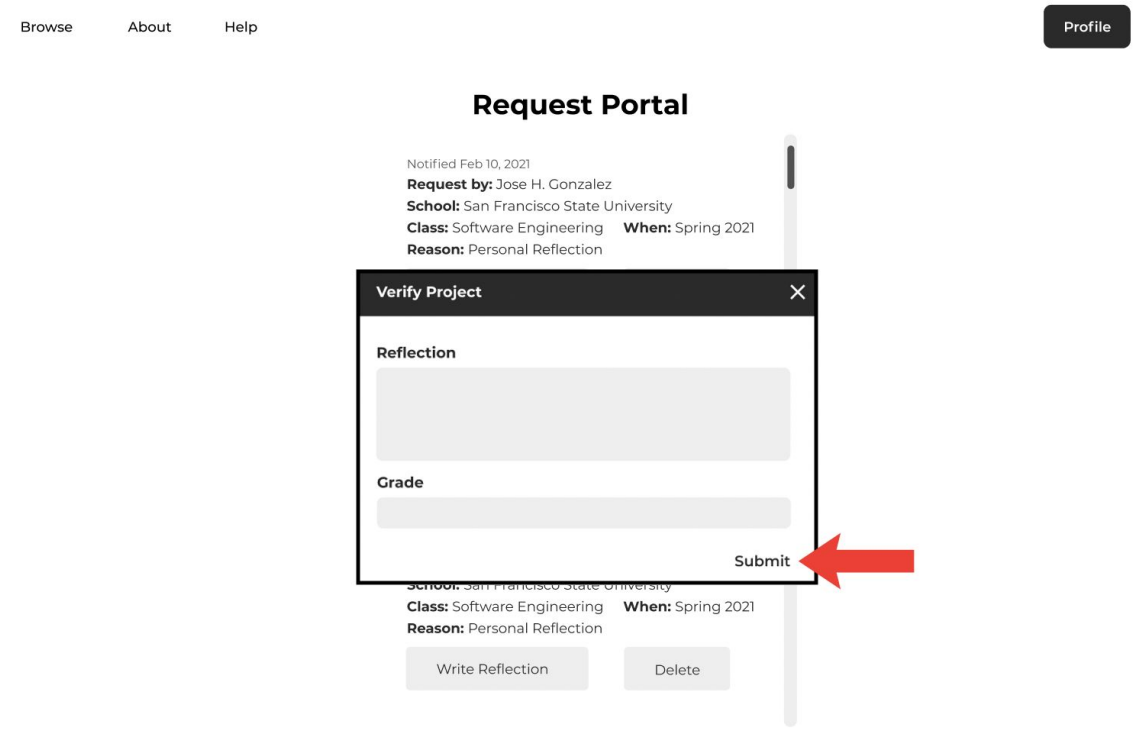
2. Project Page



3. Notifications Page - Verify Project



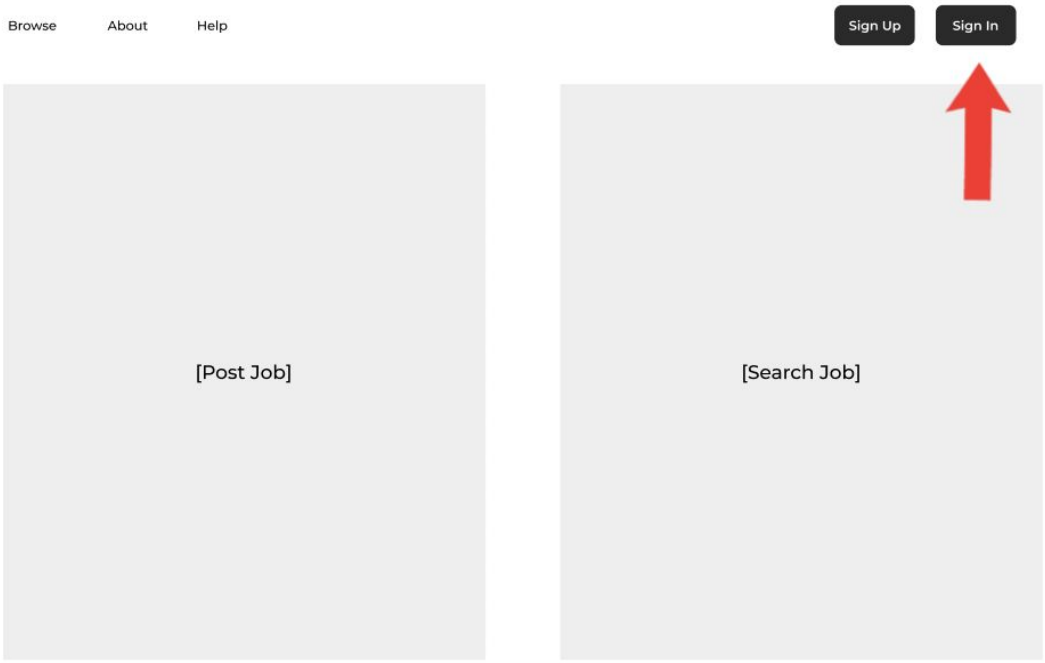
4. Notifications Page - Verify Project Popup



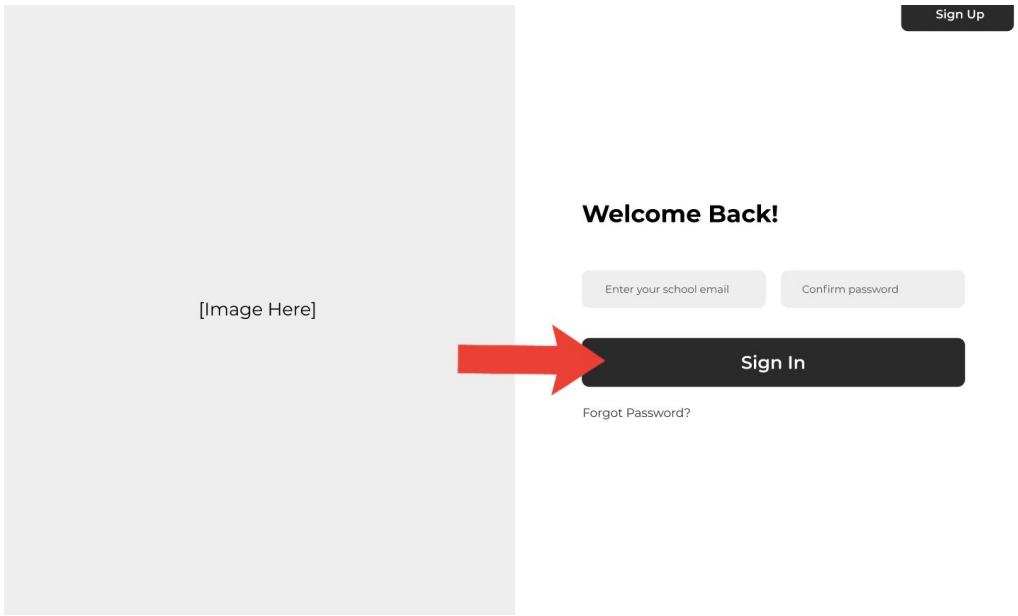
Use Case #3 | Company

Company: Searches for students based on job position needs and reaches out to them

