[February 23, 2021]

SW Engineering CSC648/848 Spring 2021

Talent Pipeline for SFSU, Hatchio

Section 02 | Team 03

Milestone 01

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	

Table of Contents

Content & Structure for Milestone 01

Executive Summary

Implementing a web-based application focused on creating a seamless pipeline for graduate students will be our team's primary focus this semester. Our motivation stems from surveys highlighting obstacles students and businesses continue to face during the recruitment process. A study by Workforce solutions found that more than 60% of employers said applicants lack "communication and interpersonal skills." The NACE also surveyed more than 200 employers about their top 10 priorities in new hires. Overwhelmingly, they wanted candidates who are *team players*, *problem solvers,* and who can *plan, organize and prioritize their work*. Technical and computer-related know-how placed much further down the list. In addition, 40 percent of college graduates take a job out of school that didn't require a degree. The Federal Reserve Bank of New York also conducted extensive research and found that recent grads tend to take some time after they graduate to find jobs that fit their education. But the fact that the underemployment rate in recent years has been higher than in the early 2000s does suggest that it has become more complex over the past decade for recent college graduates to find jobs that utilize their degrees. Overcoming the student and business concerns noted previously by creating a seamless pipeline for workforce acquisition is highly important in transitioning new grads into the workforce.

Through Hatchio, students will focus on marketing themselves to potential companies by integrating their resume, personal characteristics, education, and most importantly, projects. Through the ease of our app, SFSU students will have the ability to integrate their individual and group projects for recruiters to see. This will allow recruiters to see collaborative projects showcasing soft skills, communication, teamwork, and organization. Not only will this produce a more well-rounded candidate, but it will also allow recruiters to understand more about the student. Students will have the ability to add school team projects, where team members will be allowed to include their role played within the project and their key tasks, and strengths brought to the project. Teachers will also have the ability to validate the project in a way that produces assurance to the recruiters that the project does indeed meet industry standards and that the information is accurate (i.e., grade, student's self inputted strengths, and the completeness of the project). With this newly available data, recruiters will have a better filtration system to fit their company roles. Filtering through projects produced with specific tools, team projects (showcasing teamwork), or even students with a certain amount of teacher validation (overall student score) will provide insight for recruiters focused on soft skills and/or technical skills. Producing a hub for students to showcase their projects, resumes, and more will build a more competitive application and allow businesses to know more about their candidates.

Our team comprises four computer science students, each bringing unique strengths and perspectives to our project. Roland, the team lead and Github master, brings some experience with creating a responsive website using Javascript. Roland states, "This application is more like a review and enhances what I have learned in the past. I'm looking forward to accomplishing tasks while collaborating with my teammates." Aaron, the database manager, has past MySql experience. Aaron is ready to tackle this new challenge and strengthen his skills to the next level! Jose, the frontend lead, is excited to learn the practical methods and tools needed for an entire lifecycle of SW Development. Jose has experience developing a fully responsive website using JS, React, and Node.js. Lyra, the backend lead, brings to the table experience with long-term teamwork on large projects and knowledge of security and accessibility standards. She is interested in branching into web development and proving herself in a formal setting.

Personas & Main Use-Cases

Persona: (Professor) Henry
Use Case #1: Henry Is a professor at SFSU. Interested in ensuring his students succeed in the workplace. Henry logs into Hatchio and finds that several of his students are users of our app and have requested that he evaluate their performance. He proceeds by writing a tailored review for each student on their profile page. In addition, he includes [tags] indicating the strengths of the student.

Use Case #2: As he finalizes his tailored reviews, Henry checks his notifications and sees that one of his students is requesting a project completed for his class to be verified. Henry clicks on the notification, prompting him to the project the student included in his profile page. Henry verifies the student was indeed one of his students, verifies that the project information is accurate, and shares its score.

Persona: (Business & Recruitment) Jazmin
Use Case: Jazmin is a manager at a little-known company. Not many job-searchers are aware of this company, but she believes more people would be interested in working at her company if they were. She logs into Hatchio, enters the relative importance of [TRAITS] to the position, receives a list of students sorted by the weighted average of their scores, and after reviewing their profiles, messages several of them to offer a job.

Persona: (Student) Phillip
Use Case #1: Phillip is looking for a job. He logs into Hatchio, decides what features of a job would be most fulfilling to him, and sends an application to his top choices. A few days later, he gets a message from a recruiter, and in chatting with her, learns that her business has several other advantages and applies there as well. After accepting a position, he is allowed to leave his feedback on the company where he works.

Use Case #2: Phillip just finalized his semester and is really proud of the projects he worked on during the semester. He Logins into Hatchio, navigates to his profile page and clicks on to create a new project. He begins inputting his project information as well as his teammates (who have an account), and the professor. Since Philip added collaborators, he must wait for the entire project information to be added. Once everyone has completed, Philip requests the professor to validate the information.

