**CS673 Software Engineering**

**Team 2 : A-Team**

**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 5**

**Date and Time:** 6/11/2022 1:30pm-2:30pm

**Place**: Google meeting

**Participants:** Bharat Gogineni ,Ruiqi Chang, Chinmay Bhelke, Benoît Clemenceau, Taina Conde

**Minutes taker:** Chinmay Bhelke

**Timekeeper:** Ruiq Chang

**Purpose:** Iteration 2 submission

**Agenda:**

1. Back end

2. SPPP

3. SDD and STD

**Discussions:**

* Went through what we have now: backend and frontend in github
* Summary what we get in the Iteration
* Assige the doc part to team members.
* Make a plan for the last iteration of this project
  + By Monday - will finish Edit Job , Edit Activity , and Delete Job , Delete Activity
  + Optimistic Features :

CSS stuff (animations , etc.)

Landing Page

Items :

| Chinmay | Bharat | Ruiqi | Ben | Taina |
| --- | --- | --- | --- | --- |
| SPPP Overview | SPPP Risk Assessment | High Level Reqs | SPPP Quality Assurance Plan , nonfunctional Testing | React App Functionality - Cards , etc. |
| Edit Job , Edit Activity |  | SPPP Timeline | STD Doc | Presentation |
| SPPP - Tools |  |  |  |  |
| SDD Software Arch |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Meeting 4**

**Date and Time:** 6/4/2022 1pm-2:30pm

**Place**: Google meeting

**Participants:** Bharat Gogineni ,Ruiqi Chang, Chinmay Bhelke, Benoît Clemenceau, Taina Conde

**Minutes taker:**

**Timekeeper:**

**Purpose:** Iteration 1 feedback and lab3

**Agenda:**

1. Back end

2. Feedback on SPPP

3. Feedback on SDD

4. Feedback on STD

5. Testing

6. Implement Redux

**Discussions:**

Discussed and work on the Database and backend.

Went through te SPPP, SDD, and STD new comments

Discussed the lab3 and testing part of the project: JEST test

Discussed and show the example the Redux which we plan to apply in the project.

***Requests for Backend***

To make requests to our Go microservices - we simply need the user auth token

| **Example of how to get the Auth Token from Frontend :**  **Just call { getAccessTokenSilently } = useAuth0()**  import React from 'react'  *// import JobCard from '../components/Job\_Components/JobCard'*  import JobCardNew from '../components/Job\_Components/JobCardNew'  import { StyledCardsContainer } from '../styles/styles'  import HomePageHeader from '../components/HomePageHeader'  *// import Counter from '../components/counter/Counter'*  import { useAuth0 } from '@auth0/auth0-react'  import { jobData } from '../data/mockdata'  import NoJobs from '../components/Job\_Components/NoJobs'  *// import { Jobs } from '../types/types'*  *// type JobItems = {*  *// [key: string]: Jobs[]*  *// }*  export const Home = () => {  const { user, isAuthenticated, getAccessTokenSilently } = useAuth0()  if (isAuthenticated && user) {  const email = user?.email ?? 'none'  const jobItems = jobData[email as keyof typeof jobData]  const getClaims = async () => {  const claims = await getAccessTokenSilently()  console.log(claims)  }  getClaims()  if (jobItems)  return (  <StyledCardsContainer>  <HomePageHeader />  {*/\* <JobCard /> \*/*}  <br />  <br />  <br />  {jobItems.map((*item*: any) => {  return (  <JobCardNew  companyName={*item*.companyName}  jobTitle={*item*.jobTitle}  status={*item*.status}  />  )  })}  </StyledCardsContainer>  )  else return <NoJobs />  }  return <div>None</div>  }  The console.log() of the token from above component will look like this :  This is the exact token we will pass to the backend in the format of :  **Bearer <tokenvalue>**    Sample Requests from Postman :  ***Create Job :***  **Endpoint:**    **Body :**    **Header :** *(note to include Bearer and then the token value)*    All we need is the   * API endpoint * Authorization Header * Body   in order to get a response from the backend. |
| --- |