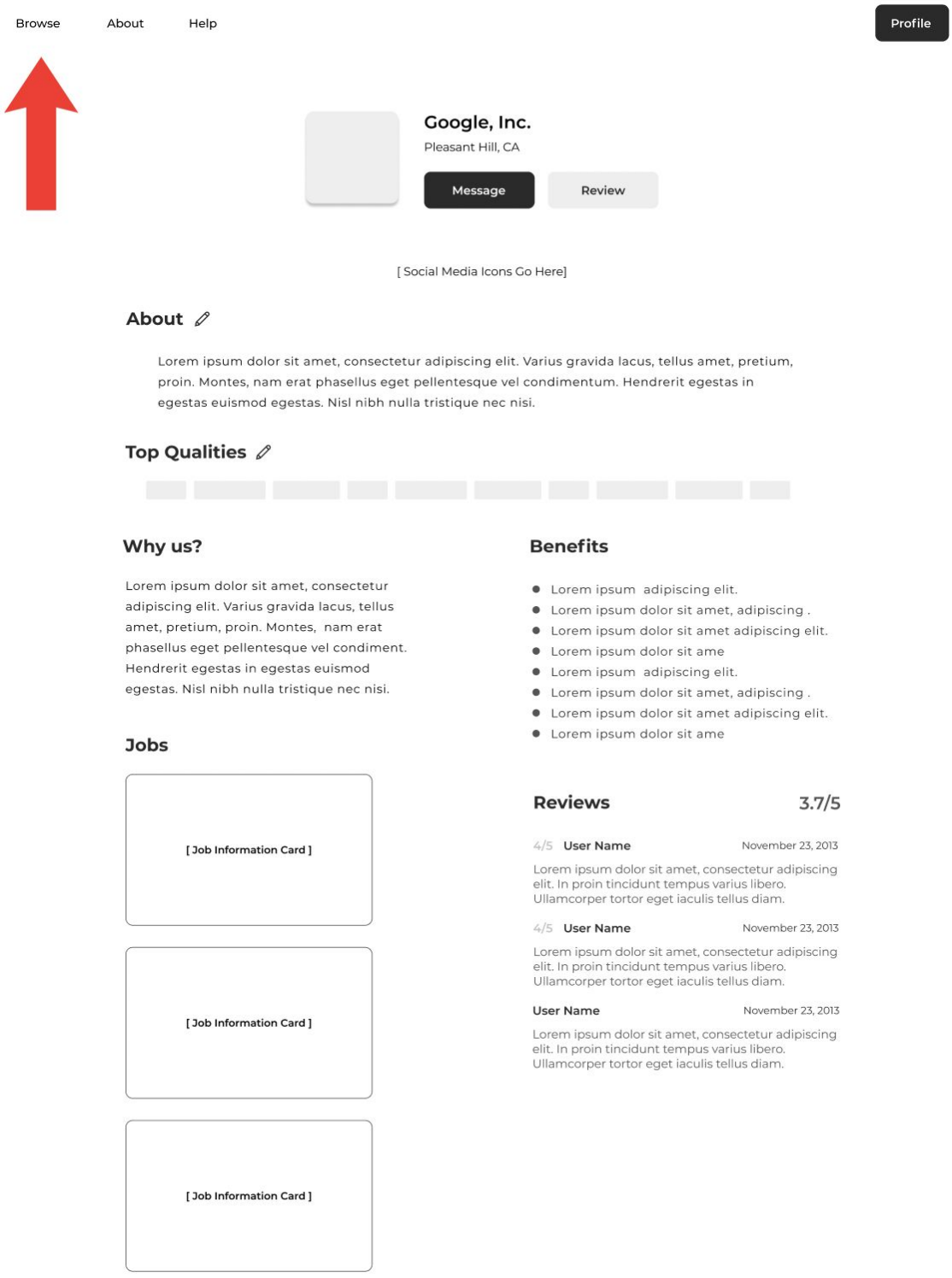
1. Landing Page



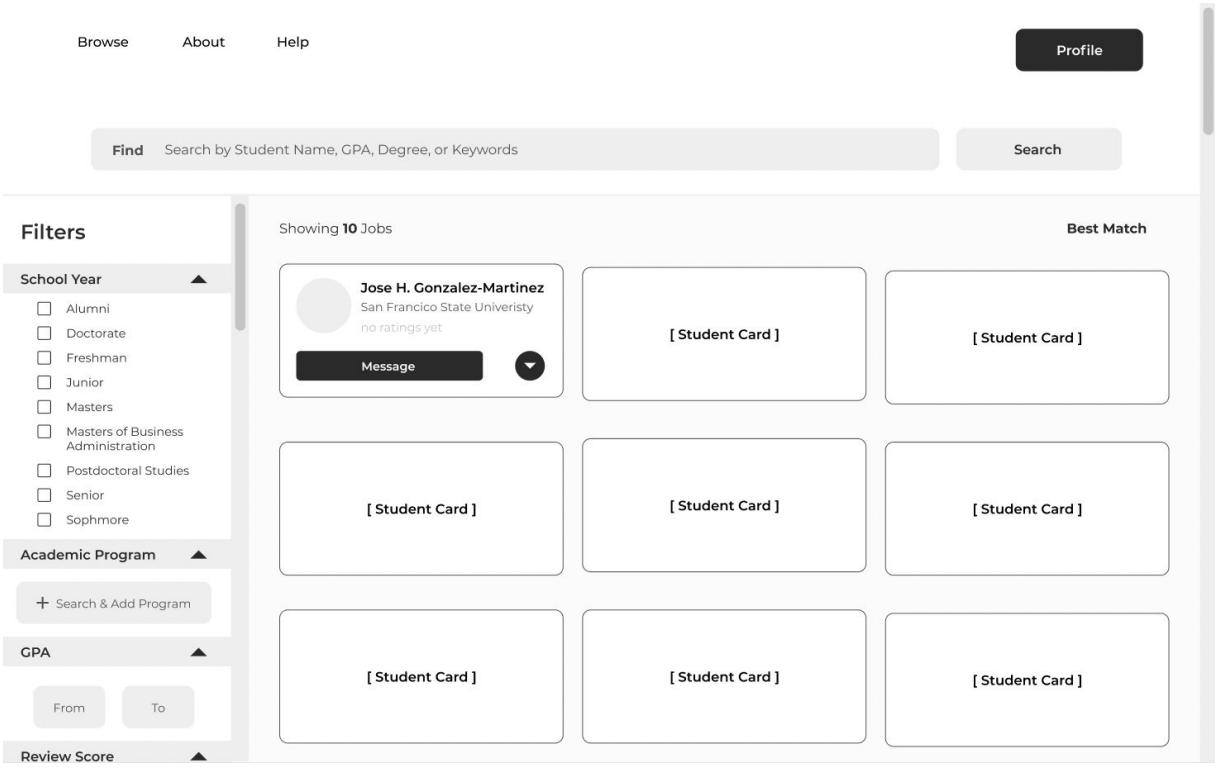
2. Login Page



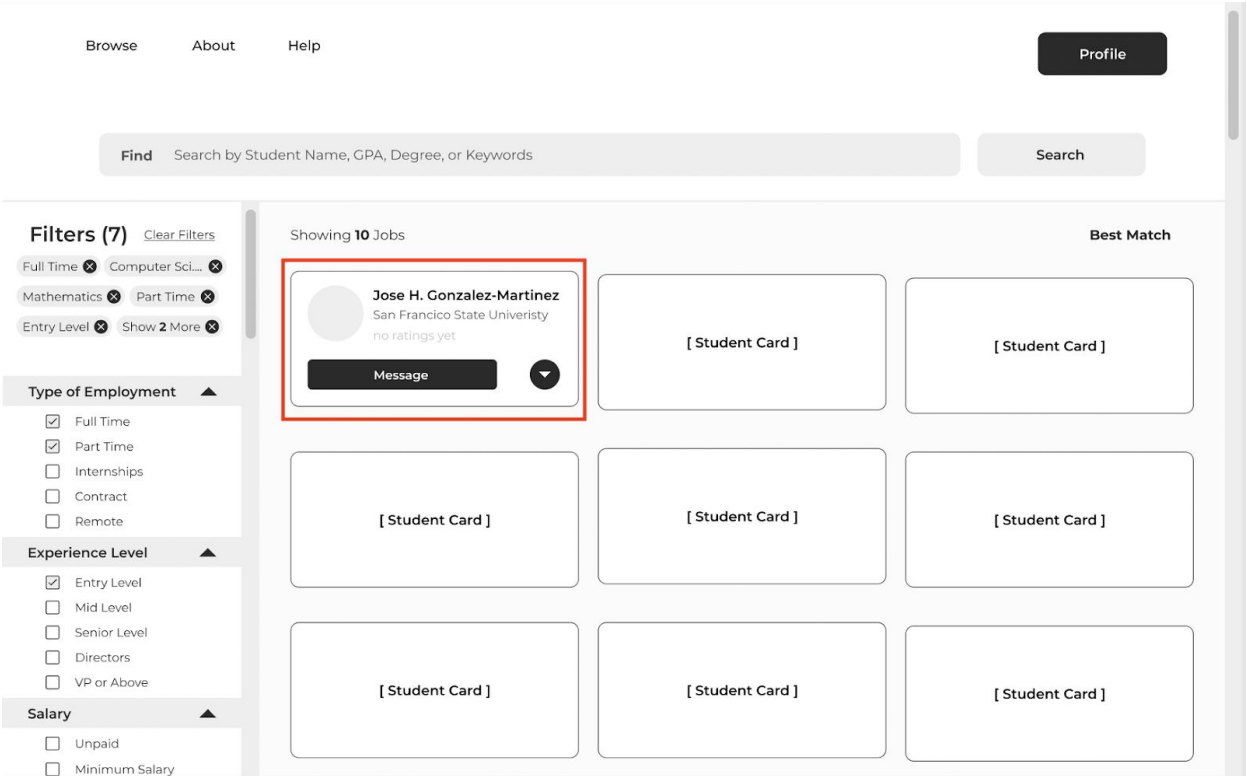
3. Company Profile (redirect)



