# SWE 606: Project Documentation

## Group 8 - CSEGSA Event Portal

Abhinand Sai Nasari (131005844)
Himanshu Singh (832001528)
Kiet Minh Nguyen (132005430)
Waris Quamer
**Client: TAMU CSEGSA - Farabi Mahmud(farabi@tamu.edu)**

## 1. Project Summary:

The CSEGSA organization has been missing a modern and user-friendly platform to display events for members as well as a centralized job portal to provide students (who are the main stakeholders) with opportunities in the job search journey. Realizing a need for that, our team built a brand new portal for the CSEGSA organization. It contains the following main pages:

- Home Page: introduce the organization and list all the officer's information together with their contact information.
- Events Page: display a calendar showing upcoming events and allow logged-in students to RSVP for events. At the same time, it allows administrators to set up new events, keep track of participants' headcount
- Job Posting: We often receive emails about job openings from the graduate advisor or from the alumni. We wanted a common platform where students could log in and see all the openings that are available.
- Login Page: authenticate only @tamu.edu emails to provide security and ensure only TAMU students can view and register events. Special admin access to modify job postings, add events and administer the site.

## 2. User Stories:

- As a Student, (10pts)
  I want a calendar highlighting all the important events of the month,
  So I can plan my schedule accordingly and RSVP for the same.
- As a Student, (9pts)
  I want a page containing all the current job/internship openings,
  So I can apply for them, get salary insights, and approach alumni for referrals.

- As CSEGSA admin, (18pts)
  I want 3 tiers of permissions - admin, alumni, and students,
  So that admin can manage content, alumni can sign up for referrals, and students can ask for referrals.
  - **Implementation Status** - Here instead of three tiers of permissions we have only provided two tiers of permissions i.e., Admin and User.
- As CSEGSA admin, (13pts)
  I want an **about & contact us** page,
  So that students can learn more about the organization and get in touch with us.