**Reference Table for Rest API’s**

| **API** | **Functionality** | **Request Body** | **Endpoint** |
| --- | --- | --- | --- |
| **createJob**  **(*POST*)** | User wants to create a job to track | **{**  **"jobtitle" : "Engineer",**  **"companyname" : "Patagonia",**  **"description" : "Engineer @ Patagonia",**  **"status" : "In Progress"**  **}** | https://x3pmfzyrll.execute-api.us-east-1.amazonaws.com/default/cs673\_job |
| **getJobs**  ***(GET)*** | We will use this API to retrieve the list of jobs for a user | **No Body Required - we will use the token to decode user’s email. Just make a GET request to endpoint** | https://x3pmfzyrll.execute-api.us-east-1.amazonaws.com/default/cs673\_job |
| **createActivity (*POST*)** | A User wants to add an activity for a job they are tracking | **{**  **"category" : "Coding Round",**  **"description" : "1 hour long coding challenge",**  **"start\_date" : "1/1/2022",**  **"end\_date" : "2/2/2022",**  **"status" : "in prog",**  **"job\_id" : "2"**  **}** | https://x3pmfzyrll.execute-api.us-east-1.amazonaws.com/default/cs673\_activity |
| **getActivities**  ***(GET)*** | We will use this API to retrieve the list of activities for jobs | **No Body Required - we will just use Query String Params. Just make a GET request to endpoint and provide the Job ID at the end** | https://x3pmfzyrll.execute-api.us-east-1.amazonaws.com/default/cs673\_activity?job\_id=2 |

| **Create Job :**  > From Frontend , pass token and body to API  **API Endpoint** :  <https://x3pmfzyrll.execute-api.us-east-1.amazonaws.com/default/cs673_job>  Request Body :  {  "useremail" : "1234@1234,cin",  "jobtitle" : "Googling",  "companyname" : "Googol",  "description" : "RajRaj",  "status" : "In Progress"  } |
| --- |

**Features :**

* The Create Job API will handle duplicate requests and return an appropriate 400 response if the user tries to add the same Job (same Job Title and Company Name fields)
* The Create Activities API will handle duplicate requests and return an appropriate 400 response if the user tries to add the same Activity ( same JobId, Category , and Description fields)

**Meeting 3**

**Date and Time:** 5/28/2022 1pm - 3:40pm (2h40min)

**Place**: Zoom meeting and Google meeting

**Participants:** Bharat Gogineni ,Ruiqi Chang, Chinmay Bhelke, Benoît Clemenceau, Taina Conde

**Minutes taker:** Chinmay Bhelke

**Timekeeper:** Taina Conde

**Purpose:** Iteration 1 submission

**Agenda:**

1. Check if anyone had problems running the project

2. Project UI designs

3. Back end set up

4. What feature we will deliver on iteration 1

5. Assign tasks to develop the feature

6. What parts of SDD and STD we are doing for iteration 1

**Discussions:**

* Discuss Agenda
* Troubleshooting SSH Key and connecting to backend issues - decided to have a session at end of meeting
* Talked about Class Diagram and UI Prototyping
* Bharat went over Backend architecture and file structure
  + Migration file for Database
  + Middleware File for Backend and Client
  + Deployment Model
* Taina went over Frontend changes branch
* Discussed first feature to deliver
* Went over user story to move to current Sprint
* Went over tasks for stories to deliver Feature in Iteration 1
* Went over Structure of Requests

Example PUT Request for Adding Job :

| jobTitle: {  type: Sequelize.STRING  },  companyName: {  type: Sequelize.STRING  },  description: {  type: Sequelize.STRING  },  userId: {  type: Sequelize.INTEGER  },  status: {  type: Sequelize.STRING  }  Example request :  POST  body : {  jobTitle : “sde”,  companyName: “amazon”,  description : “software dev engineer level 5” ,  status: “in progress”,  jobId : “12345”  }  GET  /getjobs/:userid  <JobComponent body … / >  Test Data  User Table  EmailID FirstName  [chinmayy@bu.edu](mailto:chinmayy@bu.edu) Chinmay  [abc@bu.edu](mailto:abc@bu.edu) John  Jobs Table  // User 1 signs in , creates 2 jobs - Amazon , Google  Amazon - JobID : 9999  Google - JobID – 8888  // User 1 creates 3 tasks for Amazon , 3 tasks for Google  Amazon - 3 tasks  Task 1 - ActivityID : 111 , JobID : 9999  Task 2 - ActivityID : 122 , JobID : 9999  Task 3 - ActivityID : 133 , JobID : 9999  Google- 3 tasks  Task 1 - ActivityID : 222 , JobID : 8888  Task 2 - ActivityID : 211 , JobID : 8888  Task 3 - ActivityID : 233 , JobID : 8888  // User 2 signs in , creates 1 job - Google  Google - JobID - 7777  // User 2 creates 4 tasks for Google  Task 1 - ActivityID : 800 , JobID : 7777  Task 2 - ActivityID : 801 , JobID : 7777  Task 3 - ActivityID : 802 , JobID : 7777  Task 4 - ActivityID : 803 , JobID : 7777   | Job Title | Company Name | Description | User ID | Status | JobID | | --- | --- | --- | --- | --- | --- | | SDE | Amazon | Software Dev | 1 | In Progress | 9999 | | Product Manager | Google | PM | 1 | In Progress | 8888 | | Product Manager | Google | PM | 2 | In Progress | 7777 | |  |  |  |  |  |  |   Activities Table   | Category | Description | Start Date | End Date | ActivityID | JobID (foreign key Jobs Table) | | --- | --- | --- | --- | --- | --- | | Code Challenge | Coding round |  |  | 111 | 9999 | | Leadership Round |  |  |  | 122 | 9999 | | Final Round |  |  |  | 144 | 9999 | | Code Challenge |  |  |  |  | 8888 | | Leadership Round |  |  |  |  | 8888 | | Final Round |  |  |  |  | 8888 | | Culture Fit |  |  |  |  | 7777 | | Pair Programming |  |  |  |  | 7777 | | Coding Round 2 |  |  |  |  | 7777 | | Final On Site interview |  |  |  |  | 7777 | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