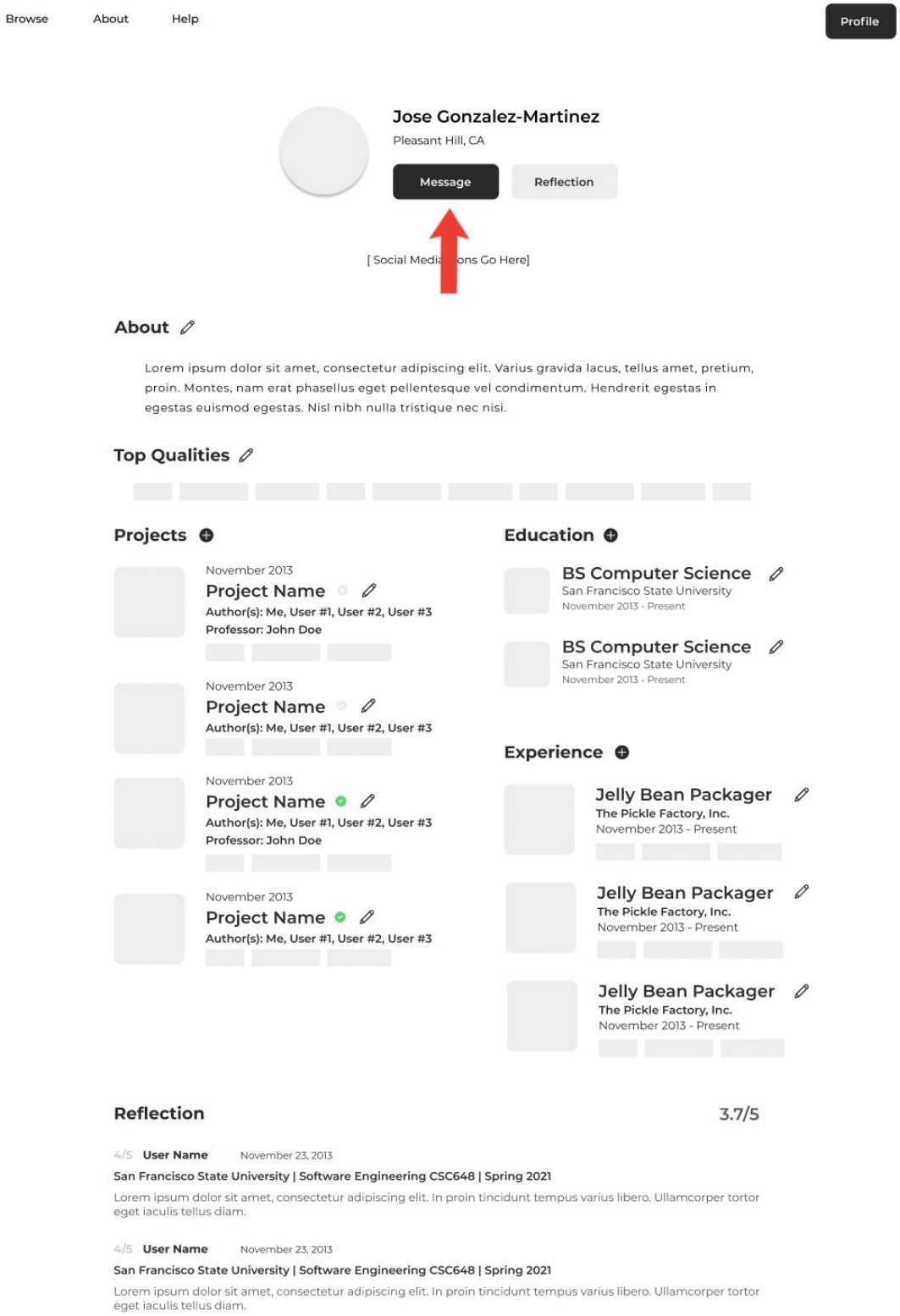
4. Student Search



5. Add Student Filters



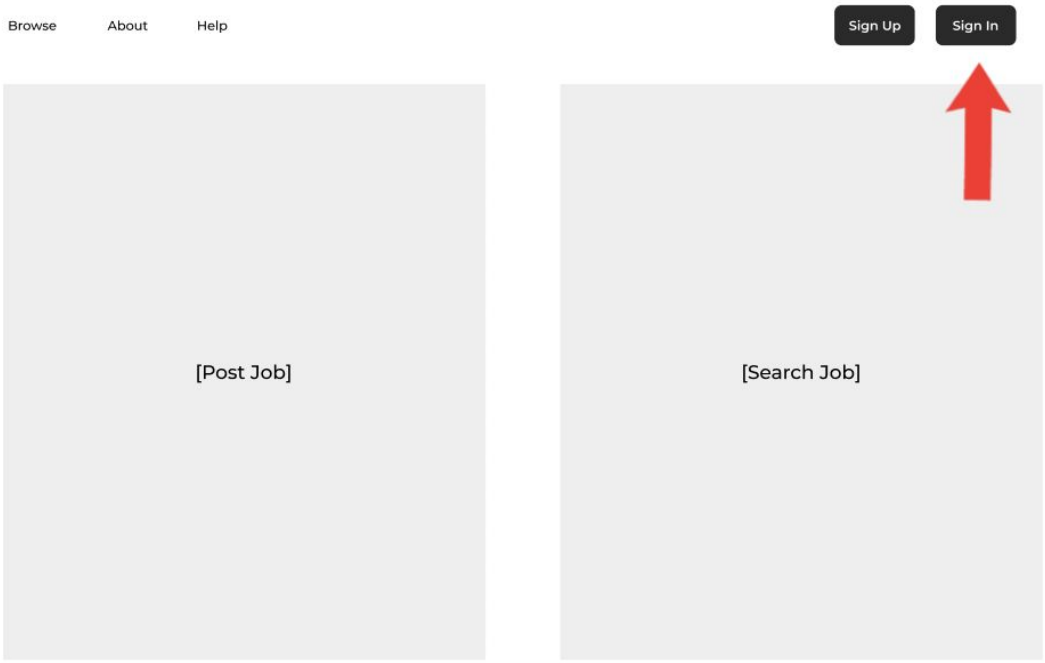
6. [Potential] Student Profile → Message



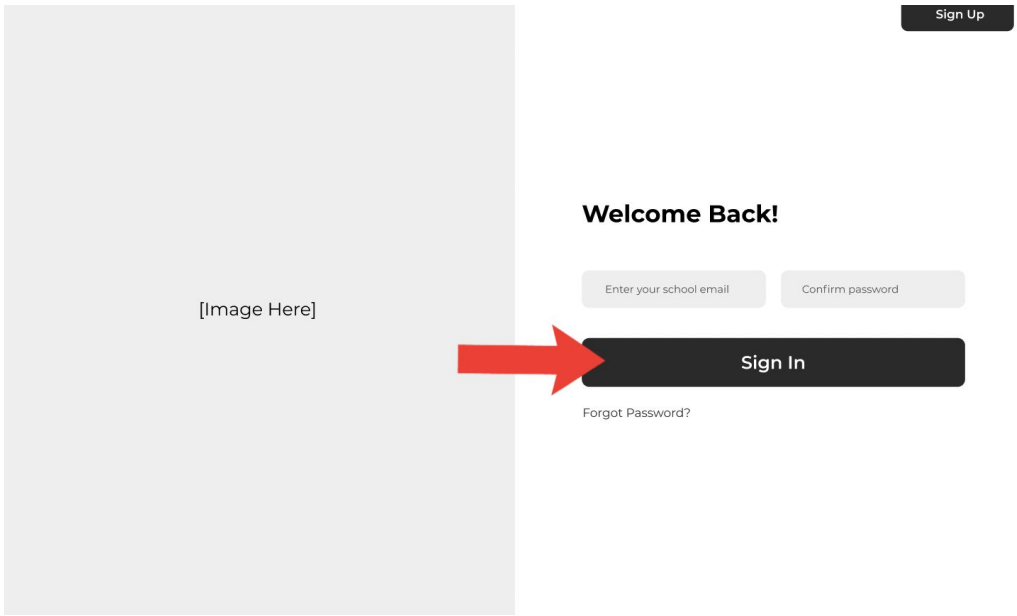
Use Case #4 | Student

Student: Looks for specific job, learns about the job position and company, and applies