**Lo-Fi Diagrams with their actual web pages:**

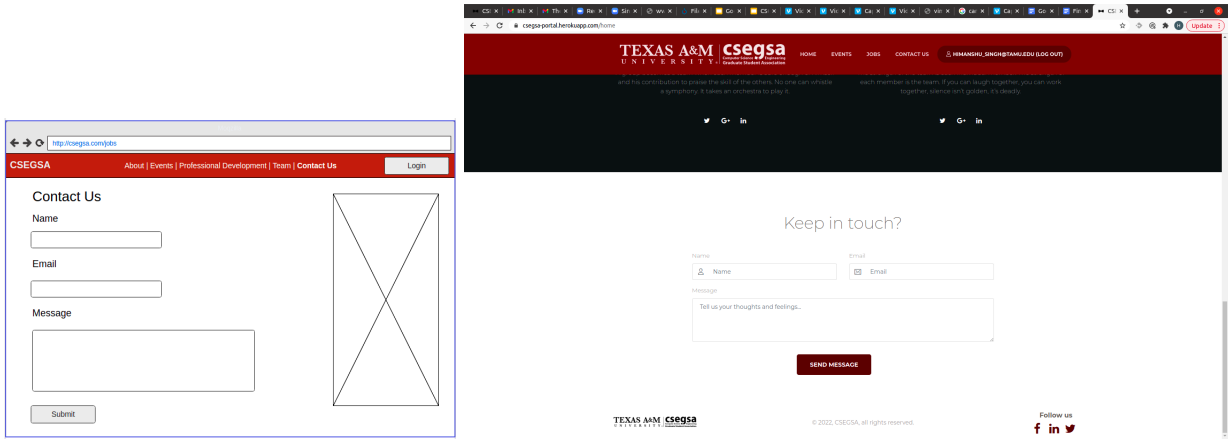


Fig.1: Contact us page

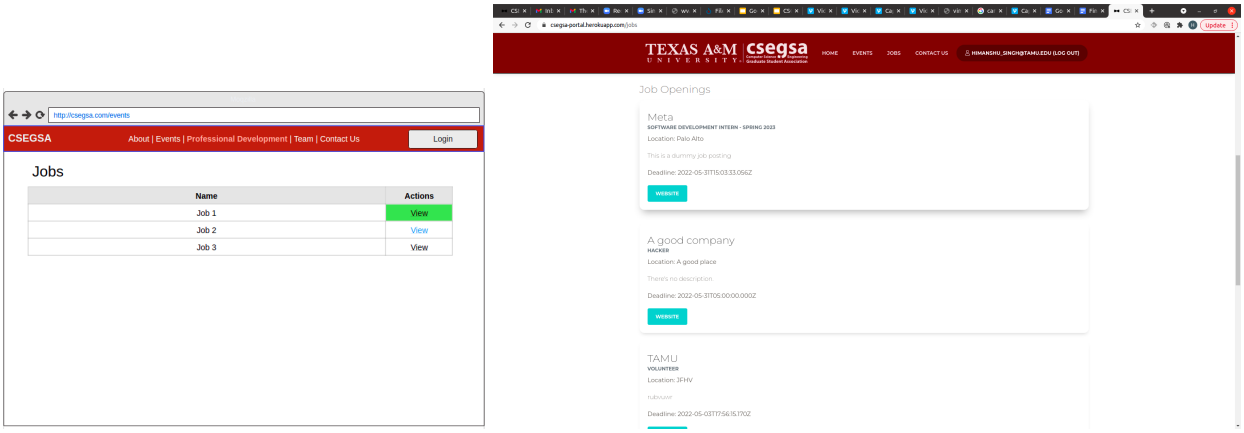


Fig. 2: Jobs Page

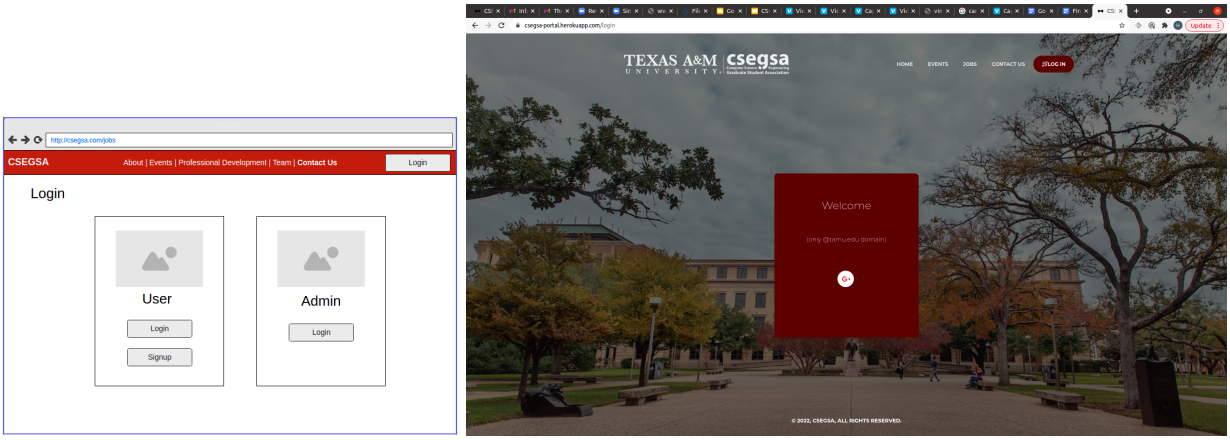


Fig. 3: Login Page

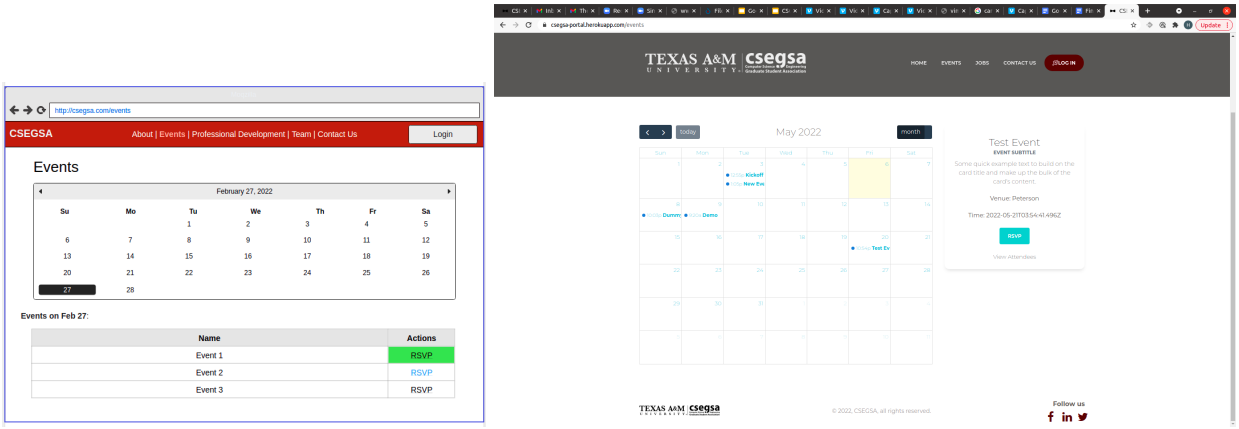


Fig. 4: Events Page

## 3. Team members and role allocation:

Our team members split the work based on our most comfortable skill sets

- **Scrum Master** & Frontend: Himanshu Singh
- Frontend and Testing: Waris Quamer
- Backend, Authentication and Testing : Abinand Sai Nasari
- Frontend, Authentication and **Product Owner**: Kiet Nguyen