Subcategories of students:

Persona: Money-Hungry
Overview: Wants to make as much money as possible and doesn't care who gets hurt.
Requirements: Ability to sort by salary

Persona: Jane Student
Overview: Wants to be able to work without being harassed
Requirements: Ability to sort by diversity

Persona: Hype-Train
Overview: Wants to work at a big-name company, something that's currently hot.
Requirements: Unlikely to use job-hunting sites. Ignore.

Persona: Toph
Overview: Needs a screen reader to browse the Internet
Requirements: WCAG compliance

Persona: Paladin
Overview: Wants to make the world a better place. Sees their career as another facet of their activism.
Requirements: Ability to filter by things like democracy, sustainability, etc.

List of Main Data Items & Entities

In this section, the *X refers to a reference into the table X
STUDENT_ABILITIES refers to the set of Initiative, Technical Ability, Communication
BUSINESS_ABILITIES refers to the set of Sustainability, Democracy, Diversity
ACCOUNT_INFO refers to the set of Name, Username, Password Hash, Salt, Email, Is_Verified

Student_Entry: *School, *Student (Not a direct mapping, as students may go to multiple schools)
Student: Major+Minor, STUDENT_ABILITIES (This information is duplicated to avoid race conditions), Resume, Additional Files, *Business (can be null), ACCOUNT_INFO
Student_Review: STUDENT_ABILITIES (can be either all or none as null), *Student, *Teacher
Teacher_Entry: *School, *Teacher (Not a direct mapping, as instructors may teach at multiple schools
Business_Review: BUSINESS_ABILITIES, *Student, *Moderator (exactly one of those is null), *Business
Teacher: ACCOUNT_INFO, Summary, Image
School: ACCOUNT_INFO, Summary, Website
Business: ACCOUNT_INFO, Summary, Website, Logo
Moderator: ACCOUNT_INFO
Position: *Business, Salary, Time Remaining, Openings, Location, Field, Summary
Application: *Student, *Position, Apply_time, Resume (if tailored)
Message: *Business, *Teacher (one of the two is null), *Student, *Moderator (if this is set, only one other is non-null)

Initial List of Functional Requirements (What the system does)

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 registration form for professors shall require a submission of documentation of their degree or experience.
4. The registration form for students shall require entering their name, email, optional address, phone, etc.
5. The system will send out verification emails of approval to any user once their requirements have been verified.
6. The system shall send verification emails when registering for an (any) account.
7. Students shall be able to upload resumes and videos, demographic information, experience, and education information for their profile.
8. The system shall allow students to connect to employers with ease of access by an intuitive interface.
9. Professors shall be able to rate students on a scale from 1-5 on multiple categories and enter recommendations.
10. Moderators and recent hires shall be able to rate employers on a scale of 1-5 on multiple categories as well.
11. Administrators shall be capable of triggering the matching alerts to the companies and alerting students to get ready for interviews.
12. 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.
13. Student users shall be able to search for professors for a review.
14. Student users shall be able to find prospective employers from the system providing an interface to query and filter profiles and positions.
15. The system shall provide a web-based messaging for users to communicate with each other and with moderators.
16. The developers shall take the system developers shall take the system offline under a cyber threat attack.
17. Moderators shall have the functionality to aggressively remove all scams.
18. The system shall deliver feedback to the admin help desk when error reports are created by visitors.
19. The system shall provide multiple viewports for users to select from in order to meet everyone’s accessibility preferences.
20. The system shall periodically update back up database in order to restore information in the event of fatal crash.
21. The system shall provide a dashboard to all user types with customizable layout to personalize their page.
22. The developers shall make sure any bugs reported will be addressed within twenty-four hours.

List of Non-Functional Requirements (How system performs)

1. The application shall be developed, tested, and deployed using tools and servers approved by Class CTO and as agreed in M0 (some may be provided in the class, some may be chosen by the student team, but all tools and servers have to be approved by class CTO).
2. The application shall be optimized for standard desktop/laptop browsers, e.g., it must render correctly on the two latest versions of two major browsers.
3. Selected application functions must render well on mobile devices.
4. Data shall be stored in the team's chosen database technology on the team's deployment server.
5. No more than 100 concurrent users shall be accessing the application at any time
6. Privacy of users shall be protected, and all privacy policies will be appropriately communicated to the users.
7. The language used shall be English.
8. The application shall be very easy to use and intuitive.
9. Google maps and analytics shall be added.
10. No email clients shall be allowed. You shall use webmail.
11. Pay functionality, if any (e.g., paying for goods and services) shall not be implemented nor simulated in UI.
12. Site security: basic best practices shall be applied (as covered in the class)
13. Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development.
14. The website shall prominently display the following exact text on all pages "SFSU Software Engineering Project CSC 648-848, Spring 2021. For Demonstration Only" at the top of the WWW page. (Important so as not to confuse this with a real application).
15. The project shall comply with WCAG 2.1
16. The website shall respond quickly to user input and shall not cause lag in any portion of the user's system. If the client must wait for the server's response, this shall be communicated to the user so they do not suspect a bug.