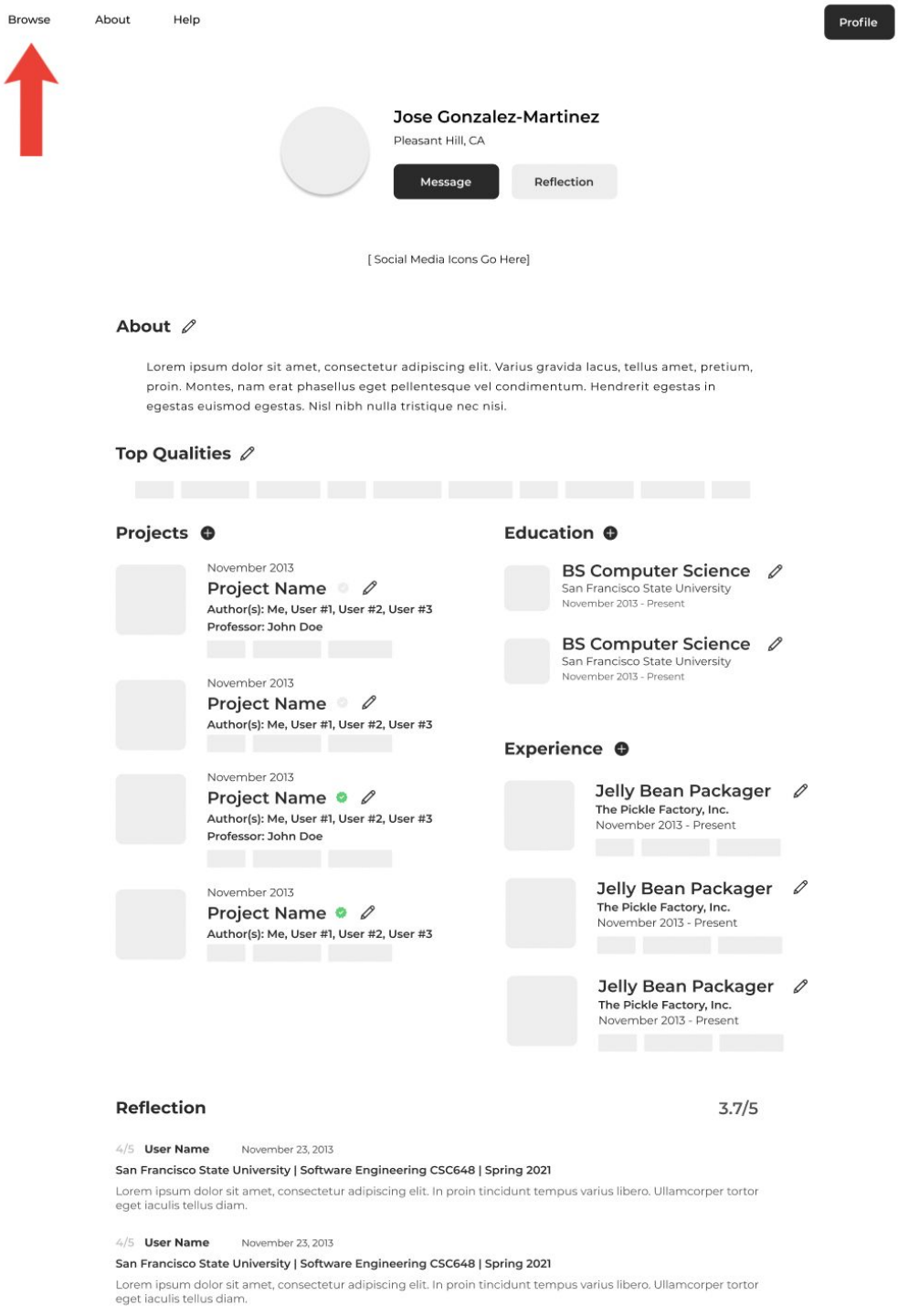
1. Landing Page



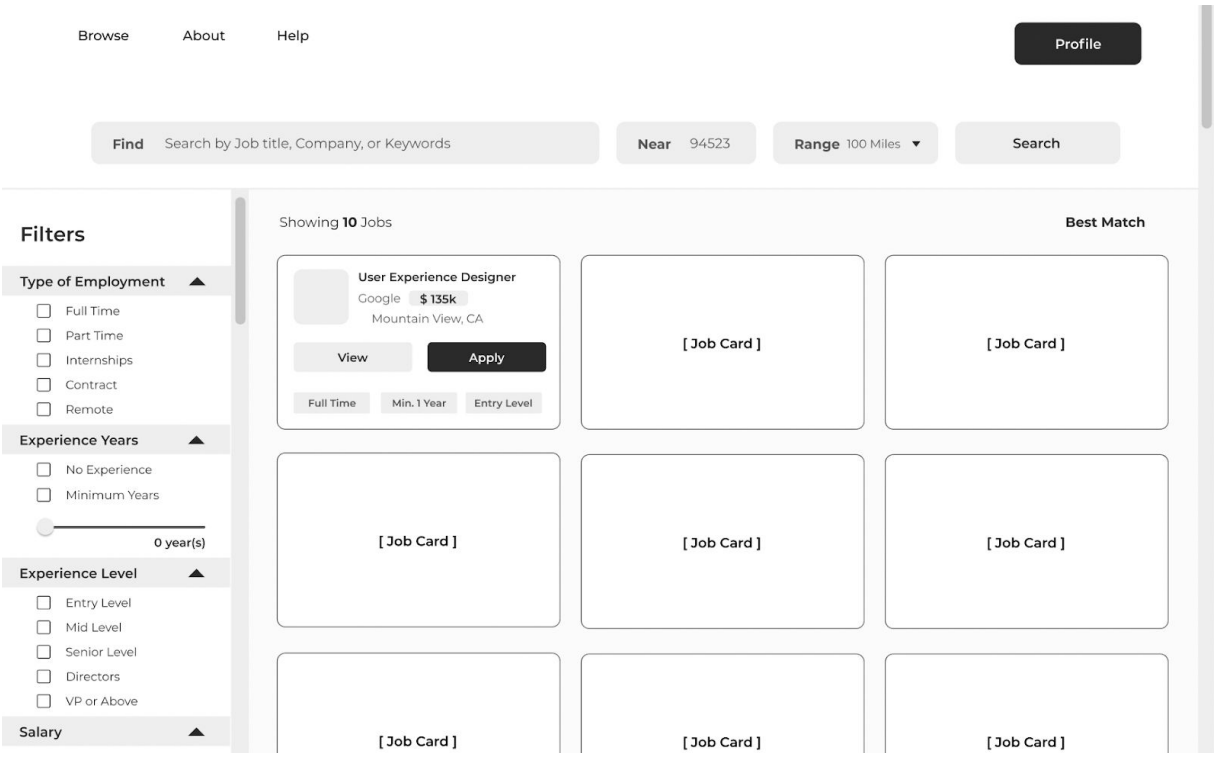
2. Login Page



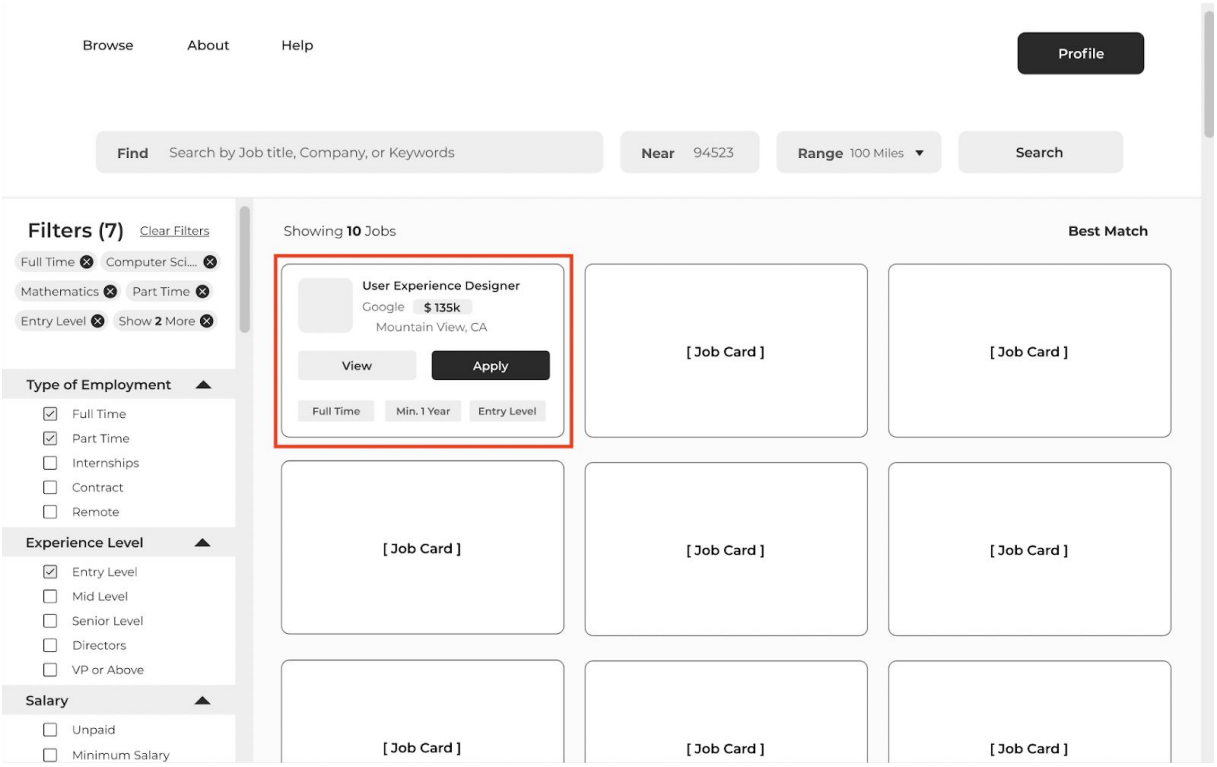
3. Student Profile (redirect)



