

## **CMPE 273: Enterprise Distributed Systems**

### **Lab 2: Using REST (Node.js), React JS, Redux, Passport, MongoDB and Kafka**

**Due: April 10th, 2020, 2:00 PM**

This lab covers designing and implementing distributed service-oriented application using Kafka. This lab assignment is graded based on 30 points and is an individual effort (Teamwork not allowed)

#### **Prerequisite**

- You should have prior knowledge of JavaScript, React, Redux, MongoDB, Passport, JWT token, AWS EC2
- You should be able to run Kafka sample example to be demonstrated in the class.

#### **Grading**

Handshake Application - 30 marks

Questions – 5 marks

Total – 35 marks

*Note: Weightage will be given to all the new features and concepts mentioned in Lab 2.*

*Late assignments will be accepted but will be subject to a penalty of -5 points per day late.*

*Submissions received at or before the class on the due date can receive maximum.*

# Handshake

You need to develop “Prototype of Handshake application”. This prototype will be a web application using React, Redux and Node with MongoDB as database. You need to use Kafka messaging service and make the system distributed. Refer Handshake website and see how it functions.

The application should have the following persona:

1. Student
2. Company

You need to implement the following features in your application for the roles given above.

1. Student Signup (name, email id, password, college name)
2. Company Signup (company name, email id, password, location)
3. Sign in
4. Sign out

A **Company** should be able to do the following functionalities:

## **Job Postings page (Dashboard – Landing page):**

1. Post Job openings (with job title, posting date, application deadline, location, salary, job description, job category – full time, part time, intern, on campus)
2. View list of job postings posted by them
3. View list of students applied for each Job posting posted by them.
4. Click and view profile page of each Student
5. Preview the Resume uploaded by student
6. Update the application status of each student – Pending, Reviewed, Declined.
7. **Pagination should be implemented.**

## **Company Profile Page:**

1. View its profile – having all its basic information (name, location, description, contact information, profile picture)
2. Update Company profile (name, location, description, contact information, profile picture)
3. Update Contact information

## **Students Tab:**

1. Search for students (using name, college name or skillset)
2. **Pagination should be implemented.**
3. Click and view profile of each Student

## **Events Tab:**

1. Post Events (event name, description, time, date, location, Eligibility – All, specific major)
2. View list of students who registered for each Event
3. **Pagination should be implemented.**
4. Click and view profile page of each Student

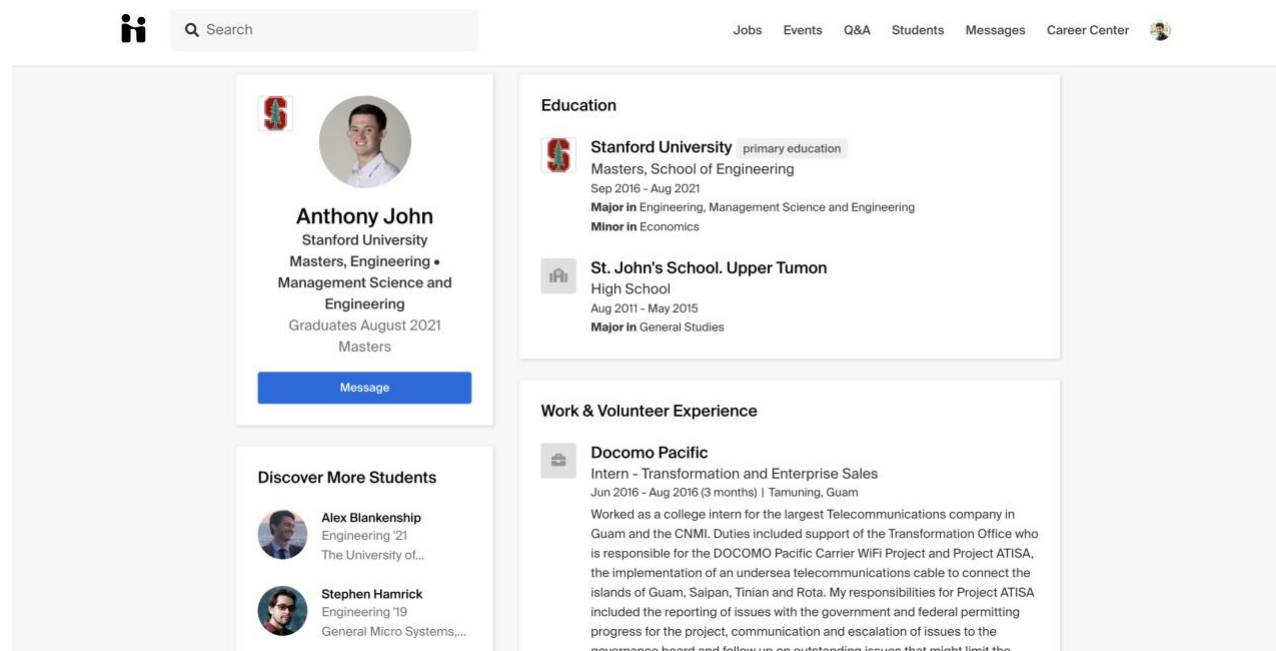
### Messages Tab (New):

1. Company should be able to message a student from student's profile page.
2. Company need not message other companies.
3. View the list of conversations it had with students.
4. Clicking on each conversation should display the entire message history with that student in time sorted order and should be able to continue the conversation by replying.