Competitive Analysis

Feature Dominance Score

Not Available	--
Basic / Minimal	+
Average	++
Leader	+++

Competitors

1. Linkedin
2. Handshake
3. College Central Network (CCN)

Key Features Table

FEATURES	CCN	Handshake	LinkedIn	Team 03
Network	+	+++	++	+
Recommendation	--	+	++	+++
Search	--	+	+++	+
Showcase Project(s)	+	--	+	+++
User Profile	+	++	+++	++

Summary

After doing extensive research, I narrowed our competitors to CCN, Handshake, & LinkedIn. CCN is very dominant within community colleges and offers exclusive job postings, and also uses indeed’s open-source API to showcase other job postings nearby. Handshake is unique because every job posted is specifically for students, and employers are actively recruiting. It’s the only place that connects the student, school, and employers. Handshake has now become the #1 place for students to find jobs. Lastly, LinkedIn is the world's largest professional network on the internet, and 87% of recruiters find LinkedIn to be the most effective in finding vetted candidates.

All three companies are great, but they are missing a key factor, project showcasing. Integrating completed projects where multiple students can be included (indicating what they did within the project) will allow recruiters to see student’s soft and technical skills all at once. Also, professors will have the option to write recommendations on students who have taken their class or write a favorable review on behalf of the project. LinkedIn allows recommendations from anyone but doesn’t have the capability of connecting them to a given project. Linkedin allows project showcasing but does not intuitively enable users to indicate roles, specs, purpose and cannot add images or readme files. CCN will allow students to create their portfolio but lacks UI/UX design, Intuitiveness, and the resources to facilitate collaboration and input. We, on the other hand, will make project showcasing and the professor input our key factors.

Competitor #1 ⇒ LinkedIn (est. ‘02)

Features

- **Network:** Connects & Strengthens professional relationships; finds connection suggestions based on your current connections, location, and industry that will help you to expand your network. (760 million users / 23,000 schools)
- **Recommendation:** Allows anyone to write a recommendation on a user’s profile page
- **Search:** Find the right job or internship, find people (LinkedIn’s Searching)
- **Showcase Projects:** projects can be inputted with collaborators but do not intuitively allow users to indicate roles, specs, purpose and cannot add images or readme files.
- **User Profile:** summary, opening statement or objective, work experience, education, and professional and personal achievements

Research

- 87% of recruiters find LinkedIn to be the most effective in finding vetted candidates—and that number jumps up to 90% for recruiters under 45. (Jobvite Recruiter Nation 2016)
- LinkedIn remains popular with college students (Pew Research Center 2018)
  - 50% of college graduates in the US
  - 9% of people whose education doesn’t surpass high school.
- LinkedIn users are active on the platforms for job recruitment efforts.
  - 20 million companies listed on the site and 20 million open jobs
  - 87% of recruiters regularly use LinkedIn.
- Twenty-two million people received an interview through LinkedIn, with 35.5 million having been hired by a person they connected with on the site.

Competitor #2 ⇒ Handshake (est. ‘14)

Features

- **Network:** (14M users / 800+ schools / 500k employers including every fortune 500 company)
- **Recommendation:** N/A
- **Search:** Robust Search filters
- **Showcase Projects:** upload resumes, cover letter, transcripts, and more
- **User Profile:** Retrieve user information through school API

Research

- positions are mainly internships and entry-level
- Handshake is unique because all the jobs you see are posted specifically for students, and employers are actively recruiting from YOUR school. It’s the only place that connects you, your school, and your employers together.
- recruiters sent more than 16 million messages, giving students on Handshake a personal connection to the top companies hiring at their school. (joinhandshake.com)
- Scams and phishers plague it. While not a technical issue, this illustrates the need for vital moderation.