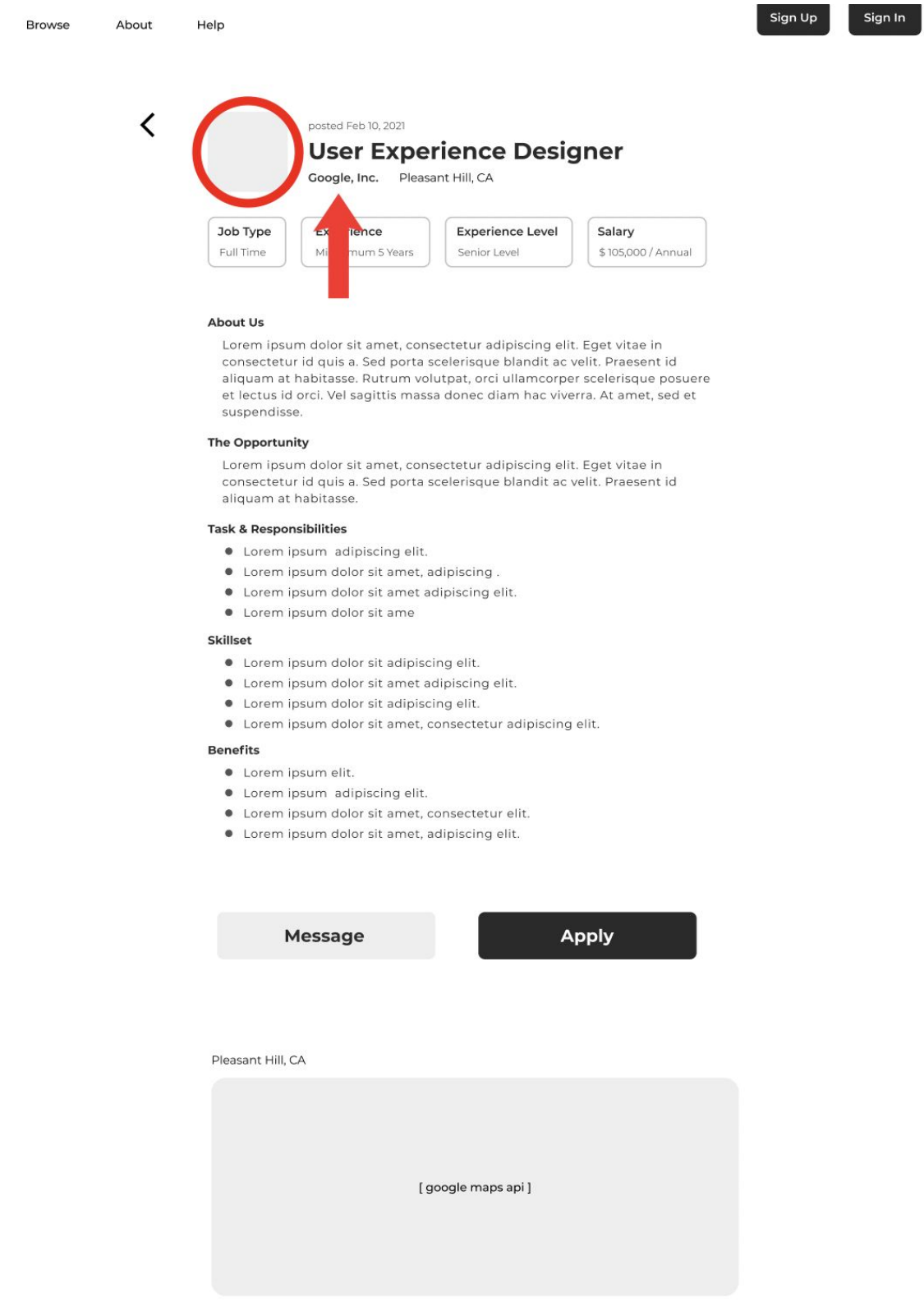
4. Job Search



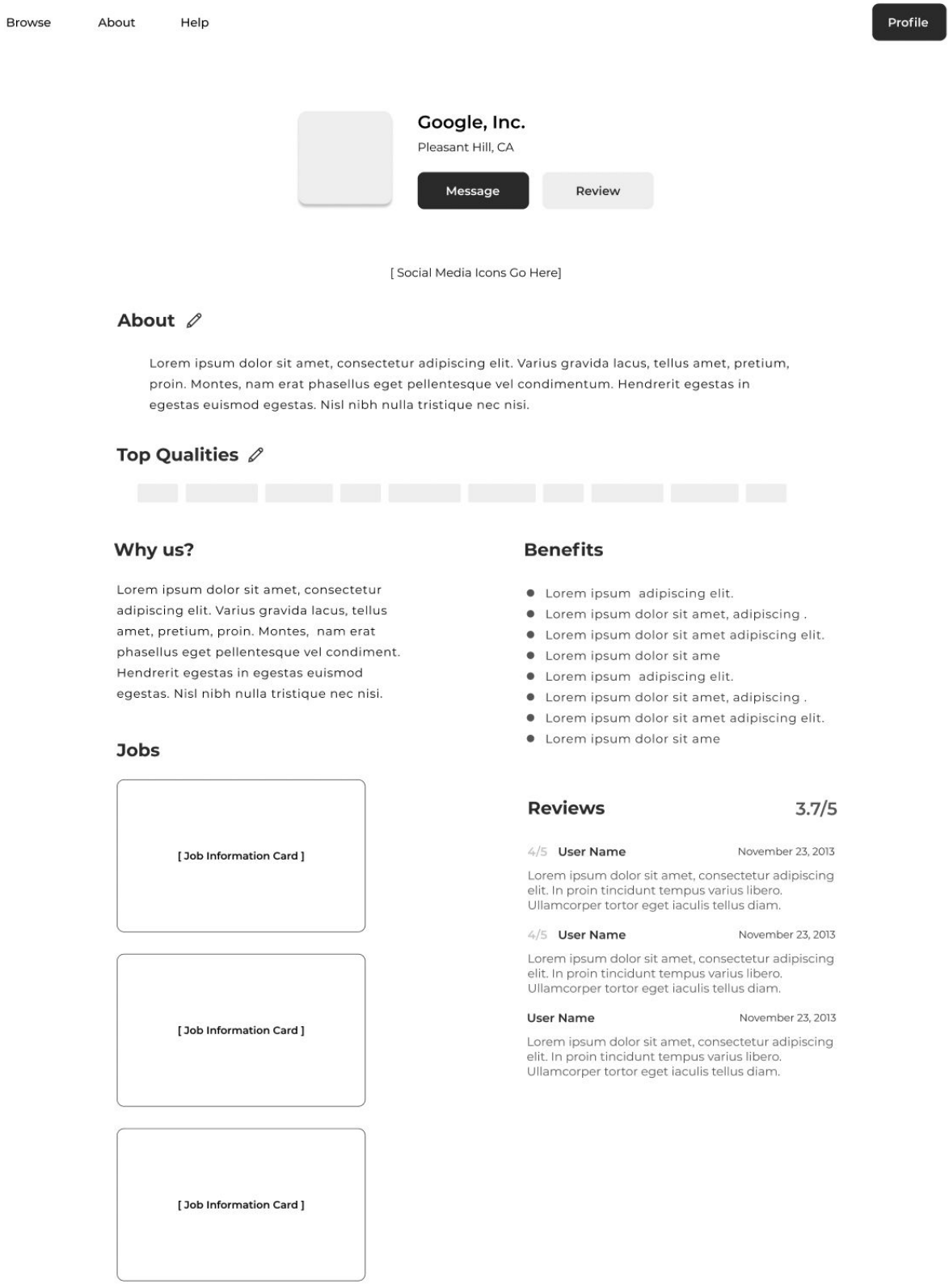
5. Add Job Filters



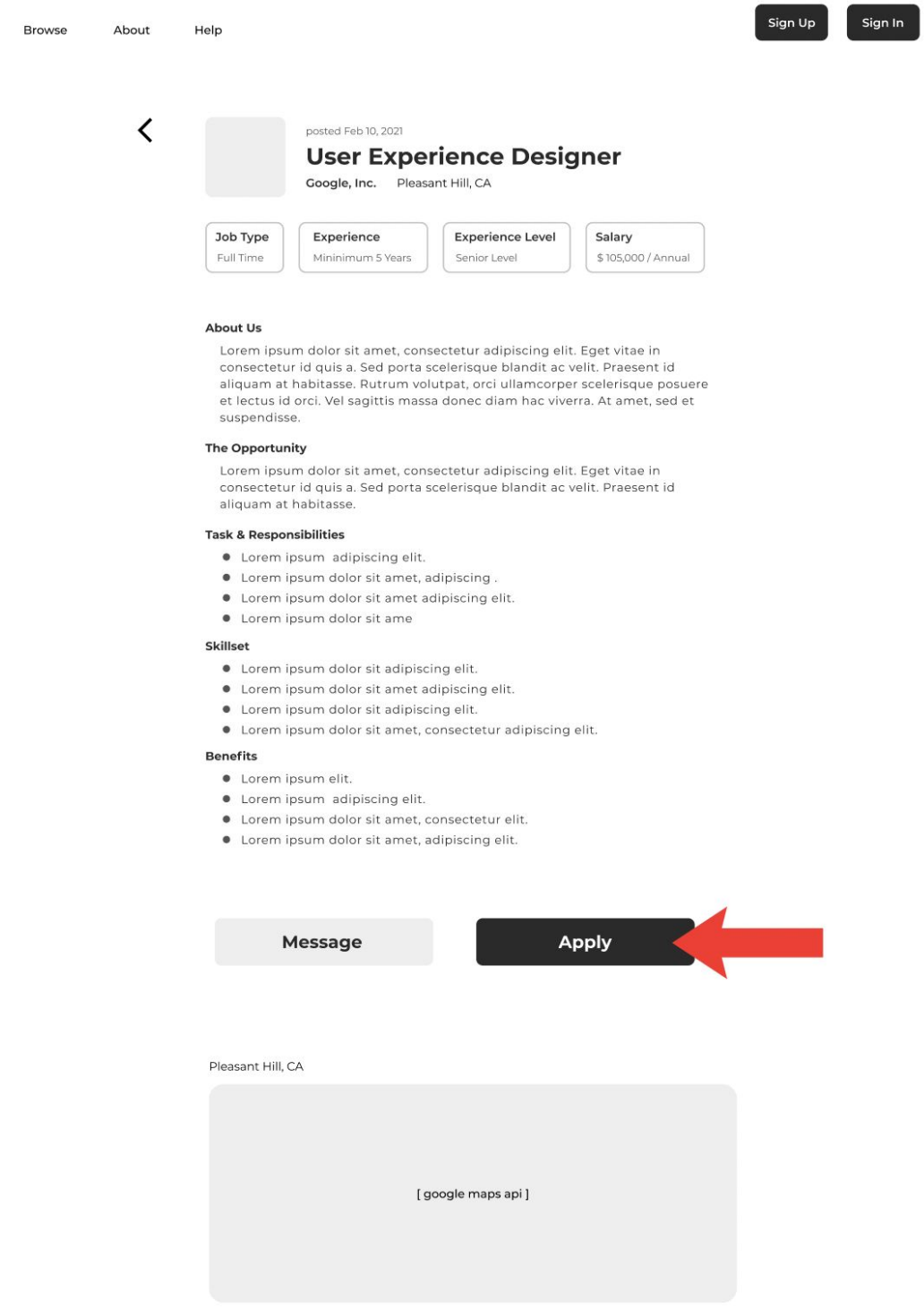
6. Job Details



7. Company Page



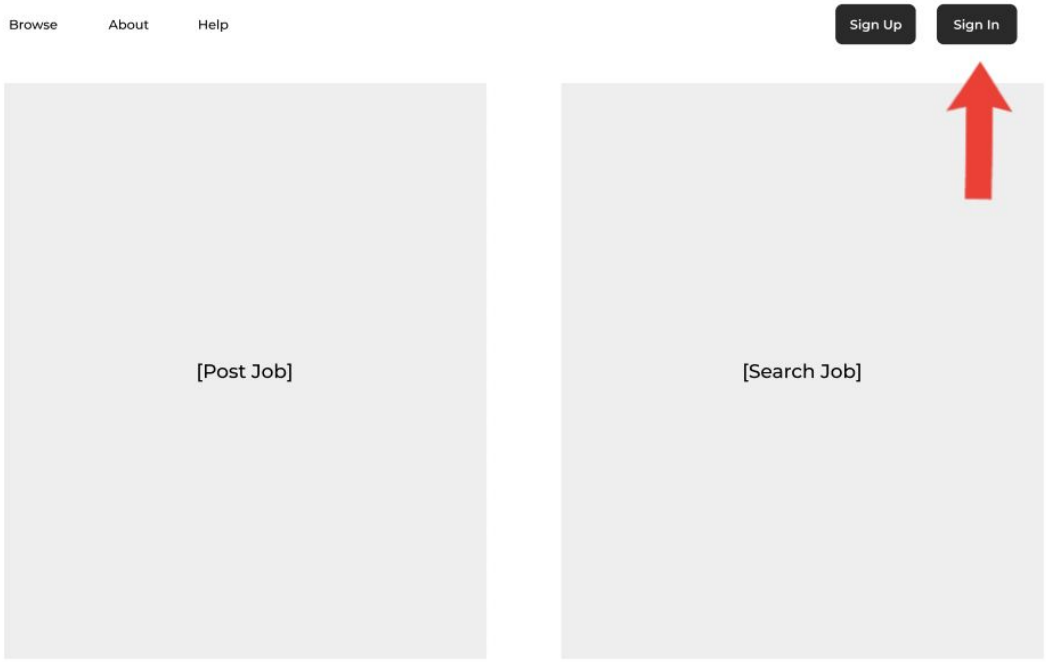
8. Applies to Job



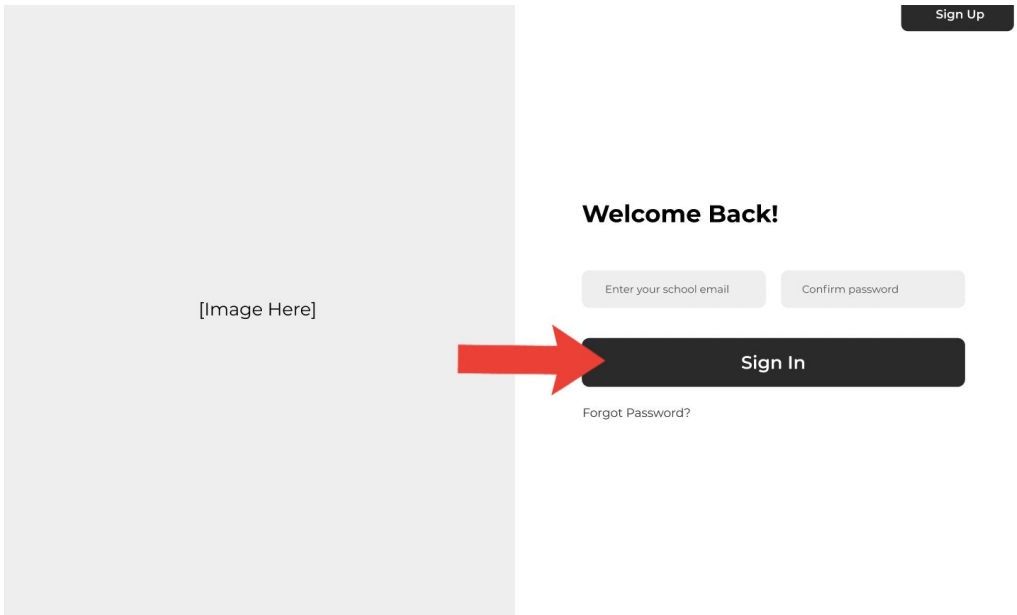
Use Case #5 | Student

Student: Showcases project by adding it to their profile page and requesting verification from their professor