* Review Risk Management SPPP
* Reviewed SDD Document and sections from SDD - including Software Architecture , Security Design , Class Diagram, Security Design , References, Glossary , Database Design
* Created a plan for Iteration 1 presentation and touched base on the Progress Report
* Decided to create the Demo on Monday night once progress on deliverables is completed

**Tasks -**

Chinmay : Readme , LucidChart Diagram , React app Component and View Jobs Logic

Bharat : Database Design , Class Diagram , Backend Table Setup and getJobs and createJob route , Presentation , Testing Framework for Backend

Taina : SDD Software Architecture MVC , UI Design, Sequence Diagram for Business Logic , React app Component and View Jobs Logic

Rachel : Class Diagram , Demo , Database tasks with Bharat ,

Ben : Frontend Tests , STD Automated Testing Report and Metrics ,

**Meeting 2**

**Date and Time:** 5/21/2022 1pm - 3pm (2 hours)

**Place**: Google Meeting

[meet.google.com/gwo-fziu-awc](http://meet.google.com/gwo-fziu-awc)

**Participants:** Bharat Gogineni ,Ruiqi Chang, Chinmay Bhelke, Benoît Clemenceau, Taina Conde

**Minutes taker:** Taina Conde

**Timekeeper:** Ruiqi Chang

**Purpose:** Work on Project feedback and discuss app design

**Agenda:**

1. Feedback on submitted documents

2. Deployment plan

3. Difference between our project and Huntr

4. STD and SDD for iteration1

5. Distribute tasks and user stories

6. Design and architecture

7. Points system in Pivotaltracker - how much work does 1 point mean?

**Discussions:**

1. **Feedback on Submitted documents**

* Discussed and changed the SPPP where the professor left some feedback.
* Added iteration 1 timeline
* Nonfunctional requirements test
* Risk management - sheet. Figure out what calculation is wrong. Decided to ask the professor what is wrong with that or what is the logic between low priority and high risk.

1. **Deployment plan**

* We are planning to deploy the application on AWS.

1. **Difference between our project and Huntr**

* We will focus on being a personal calendar for job applications instead of being a job card dashboard. We would have a single board with timelines and the next steps for each job application.

1. **STD and SDD for iteration1**

* We just pre-assigned some tasks to each team member. We will take a look further at those documents next week.

1. **Distribute tasks and user stories**

* We will start developing the user authentication first: user registration, login, and log-out features.
* Then, we will add the features of creating and deleting jobs.
* To sum up, for iteration1 we will work on the user stories related to user accounts, and adding and deleting job applications.

1. **Design and architecture**

* The project will have a landing page, where information about FollowUp will be displayed to the user with sign-in and sign-up buttons.
* After the user logs in the app shows a board with his job applications on a calendar basis. There will be an Add job button.
* When the user clicks the add job button, it will open a popup (modal) with a form so that the user can fill out the job information. The form will have fields for company, salary, activities (tasks), and deadline. We can use an API called DiscoverOrg to populate the company information.

1. **The points system in Pivotaltracker**

* 1 point in Pivotaltracker means 4 hours, 2 points mean 8 hours, 3 points 12 hours, and so on.