## 4. Iteration Details:

- **Iteration 1: (12pt) - We created a basic layout of the website based on the lo-fi diagrams and presented it to the client.**
  - Homepage Section 3: Contact us Form - 2pt
  - Write Cucumber Test cases and Selenium Automations - 3pt
  - TAMU Themed Navbar for Web App - 1pt

        - Homepage Section 2 : CSEGSA About Us section - 2pt
        - Footer design with logo and social media handles - 2pt
        - Basic outline of Events page with Event Calendar - 2pt

    - **Iteration 2 ( 8 Points): - Finalized the design and deployment strategies and deployed the website on heroku**
        - Set up CD for the project on Heroku Via github Actions - 3pt
        - Homepage: Section 1 - Photo collage with CSEGSA highlights - 1pt
        - Login Integration with the frontend and assign user role - 2pt
        - Login API.  -2pt

    - **Iteration 3: (16 Points) - Created APIs to perform CRUD operations on jobs and events and made changes to the calendar design and contact us form.**
        - CRUD API for events -2p
        - Job Description List - 2pt
        - Add API call to send email to admin via contact us form - 2pt
        - View Posted jobs - 2pt
        - Events Calendar Week View, option to change to month view via toggle button  - 2pt
        - Event Page: Events details  and option to rsvp - Iteration 3 - 2pt
        - Admin Secured Routing -2pt
        - Test cases for Events and Jobs API -2pt

    - **Iteration 4 (24 points): Worked on securing the API calls, Authorization and Admin Role management.**
        - Admin View: Add a Job - 2pt
        - CRUD API for Roles - 3pt
        - Admin Portal: Add and Delete Roles - 3pt
        - Admin NavBar Display only to Admin - 2pt
        - Admin Portal: Display Cards for all roles - 2pt
        - CRUD API for job postings - 2pt
        - Admin View: Add Events - 2pt
        - Admin View: Add officers details - 2pt
        - Contact us page - Clear form and alert - 1pt
        - API Authentication and Authorization - 3pt
        - Homepage Section 3 : CSEGSA officers details - 2pt

## 5. Meetings with clients:

- Feb 21 : Met with Sanjeevani to pitch our idea. We came up with our initial proposal of a job referral platform that supports discussions about salary

and reviews of work/companies. After our discussion on potential drawbacks and certain compliance issues, we modified our projects into a CSEGSA portal for internal events and activities with the potential to support job postings from the department and other online job search platforms.

- Mar 15: Met with Farabi in person to show our first draft of the portal. We had the initial skeleton of the website with 4 main tabs (Home Page, Events, Job Post, log in) without any working functionality yet.
- Mar 31: Met with Farabi to show progress: working login page that authenticated only @tamu.edu domain. The events page can set up and display events.
- Apr 19: Met with Farabi to show all the features combined and deployed on heroku. The features included login page, job posting, events page and contact us form. We were asked to integrate email features on contact us form and we finalized the design of the admin portal.
- May 5: Met with Farabi to show the final set of features (new admin page) and the functionalities that we could implement in the future.

## 6. BDD/TDD process and benefits:

We used **Cucumber** for **BDD** development, **Jest** for unit testing and **Postman Collections** to test our backend API.

**Benefits:**

- Writing the test first gave a clear picture of what is required to implement.
- It also helped in deciding the order in which things can be implemented incrementally.
- API tests ensure that other dependent functionalities need to be made available before proceeding with the API implementation.
- Running the tests before every commit helped in ensuring the changes were not breaking the existing functionality which would have been noticed late.