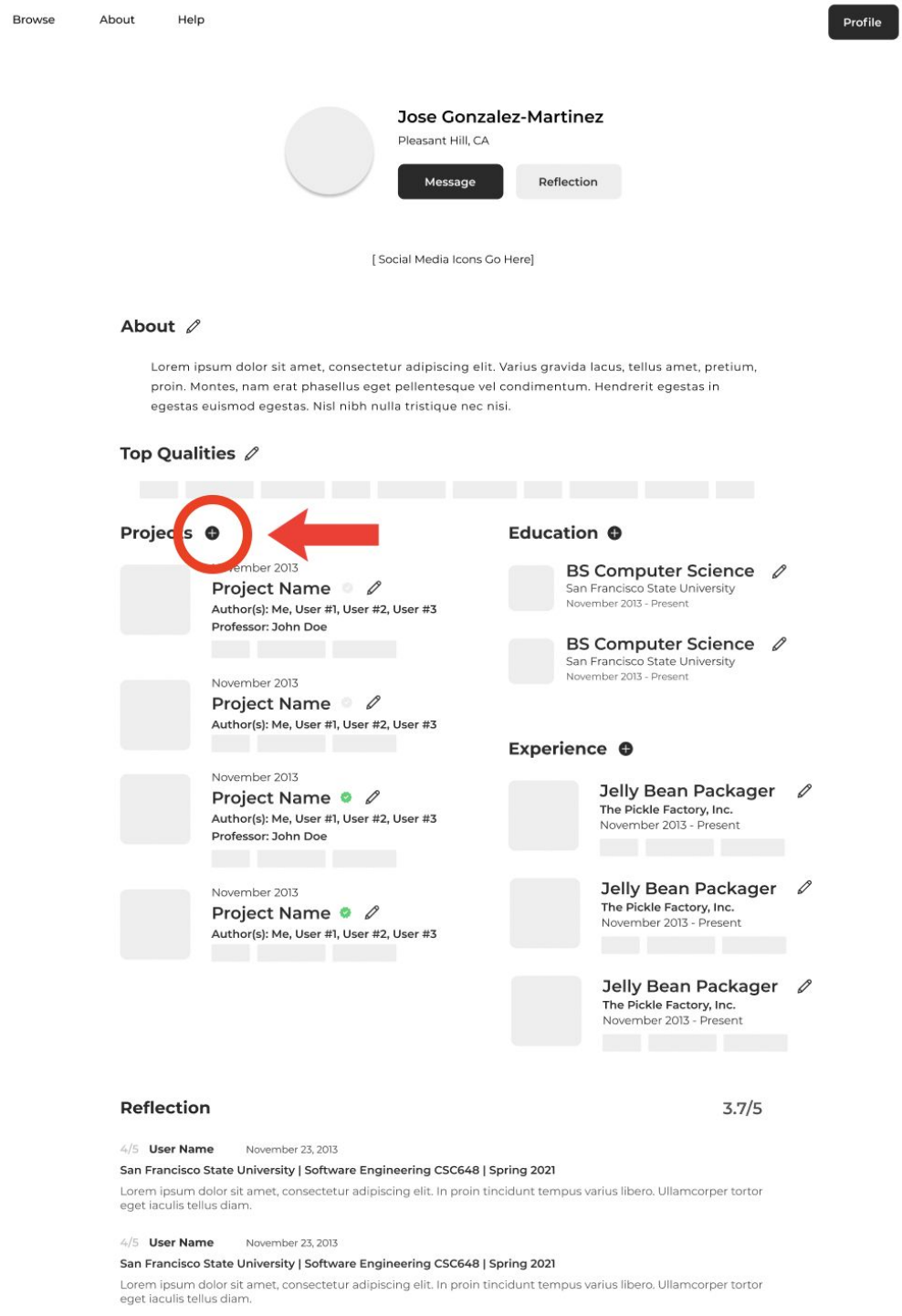
1. Landing Page



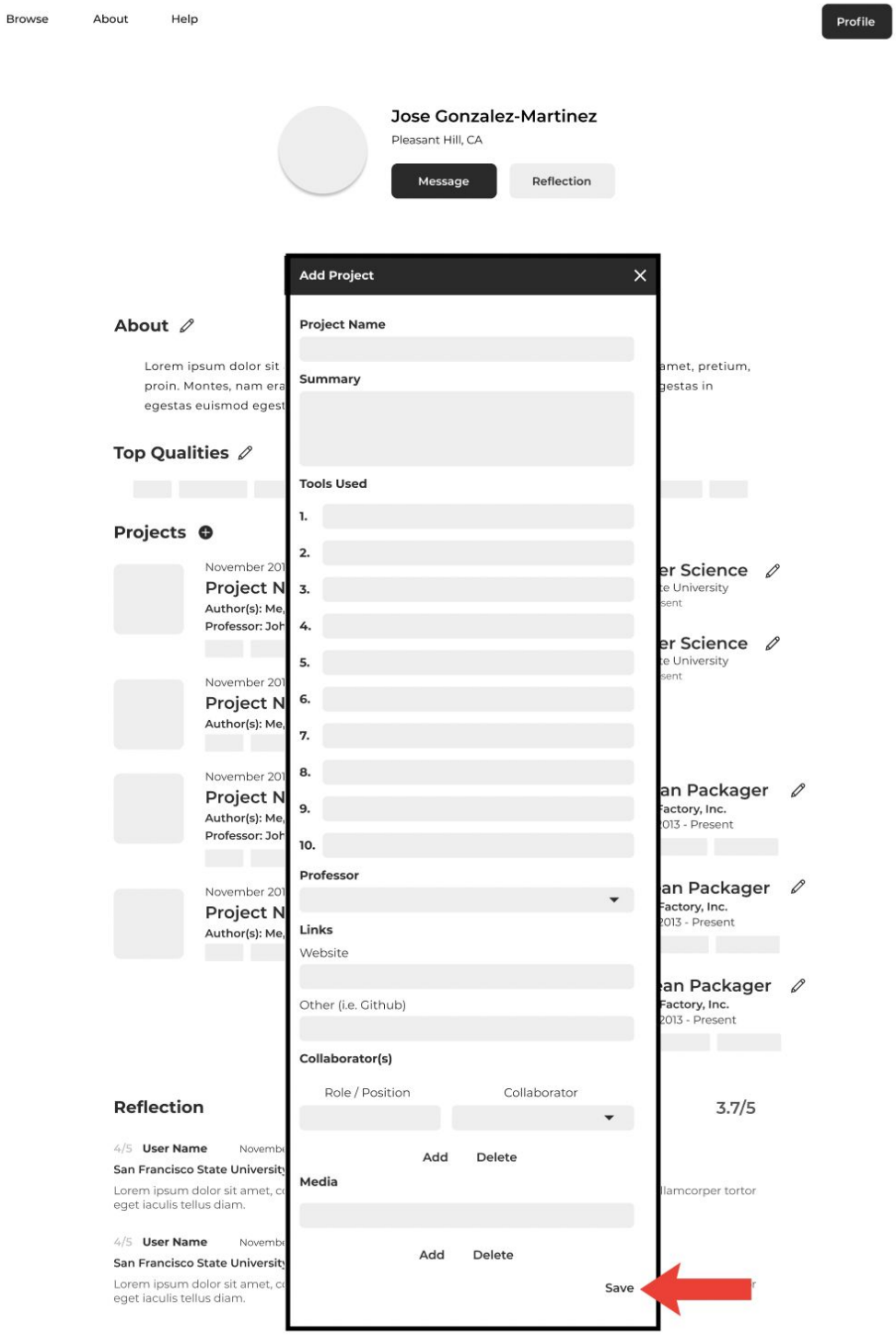
2. Login Page



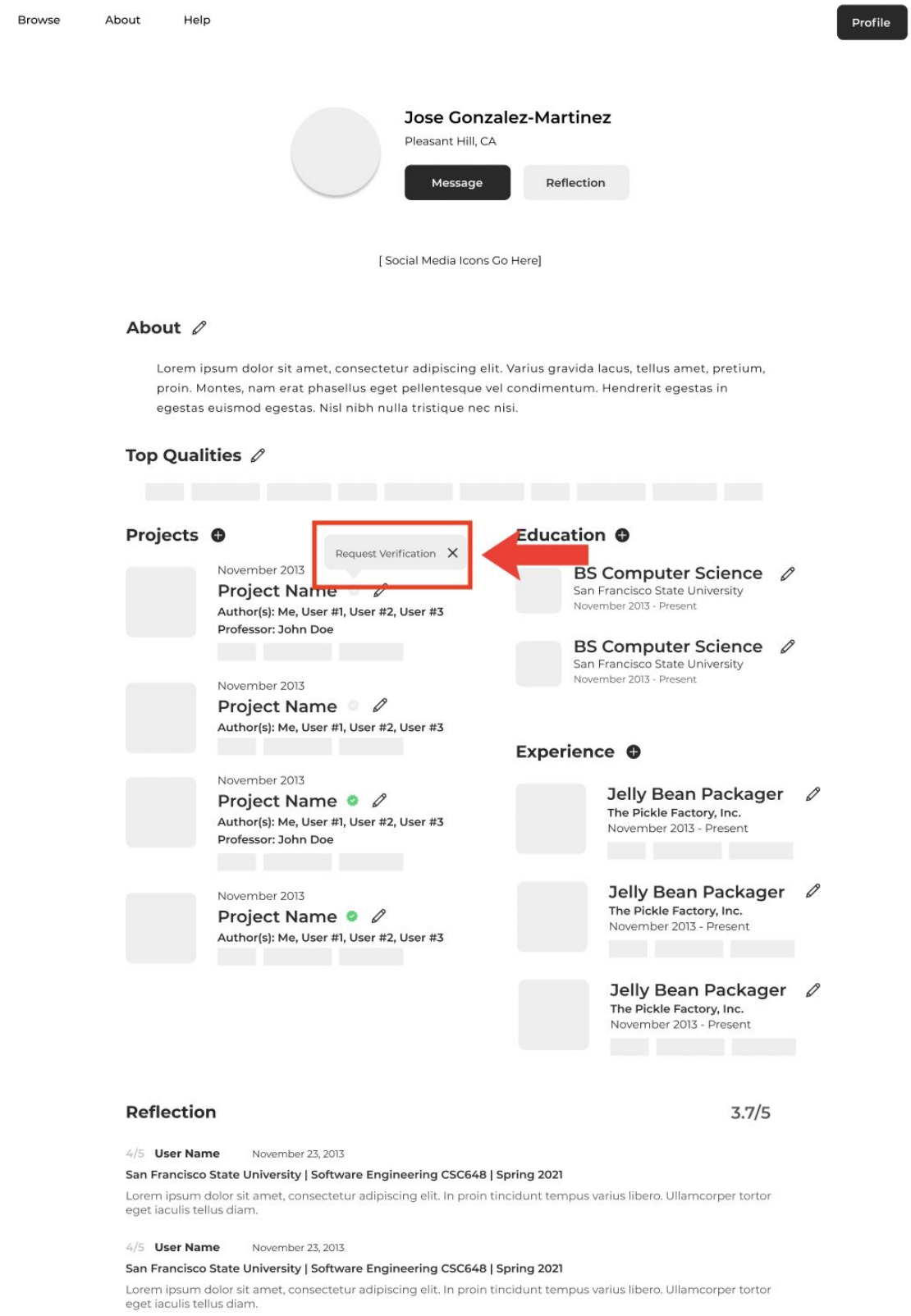
3. Student Profile (redirect)