**Key Decisions:**

* User authentication is the first user story to complete - user registration, login, and log out.
* For the iteration1 will just write the ability to create jobs and delete jobs
* Bharat will build the sample testing structure
* The entities in our project are: user, job, and activity
* Assign the SDD file to team members
* 1 Point in Pivotaltracker is equivalent to 4 person-hours.
* We are using AWS to deploy the application
* Ask the professor what is wrong with the risk management calculation
* The difference between our app and Huntr is that FollowUp will focus on being a personal calendar for job applications instead of being a job card dashboard.

**Action Items:**

* UI design - Ruiq and Taina
* Software architecture - Chinway and Bharat
* Database Design - Bharat

**Meeting 1**

**Date and Time:** 5/14/2022 1pm to 5pm(4 Hours)

**Place**: Zoom Meeting and Google Meeting

<https://zoom.us/j/95175633072?pwd=UFNDdFdiREh6ZnkvTUY5NlBOblQxUT09>

<https://meet.google.com/vut-yffs-hxk?authuser=1&hl=en>

**Participants:** Bharat Gogineni ,Ruiqi Chang, Chinmay Bhelke, Benoît Clemenceau, Taina Conde

**Minutes taker:** Ruiqi Chang/ Chinmay Bhelke

**Timekeeper: N/A**

**Purpose:** Project Kickoff Meeting

**Agenda:**

* Introduce each other
* Decide what language and frame we use in the project
* Set up a communication and meeting schedule.
* Brainstorm the project
  + Determine an approach/process and environment to use
* Discuss risks
* Decide the team name. Assigned team leaders, other leader roles and responsibilities for each role.
* Iteration 0 (Proposal) Group Submission file

**Discussions:**

* Introduce each other
  + Discuss the strengths and weaknesses of each team member
* Decide what language and frame we use in the project
  + Based on the team members interests and experience, find the common language and frame for front end and back end
* Set up a communication and meeting schedule.
  + There will have at least weekly meetings to update the project process
  + Normally, the team discussion will be on Discord and group chat.
* Brainstorm the project
  + First project idea
    - Web Application for Group Creation for CS673

User logs in

CICD for semesters

Groups are created based on timeline and interests(Language)

for ex user chooses JavaScript as interests and they are put in the JS group

* + - Photo Gallery Application with Features

Authorization for user

User can login and view and upload images

Functionality to download zip file of all images and index the images

* + - Job Tracker Application

Target to the people who are looking for job

User logs in , has list of tasks and component/card for every jobs which want to apply

Track the stage of each job: submit resume? First interview?

Need database?

* + Determine an approach/process and environment to use
    - Includes management tools and development environment.
* Discuss risks
  + Keep it simple/limit scope creep
  + What differentiates us from google sheet
* Develop progress
  + Project Management tool
  + Waterfall? Scrum? Agile?
  + What platform we use for this project?
* Iteration 0 (Proposal) Group Submission file
  + Decide the team name. Assigned team leaders, other leader roles and responsibilities for each role
  + Assign the project files to each team member

**Key Decisions:**

* Team name: A-Team
* Project name:FollowUp
* Project: Job Tracker Application
  + Front end- TypeScript and React
  + Back end- JavaScript and Express
  + We need backend DB running to store info for rows/tasks
  + REST API to transmit data from frontend to backend
  + Use GIT as the version control tool and github (https://github.com/) to host your project.
* Time tracking
  + Week start on Tuesday
  + Week end on Monday
  + Get time to Bharat by noon on Monday
  + Every week meeting will hold on Saturday.
* Communication Plan
  + Use discard group and text group for communication
  + Use Jira or pivotaltracker for task tracking (to-do and complete)
  + Use Git and GitHub for document and code repository, version control
* Roles assigned:
  + Bharat Gogineni: Team leader
  + Ruiqi Chang: Requirement Leader
  + Chinmay Bhelke: Design and Implementation Leader; Security Leader
  + Benoît Clemenceau: QA Leader
  + Taina Conde: Configuration Leader

**Action Items:**

* Review TypeScript and JavaScript of project -Bharat, Ruiqi,Chinmay,Benoît, Taina
* Finish SPPP -Bharat,Chinmay,Benoît, Taina
* Risk Management -Chinmay
* Meeting Minutes files -Ruiqi
* progress report-Bharat
* Slides and presentation-Benoît, Taina
* Submit time to Bharat by noon Monday-Ruiqi,Chinmay,Benoît, Taina