A **Student** should be able to do the following functionalities:

### Profile Page:

1. Display complete profile of a student (basic details, career objective, education, experience, skillset, profile picture)
2. Upload profile picture
3. Update basic details (name, date of birth, city, state, country) and Career Objective
4. Update **multiple** Education Details and delete any of them (College name, Location, Degree, Major, Year of passing, current CGPA)
5. Update **multiple** Experience details and delete any of them (Company name, Title, Location, Start & End dates, Work description)
6. Update Contact Information (email id, phone number)
7. Update Skillset information



### Job Search tab (Dashboard – Landing page)

1. Search for job postings (using company names or job titles)
2. Filter job search results based on category (Full time, Part time, On Campus, Internship) and location (city)
3. **Student should be able to sort the search results based on location/posting date/application deadline both ascending and descending.**

4. **Pagination should be implemented.**
5. Click and view a Job posting description
6. Click on the company name to view company profile (description and contact information).
7. Apply for a Job opening by uploading Resume document (PDF) and clicking Submit Application button


The screenshot displays a job search platform interface. At the top, there is a navigation bar with links for Jobs, Events, Q&A, Students, Messages, and Career Center. Below this is a 'Job Search' section with a search bar and filters. The search bar contains the text 'Job titles, employers, or keywords' and a location filter 'City, State, Zip Code, or Address'. Below the search bar, there are tags for various job titles like 'software', 'computer science', etc., and buttons for 'Full-Time Job', 'Part-Time', 'Internship', 'On-Campus', and 'Filters'. A 'My Favorite Jobs' button is also present. The main content area shows a list of job results. The first result is 'Software Engineering Summer Intern 2020' by Zanbato, Inc., located in Mountain View, CA. It is a Full-Time Internship from 6/6/20 to 9/6/20, paid, and posted on Aug 14. The job description includes a deadline for applications (January 1st, 2021 at 4:35 pm) and a green 'Apply Externally' button. Below the job title, it states 'You match all of Zanbato, Inc.'s preferences' with checkboxes for Major, GPA, School Year, and U.S. Work Authorization. The description also mentions that the employer will sponsor a work visa and accepts OPT/CPT. The job details include 'Learn something everyday building real products for real users' and 'What you will do' with bullet points: 'Ship code to users in your first week and every week after' and 'Take ownership and lead your own project within 8 weeks'.


## Applications Tab

1. View list of all the job postings applied (along with application date, application status)
2. Filter the applications based on the application status – Pending, Reviewed, Declined
3. **Pagination should be implemented.**

## Events Tab

1. View list of upcoming events in the order of increasing date.
2. Search for event (using event name)
3. **Pagination should be implemented.**
4. Click and view event details
5. Register for an event (only if eligible)
6. View list of registered events




[Jobs](#)
[Events](#)
[Q&A](#)
[Students](#)
[Messages](#)
[Career Center](#)


---


[Upcoming Events](#)
[Event Search](#)
[Fair Search](#)
[Calendar](#)

Events today




**Bank of America Campus Connect Winter Webcast Series #CAMPUSCONNECT**  
 January 14 to March 13, 2020  
[View Details](#)

[View Event](#)  
 232 students registered  
 Virtual Session




**Bank of America Campus Connect Winter Webcast Series**  
 February 12 to March 11, 2020  
[View Details](#)

[View Event](#)  
 944 students registered  
 Virtual Session




**CAMPUS EVENT: Travel Around LCOB Day/ LCoB**  
 March 3, 2020 from 1:00 pm to 3:30 pm  
 Student Union, Ballroom, 211 S 9th St, San Jose, CA 95112, United States  
[View Details](#)

[View Event](#)  
 2 students registered  
 Workshop



**CAMPUS EVENT: Data Science for All Seminar -**  
 ...  
[View Event](#)

Your Upcoming Events



**Friday Photo Booth / LinkedIn Profile Review**  
 MAR 20 09:00 AM - MAR 20 12:00 PM  
 Organized by San Jose State University

Your Upcoming Appointments

You have no upcoming appointments.

[Request an appointment](#)

Your Upcoming Career Fairs

You have no upcoming career fairs.

[Find some career fairs](#)

Your Upcoming Interviews

You have no upcoming interviews.