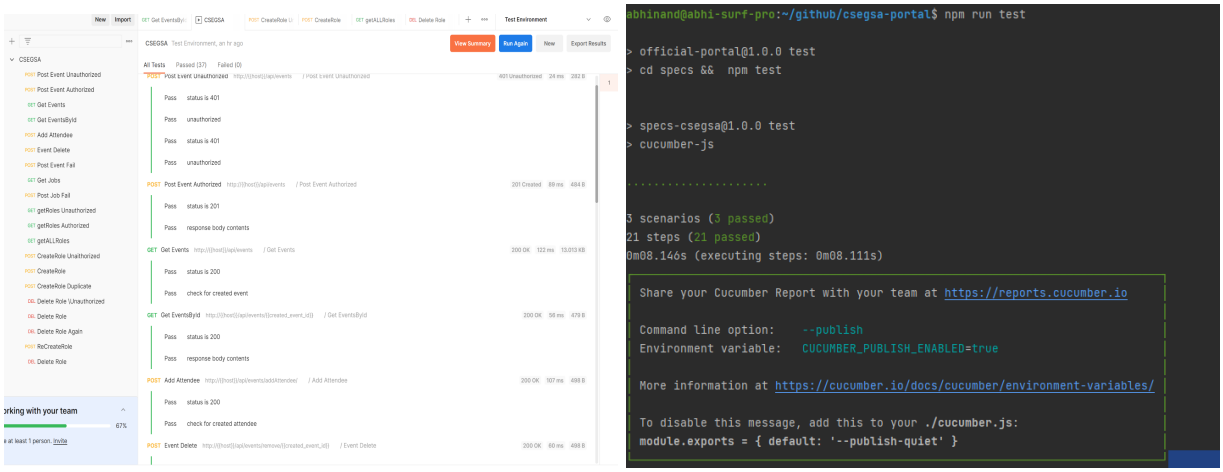


Fig. 5: Postman and Cucumber Test Runs

**Implementation issues:**

Some of the scenarios are hard to test like flows where the user action is not easy to emulate. Implementing the test took considerable time and sometimes the effort was more than the feature implementation. This is not exactly a problem of the process but the trouble we got into.

## 7. Approach to Configuration management:

- In terms of configuration management, we had a .env file with the API keys for all services we were using. This .env file was stored in the **Heroku** environment. Git ignore directives were put in place to avoid accidentally committing any sensitive configuration.
- We did spikes for **Api Authentication** and **Authorization.**
- We iteratively merged each feature and released it. We worked on forked repositories to avoid a single person accidentally overriding the main repository. A branch per each major feature from each contributor 12 branches and 8 releases.

## 8. Issues regarding the Heroku production release process:

- Recent incidents on **Heroku** made them disable the **GitHub integration** feature. Had to switch to **Heroku CLI** and this caused some issues with syncing it with the currently deployed **Heroku** main branch. Also, currently there is a notification that passwords will be reset for all accounts from **Heroku** and can affect **CI** action already integrated. We noticed that the deployed version of the application often missed the configuration

variables. Apart from these, the release process was smooth and we were able to release new features smoothly and *heroku logs —tail* was sufficient to debug the release build issues.

- Some other common release build failures are npm packages saved in package.json as dev-dependencies caused build and startup issues of the server.

## 9. Issues with AWS and Github:

We used **Github** for our version control. One issue we came across was the setup for both frontend and backend code. We initially created a repository for the frontend and intended to create a different one for backend code. Later, we decided to merge the 2 under one repository with different subfolders. Recreation of **Github** repositories a few times to accommodate code changes and set up directory structures.

## 10. Other tools:

- **EsLint** was used to enforce the coding standards of the javascript. **SonarLint** Plugin was also used to code suggestions and warnings. **CodeClimate** is integrated to the **Github** and it runs forever Pull Request and gives the Code Duplications introduced and only addressing those issues the PRs are merged.
- This process ensures that a common style/standard is maintained (using single quotes vs double quotes, variable naming conventions, etc..) and code quality such that the cyclomatic complexity of a function is not too high. Warnings about unused and dead code helped us get rid of it. This helps in maintaining the code quality and can be used to enforce improvement of these metrics as a requirement for each PR.
- For Design Documentation **PlantUML** is used , this makes generating the diagrams easy and any modifications to the design can be tracked by committing these design uml code to version control software. This came is handy to explain and present our thoughts in an easy and concise manner instead of using other software to draw thes design specifications.

## 11. Github repo and deployment process

We used **MERN stack** to build this project. To keep the development and deployment process simple the repository provided below contains both the frontend and backend code. The details to deploy on **Heroku** and start it locally

are provided in the README.md at the root folder. The Setup requires a Firebase project, MongoDB Atlas  instance and a Heroku account

The structure of the github repo:

- **Client** - contains the react scripts for the frontend
- **Server** - contains the scripts for the node express server
- **Spec** - Contains the scripts for BDD, TDD and API tests.
- **Documentation** - contains the progress report for each iteration and the final report.