Competitor #3 ⇒ College Central Network (est. ‘97)

Features

- **Network:** (3.5M students attending CCN-powered institutions / 46+ CA schools / 2M employers)
- **Recommendation:** N/A
- **Search:** exclusive job listing, uses Indeed open-source API
- **Showcase Projects:** build, update and forward online career portfolio
- **User Profile:** minimal

Research

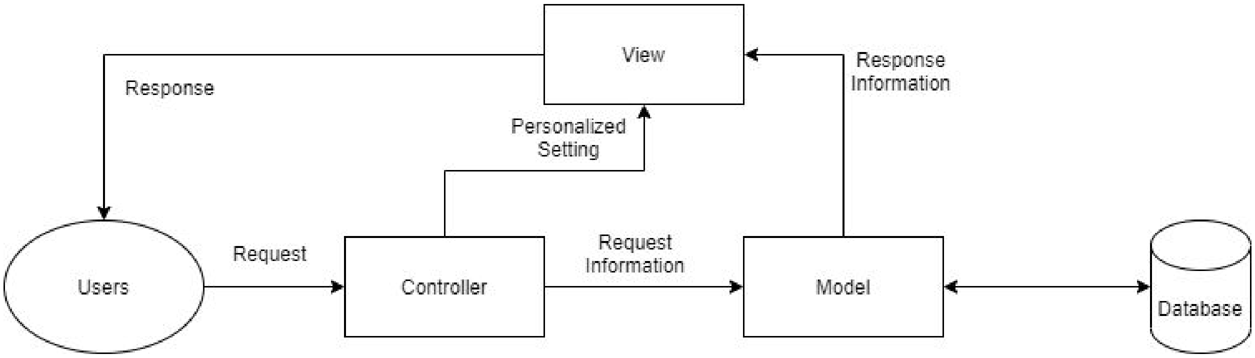
Founded in 1996, (CCN) connects job seekers to entry-level jobs. More than 3.5 million students attending CCN-powered institutions rely on our market-leading technology to interact with their career or employment centers and more than 1 million registered employers. It makes it both FREE and comfortable for all employers to register just once and then post an unlimited number of jobs to California's community college students and alumni.

High-Level System Architecture & Technologies Used

Technology Used:

- Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.
- React is an open-source JavaScript library that is used for building user interfaces specifically for single-page applications.
- Redux maintains the state of an entire application in a single immutable state tree (object). When something changes, a new object is created using actions and reducers.
- Amazon web services is a subsidiary of Amazon providing on-demand cloud computing platforms and APIs to individuals, companies, and governments.
- MySQL is an open-source relational database management system.
- Visual Studio Code is a free source-code editor made by Microsoft for Windows, Linux and macOS.
- Ubuntu is a Linux distribution based on Debian and composed mostly of free and open-source software
- Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the Chrome V8 engine and executes JavaScript code outside a web browser.
- SASS is a preprocessor scripting language that is interpreted or compiled into Cascading Style Sheets.
- Caddy is an open-source web server written in Go.
- Google Analytics is a free web analytics service that allows developers to analyze in-depth detail about the visitors on your website.
- Lets Encrypt is a non-profit certificate authority run by Internet Security Research Group that provides X.509 certificates for Transport Layer Security encryption
- Supported Browsers are: Safari, Chrome, Blackberry, Baidu, Microsoft Edge, Firefox, Opera, QQ, UC, and Internet Explorer 9, 10, and 11 require polyfills.
- Indeed Open-Source API: fetching nearby job positions

Application Architecture



MVC Pattern Overview

Model-View-Controller (MVC) is a software architecture pattern which is most commonly used in web applications. It consists of three components: Model, View, and Controller. In an MVC model, a user sends a request to the controller through a UI. Then the controller passes the information to the model and returns specific data. After that, the data gets transferred to view and presented via the specified view. In addition, data is all stored in the database, which can only be accessed through the model. Brief descriptions on each software components are as follows:

The Model

The Model manages all tasks related to data: validation, registration, modification, session state and control. In addition, the Model is responsible for the business logic of an application. It will encapsulate methods to access data (feedbacks, ratings, uploaded files, etc.) and handle data source structure to ensure data restoration and backup features are feasible.

The View

The View is responsible for graphical user interface management which represents all forms, buttons, graphic elements that are used in the application. Ideally, the View component should never contain business logic that belongs to the Model and controls how the data is displayed and how the user interacts with it. In this application, the View layer is formatted in CSS using Javascript.

The Controller

The Controller manages all possible events that will happen in the application. These events can be triggered by either user interaction with the application or system process. When a user's request requires data access, the controller should be able to interact with the Model and retrieve the corresponding data. If the user's request doesn't require access to the data, it's most likely to be a customization setting of the View which should be forwarded to the View component.

Team & Roles

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

Checklist

1. ✓ The team found a time slot to meet outside of the class Tuesday 10PM
2. ✓ Github master (Roland Lee)
3. ✓ The team decided and agreed together on using the listed SW tools and deployment server.
4. ✓ Team ready and able to use the chosen back and front end frameworks, and those who need to learn are working on learning and practicing
5. ✓ The team lead ensured that all team members read the final M1 and agree/understand it before submission
6. ✓ Github organized as discussed in class (e.g., master branch, development branch, folder for milestone documents etc.)