**CS673 Software Engineering**

**Team 2 - CareerForge**

**Meeting Minutes**

All meeting minutes are kept in this single document. The latest meeting minutes should be at the beginning of the document. For example, meeting 3 minutes is placed before meeting 2 in the document. The team leader should prepare a basic agenda for the meeting and team members should rotate to be the minutes taker. Each group should have at least one meeting per week, and you may have multiple meetings if needed.

**Meeting 3**

**Meeting 2**

**Date and Time:** September 10th

**Place**: Google Meet

**Participants:** Gopi, Pedro, Yongxiang Chen, James, Qi Chen, Stacey

**Minutes taker:** Stacey

**Timekeeper:** All team members

**Purpose:** Discuss next steps

**Agenda:**

* How are we handling document submission through GitHub?
* Finalize deliverables
* Get started on Iteration 1

**Discussions:**

* CareerForge Design Proposal
  + Login/Register Screen
    - Username, Email and Password
      * Do we want to include email field here? TBD
    - Register Button
    - Login Button
  + Once logged in, the left side will have a list of jobs, the middle will be 1 job, and the right field will have field, location, type, and my jobs links
    - Allow change in screen color (dark and light theme)
    - Need GIT request to API to display the apps
    - If user clicks apply, the information gets pushed to the backend with the app ID, description, etc
    - If user doesn’t click on apply button, we will sent data to backend in a different column from applied button
    - What’s the difference in functionality between saving and applying?
      * One will be saved and one will be applied
    - On landing page, how will we determine which job is highlighted?
      * Whichever job is at the top of the list will appear here
    - Will the Rise API be in the backend and then pass that to the frontend?
      * Rise API is free and will provide a list of jobs already built into the return call
  + Purpose of the app is to have a tracking system
    - Will see all jobs you have applied to and/or saved and/or delete
* CareerForge Roadmap
  + Will be splitting the team up between the front and backend
  + First week we need to set up front and back end, database schema
  + Second week - authentication with JWT
    - Frontend team will create job results page, job detail components, and algorithms to filter
    - Backend team to work on authentication and REST endpoints
  + Third week we will implement saving, applying, deleting to database and implement the my jobs dashboard
  + Fourth Week will be meant for testing and deploy database (SQLite or PostGres(RDS from Amazon))
  + Fifth week - polishing, finalize testing, prepare for presentation demo
* Login/Registration Activity Flow
  + Login path that has a forgot password link - Is this going to be too much?
  + User chooses login or register
    - If password is not valid or weak, we need to throw error
    - If valid, we need to hash the password
    - All validation will need to be in backend
      * Actual email, etc
      * JWT token to be used

**Key Decisions:**

* Spec design has been approved

**Action Items:**

* Team to download postman and check out Rise API and the data it returns
* Jimmy to find a platform to create tickets and manage the project (ClickUp?)
  + Need something with agile layout (user story and sprint function)
* Gopi and Pedro will to start creating tasks/tickets
* If anyone has any suggestions on designs, please raise ASAP
* Plan to meet again on Monday at 7pm

**Meeting 1**

**Date and Time:** September 7th 11-1pm EST

**Place**: Google Meet

**Participants:** 5

**Minutes taker:** Stacey Burns

**Timekeeper:** All team members

**Purpose:** Kick off meeting

**Agenda:**

* Introductions
* Discuss tech stack to be used for the project
* Discuss and decide on what project we should do
* Assign roles
* Discuss deliverables needed for Iteration 0
* Review \_SPPP document