** The readme file in our github repository contains the steps to recreate this project for subsequent developers.*

## 12. Deployment links:

**Updated links to your Pivotal Tracker, public GitHub repo, and Heroku deployment, as appropriate.**

- Pivotal Tracker: https://www.pivotaltracker.com/n/projects/2556123
- Public GitHub repo: https://github.com/csegsa/official-portal
- Heroku deployment: https://csegsa-portal.herokuapp.com/home

## 13. Poster and Demo links:

- Based on the instructions given in the poster submission assignment, we have combined out poster video and project demo video into one single file and have uploaded the same on vimeo.
    - Video Demo Link: https://vimeo.com/707115982
- The poster has been uploaded as a PPTX file on the public github repository under the documentation folder.

# 14. Design

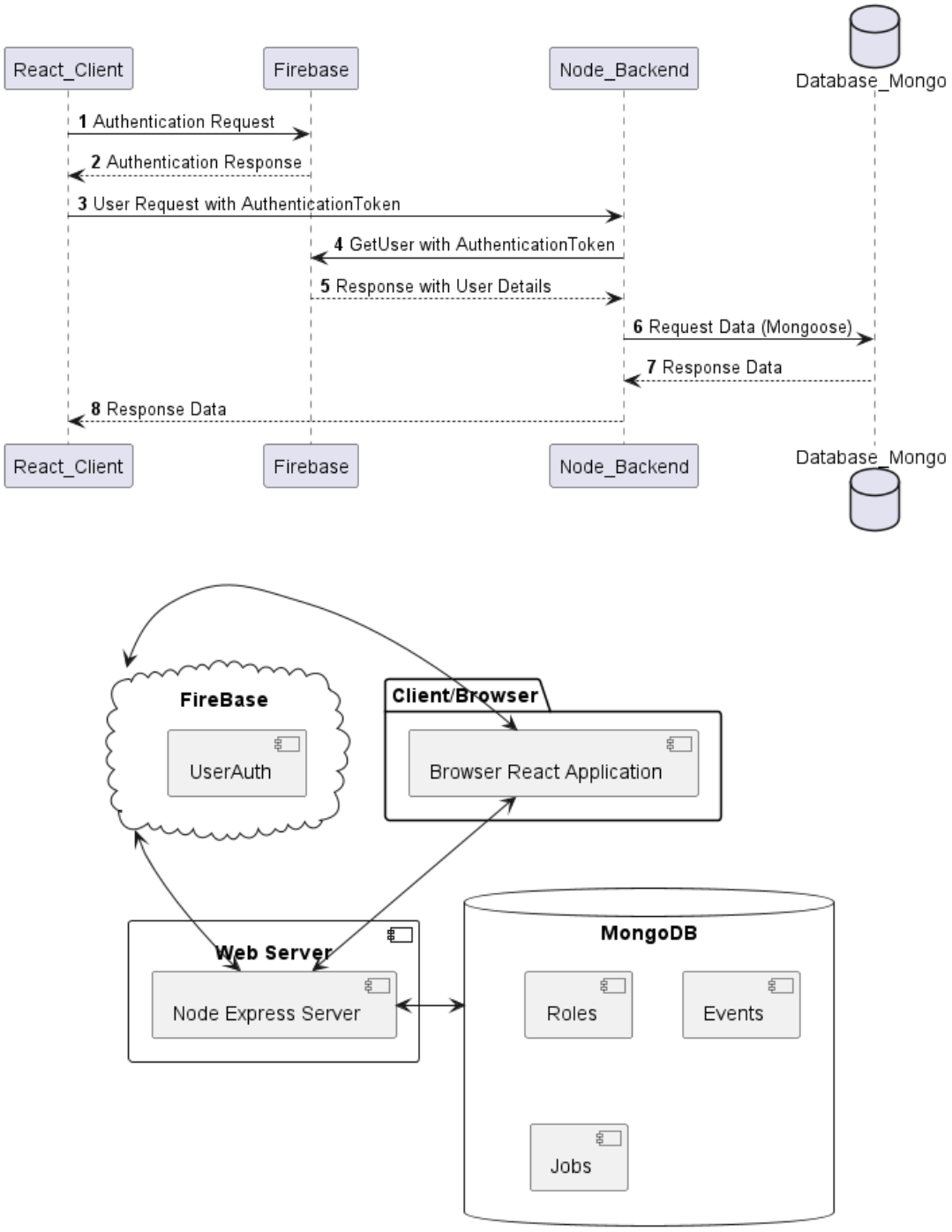


Fig. 6: Flow diagram and Flow diagrams