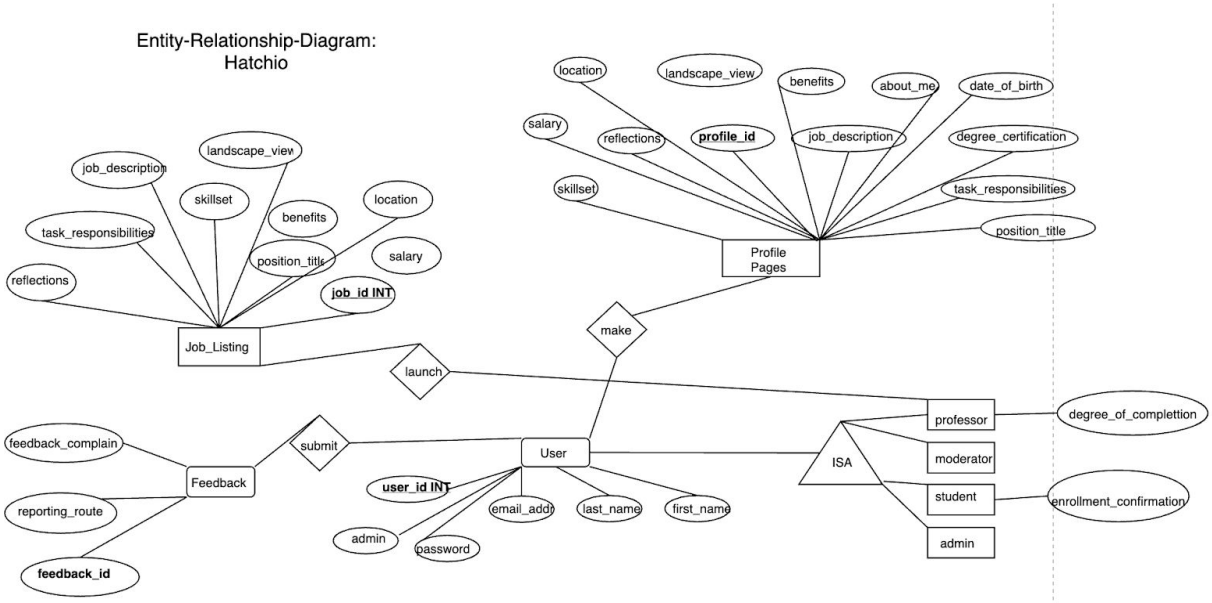
9. Student Profile → Add Project Popup

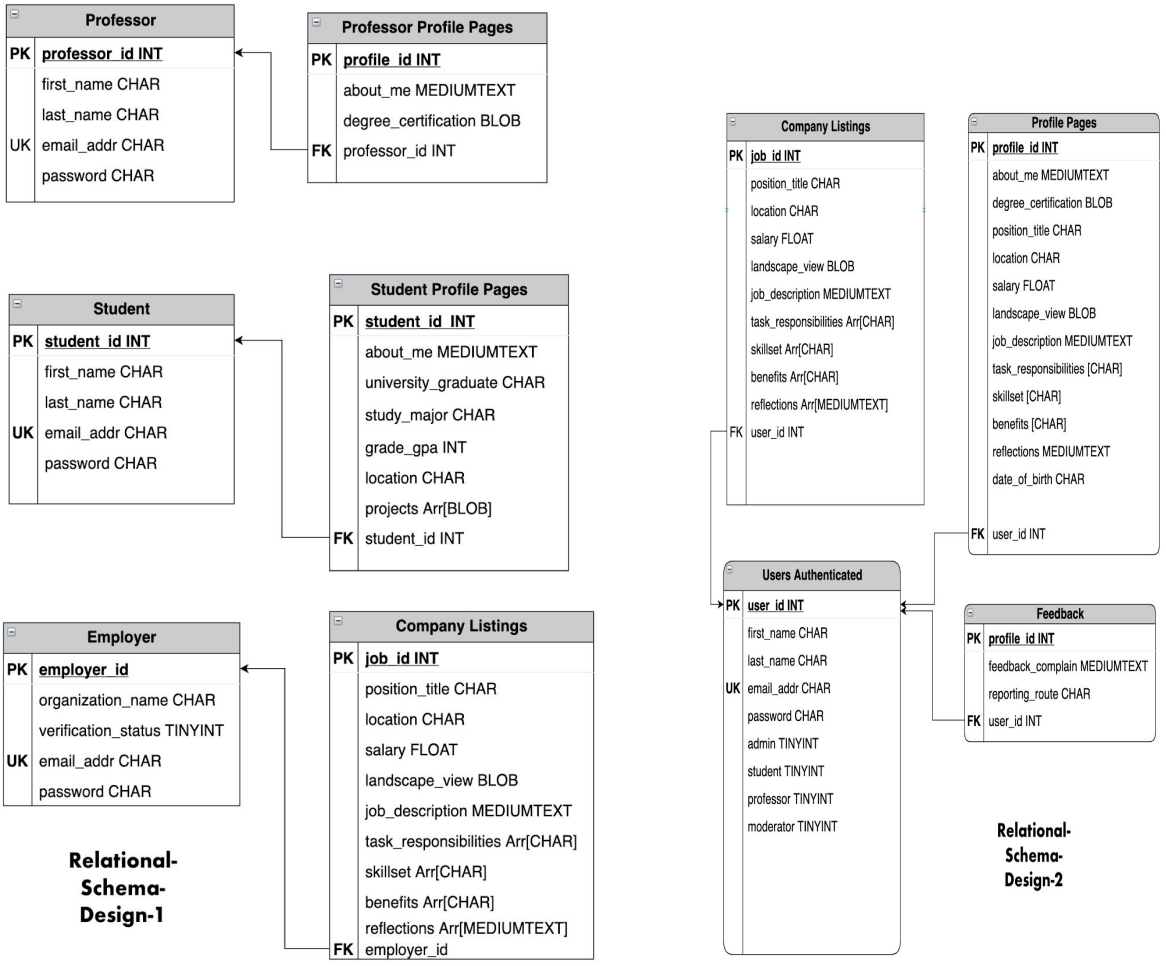


10. Student Profile → Request Verification From Professor



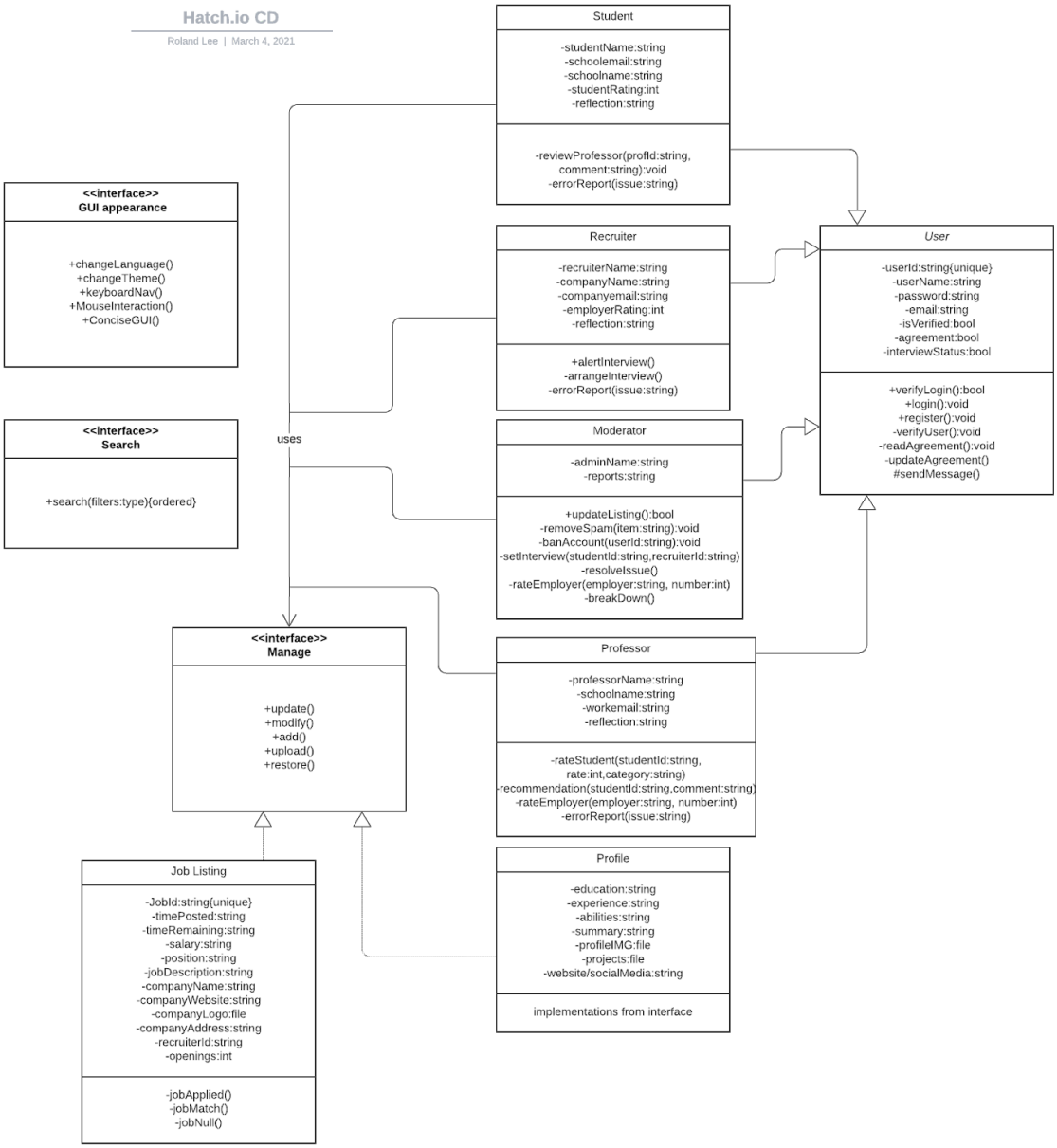
High level Architecture, Database Organization