**Discussions:**

* What project ideas do people have?
  + One idea is to have a job search web application where you can log in, search for jobs, apply for jobs, add and delete. Users will be able to create their own credentials. It would be simple and would include both front and back end dev work. Pedro has checked out some free APIs we can use here. Pedro would be in charge of the requirements with this one.
  + Another idea we discussed was a study group matcher. It would connect students in the same class based on schedules and study preferences. Cuts down on the friction of forming groups.
  + Another idea was an app to track calendar events. Campus Event Hub- Centralized calendar for university events, with RSVP, reminders, and map integration. Solves the “I didn’t know this was happening” problem.
  + Another idea was MindTrack Software. It would be an app based around emotions, mood tracking, and more.
* What tech stacks are people comfortable using?
  + JS with Typescript, [node.js](http://node.js), and react, java, and python
* Started to work through the \_SPPP document
  + What should our team name be?
    - Decided between a few, but CareerForge won out
  + What roles would people like to take on?
    - It looks like most people are comfortable with the roles they signed up for. We just need a team leader.
  + Added revision history to the document
  + What is the overview of our project?
    - What is the motivation, purpose, and the potential users for this project? What is the basic functionality we want to implement?
    - We discussed how we want the user to be able to search for jobs, create their own accounts, save jobs, and track the status of their applications
  + What are some similar software applications that currently exist and how do they differ?
    - We focused on LinkedIn, Indeed and identified the simplicity of our app with no ads
  + Requirements
    - We discussed whether we want to focus on just employees or on employer as well. If employers are able to use the site, they can post their jobs for the employees to see. However, there is concern about time and complexity of the project. An employer would need their own user group and completely separate interface so that would require more work and time we might not have. Added this as a desirable feature
    - Should we be able to export data?
    - Since employers will not be our sole focus at first, we discussed using stubbed/mock data for job postings initially.
    - What are the person hours per task? We are unsure of the total person hours for each, but we discussed that the more complex tasks would be around 40 person hours. For the first requirement to be completed, we would need the frontend, the backend, API, and DB set up and ready to go.
    - What are some desirable, optional, and non-functional requirements we want to focus on?
    - Security requirements - Level of complexity?
      * MFA, password hashing
  + Management Plan
    - Based on the above, we discussed how our first iteration would focus on user creation/access and the ability to view job listings. Future iterations will focus on search functionality, and job applications.
  + Risk Management
    - We need to complete the Risk Management \_SPPP for Iteration 0.
  + Timeline
    - For the first iteration, we will focus on user account creation and mock data for job postings. We discussed the person hours it would take for this and the tasks needed.
    - The second iteration will focus on MFA and password hashing. We discussed the person hours it would take for this and the tasks needed.
    - The third iteration will focus on search and save functionality. We discussed the person hours it would take for this and the tasks needed.
  + Configuration Management Plan
    - Discussed what kinds of CI/CD pipelines people are used to working with. Jenkins has been used. For purposes of our project, we will be using GitHub Actions.
    - ChatGPT, IntelliJ, and VS Code were discussed
    - Docker - are folks experienced with it? Is it something we will be able to use for this project?
    - What SAST/DAST tools can we use?
  + Code Branching Strategy
    - Looking at the git branching strategies article, we discussed GitFlow and trunk based development. GitFlow seems to make the most sense for us, using a dev for development, QA for testing, and master as the main branch we merge to.
  + QA Plan
    - Discussed automation framework to be used, including metrics - what they mean, coding standards, and code review process.
      * We should use a checklist for reviews
    - Testing will involve automation, manual, and unit testing.
    - Defect management - should we raise a task in JIRA? It was noted that GitHub has it’s own defect management tool. We will use that - GitHub Issues.
  + AI Usage Log
    - Noted that AI that is being used will need to be added as an appendix to our documents
* What are our next steps?
* How often should we be meeting weekly?

**Key Decisions:**

* We decided to go with the job search web application
* We choose to build the back end using java and the frontend using JS with react and typescript
* Our group name/project name would be CareerForge
* The team roles will be as follows:
  + Gopi - Config Leader
  + Pedro - Design and Implementation
  + James - Team Leader
  + Qi Chen - Requirement Leader
  + Stacey - QA Leader
* For CI/CD, we will be using GitHub Actions
* Our commit and branching strategy will follow similarly to GitFlow
* We should aim to meet twice a week

**Action Items:**

* Stacey to write up the meeting minutes formally
* Gopi to create a time management link to schedule standups and meetings for the future
  + Plan is to meet at least twice a week
* Pedro to complete the [Readme.md](http://readme.md) file
* Pedro to set up the GitHub repository
* James to get familiar with GitHub, VS code with copilot
* Stacey to complete the Risk Management\_SPPP for submission
* Stacey to create a JIRA board
* Team to research SAST and DAST