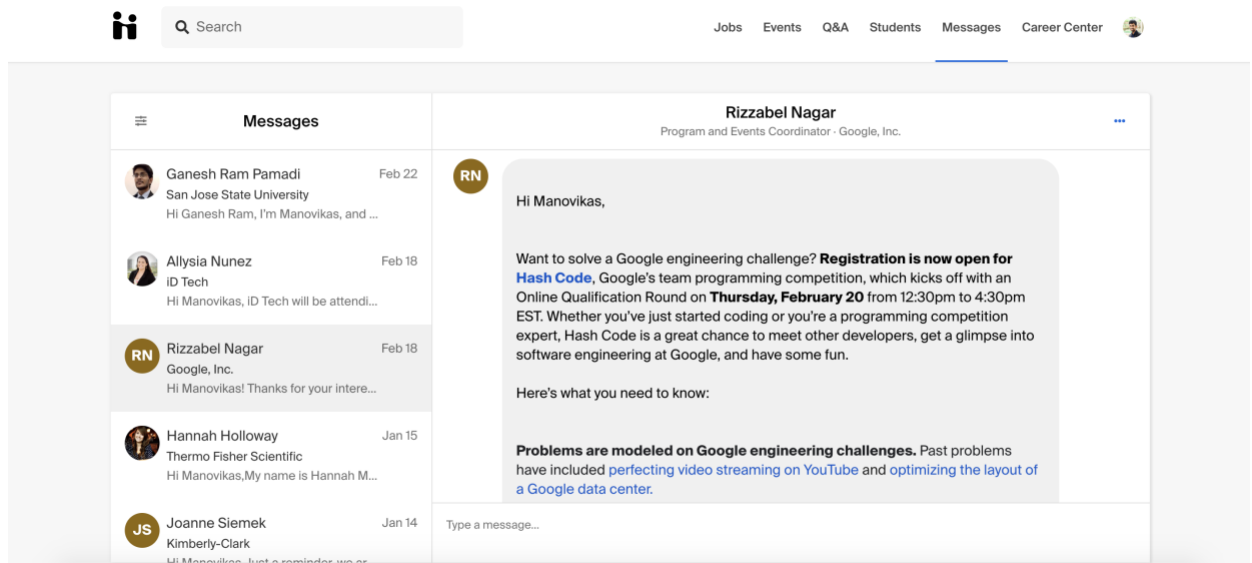
[Apply to some jobs](#)

## Students Tab

1. View list of students enrolled in Handshake
2. Search for students (using student name or college name)
3. Filter the results based on their Major
4. **Pagination should be implemented.**
5. Click on Student details to view Student Profile

## Messages Tab (New):

1. Students can message other students from their profile.
2. Student should not be able to initiate a conversation with a company.
3. Student can view messages and reply to company only if the conversation was started from Company side.
4. View the list of conversations he/she had in Messages Tab.
5. Clicking on each conversation should display the message history in sorted order and student should be able to continue the conversation by replying.



- Every service should have proper exception handling and every input field should have proper validation.
- Password values should be encrypted.
- **Passport.js and JWT Tokens should be used for authentication.**
- **MongoDB with connection pooling should be used.**
- **Redux should be implemented in all the components.**
- **Kafka should be used.**
- ESLint should be used in your code following Airbnb style guide.
- The application should be deployed on cloud (E.g. Heroku, AWS EC2)
- A simple, attractive and responsive client attracts good marks.

## Testing

1. Testing of the backend server should be done using JMeter and Mocha.
  2. Enzyme should be used to test the frontend at least 3 views/pages.
- Following tasks to be tested using JMeter:  
Test the server for **100, 200, 300, 400 and 500 concurrent users** with and without connection pooling in database. Draw the graph with the average time and include it in the report.
  - Following tasks to be tested using Mocha:  
Implement five randomly selected REST web service API calls using Mocha. Display the output in the report.

## Questions

1. Compare passport authentication process with the authentication process used in Lab1.
2. Compare performance with and without Kafka. Explain in detail the reason for difference in performance.
3. If given an option to implement MySQL and MongoDB both in your application, specify which part of data of the application you will store in MongoDB and MySQL respectively

## Code Repository

- Use Bit bucket for saving your code and keep it private.
- In your repository, create sub-folders, Place all your source code in respective Folders.
- Add a proper description for every commit describing what code changes are added.
- Regular commits to your repository are mandatory. (Penalty of 3 marks if missed).
- Do not submit dependencies or supporting libraries (e.g. node\_modules) (including them causes deduction of 2 marks).
- All the dependencies should be included in package.json file.
- Readme file of your repository should contain the steps to run the application.

## Project Report

- Introduction: State your goals and purpose of the system
- System Design: Describe your chosen system design
- Results: Screen captures of testing results and important screens of the application.
- Performance: What was performance? Analyze results and explain why you are getting those results.
- Commit history – screen capture
- Answers to the questions.

## Submission

Please upload your report (John\_Lab2\_Report.doc) on Canvas before deadline. (Number of pages in Report should be below 30)