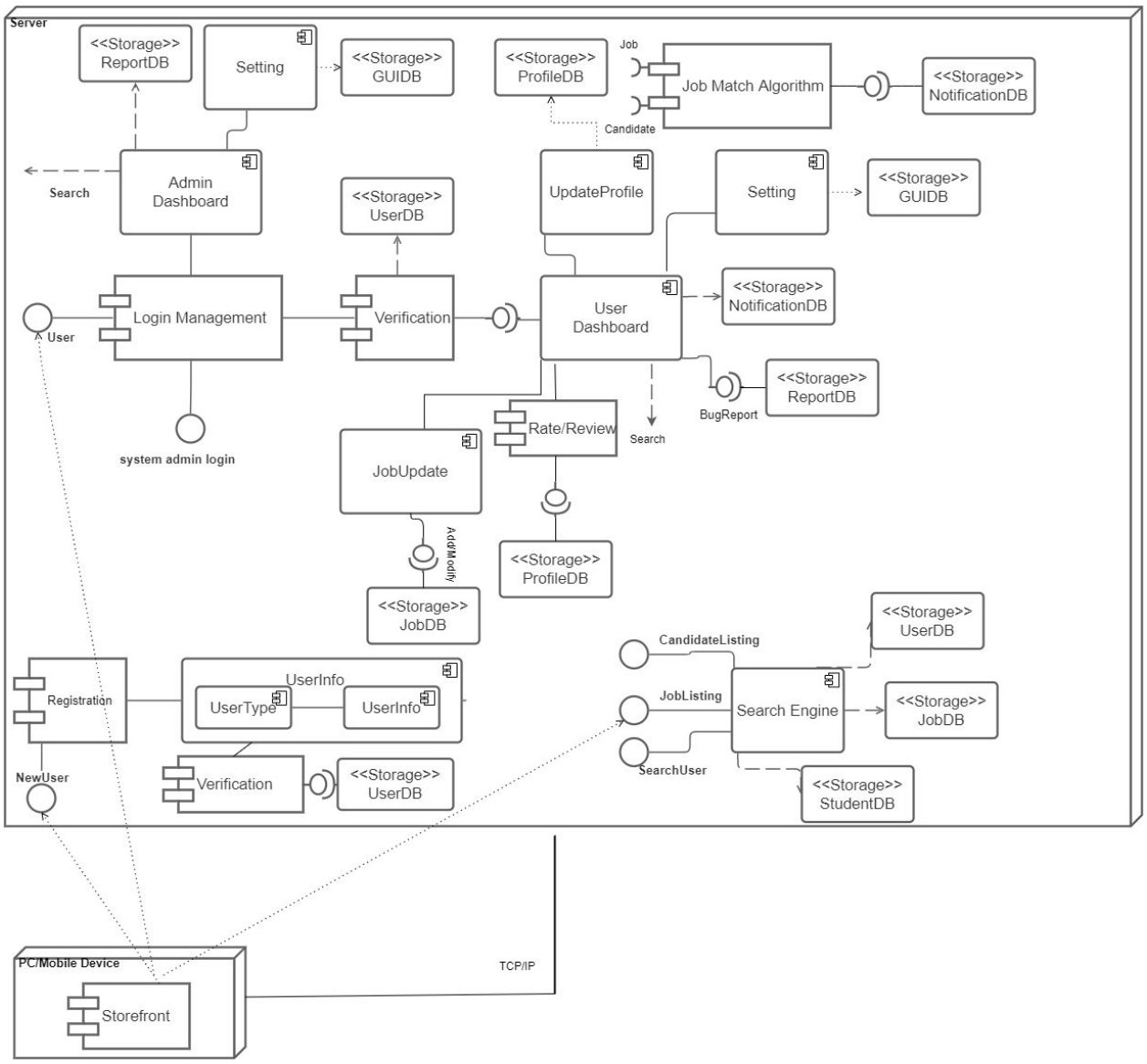


High Level UML Diagrams

UML Class Diagram



UML Component and Deployment Diagrams



Key Risks

Skill risks:

- Correct usage of technologies. People tend to look up tutorials online and use open source code without understanding the libraries/technologies that are used. To resolve this, every member of the team needs to be knowledgeable about the technologies that are listed in the requirements and code correctly and accordingly. If there are any libraries/technologies that need to be included, consult the team lead/CEO.
- Limited resources on selected technologies. Depending on how new/old the technologies are, there might be a limited amount of resources that you can look up. To resolve this, make sure technologies that are used in the project are relevant and prevent using outdated/latest technologies.

Schedule risks:

- Sudden growth in requirements. As a project progresses, issues that are not identified earlier can create a last-minute impediment to meeting deadlines. To resolve this, always think big in the early stage of development and anticipate the worst-case scenario and plan accordingly.
- Gold plating. Gold plating refers to showing off unrelated skills by adding unnecessary features. To resolve this, the team lead needs to ensure no additional functionalities can be added unless all functional requirements have been implemented.
- Productivity issues. Especially for a first-time collaborating team, understanding how well the team can work under pressure and deliver works on time is crucial. To resolve this, set a realistic schedule and communicate with team members. If they need more time on doing research, the team lead should assign tasks accordingly by allowing early project development.

- Estimation of workload and deadline. It is hard to visualize how much work needs to be put in for each assignment. Typical work overload can occur when assignments exceed an individual's ability to handle them. To resolve this, communication is crucial. The team/team member needs to consult the team lead immediately if they fail to deliver products on time. To compensate for it, the team lead also needs to set deadlines accordingly.

Technical risks:

- Breakdown of specification. As the project develops, it's common to have two or more requirements conflicting with each other. Moreover, the team might find some requirements are unclear or incomplete. To resolve this, prioritize functional requirements and add any requirements while sticking to the schedule.
- Resources and performance management. Poor data management and lazy coding will result in deteriorating application performance. To resolve this, make sure implementations are well-established and revised by the tech lead. If necessary, consult with the CTO for improvements on the program's performance.
- Breakdown of server/cloud service. Due to poor network quality, DDoS attacks on the company's server, and other unknown technical reasons, any server or cloud platform can go offline temporarily. To protect all necessary data, make sure to back up necessary files on one's local device and update them periodically.
- Unavoidable events. This includes changes in government policy, the obsolescence of software, natural disasters, or other risks that cannot be controlled or estimated. Since there's no possible way of preventing such events to occur, try to minimize the effect on team performance as much as possible. Or even foresee such events might happen someday and plan accordingly (consult team lead and CEO for immediate response).

Teamwork risks:

- Employee turnover. Every project has a number of developers working on it. When a developer leaves, he or she may take critical information with him/her, which can delay, and sometimes derail the entire project. To resolve this, make sure project history is enabled for new employees to pick up what is left. In addition, make sure resources, research, and knowledge that are required for the project are well placed and organized. Consult the CEO if there's any change in plan that is caused by such an event.
- Team conflicts. Often teams can have disagreements over their goals, methods, or needs of the team. To resolve it, meditation is crucial. The team lead must understand each side of the story and compromise to meet each side's standard. If opinions are divided, the team lead must come up with a decision while respecting other people's perspectives.
- Tasks management. Poor task management will result in typical work overload when assignments exceed an individual's ability to handle them. To resolve this, the team lead needs to make sure the team's workload is divided fairly. Additional tasks can be assigned to one who completed his/her tasks.

Legal/Content risks:

- All features of the website must comply with labor and civil rights laws. Per *Fair Housing Council… v. Roommates.com,* any demographic elements that can be used against minorities and ethnic groups must be prohibited based on legal and moral reasons.
- The website may only use images we are legally allowed to use. This includes default images generated by server configuration. Often there is a possibility that copyrighted material can be placed on open source websites. To resolve it, make sure all assets that are required in the project are open to use, and avoid at all costs using copyrighted materials without the author's consent.

- One risk that can be posed for commercial success is how the content is advertised over other sites. Overly intrusive advertisements may drive visitors away. In order to prevent us from overly branding, we can focus on limited numbers of services to reach a global industry as opposed to targeting multiple organizations.
- One risk that can cause a legal concern is in how to handle payments as the software changes it's milestones over time. It's common for software applications to undergo subscription model changes in order to use their service. A solution to this would be to have a constant income stream. For example, having a free tier with unlimited advertisements from third parties while letting the other tiers can undergo updates.
- Another legal concern might have to deal with is Intellectual Property Ownership. Our software application primarily uses javascript libraries in building the backend and frontend. This includes several open source libraries and therefore require us to understand the full scope of what the license on such libraries like React and Express allow business to use.

Security risks:

- User-generated content must be prohibited from executing javascript on other users' devices, and must be prohibited from using custom styling as that may break accessibility requirements, site layout, and general usability. A simple workaround is to escape the characters '<' and '>', and use Markdown rather than HTML for formatting.
- No user input may cause SQL injection. A possible solution to this is prepared statements.
- User-generated content may not be sufficiently large to cause latency problems for other users. Solution to this is adding input text/file constraints.
- Any sort of security checks for the project must be done server-side, as any client-side restrictions can be trivially bypassed. This doesn't mean that client-side checks cannot

be used, in fact they are helpful to provide better feedback that the user's input is invalid for non-malicious reasons, but this must be duplicated server-side.

Project Management

For each milestone, our team first starts out reading the requirements that are essential to the grading. During class breakout room sessions, our team will have the opportunity to exchange ideas on how we should assign tasks and when the first draft deadline will be. For every milestone, tasks are primarily divided based on the person's team role. If there are any non-specific tasks, they will be distributed across the team to ensure everyone has something to work with. Using M2 as an example, UI mockups and storyboards are frontend tasks which are designated to the frontend team. High level Architecture and database organization are assigned to the database manager. Although Vertical SW prototype isn't part of the M2 submission, the backend team is assigned to the task for early development (since no one in our team is familiar with VP, it's best to have someone look at it in advance). For communication tools, our team has been using google docs for note taking and posting screenshots, Trello for visual task assignment, and discord to achieve team collaboration and services as the primary tool for communication.