CMPE 273: Enterprise Distributed Systems

Lab 1 Assignment: Using REST (Node.js) and React JS

Due: March 6th, 2020, 2:00 PM

This lab assignment covers developing REST services using Node.js (Express) and ReactJS. This lab assignment is graded based on 30 points and is an individual effort (Teamwork not allowed)

Prerequisite
● You should be able to run the basic node.js and React application discussed in class.
● You must know programing language basics, JavaScript.

Grading
Handshake Application - 30 marks
Questions – 5 marks
Total – 35 marks
Note: 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 and Node. 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.

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. 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. Click and view profile page of each Student

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 Education Details (College name, Location, Degree, Major, Year of passing, current CGPA)
5. Update Experience details (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. Click and view a Job posting description
4. Click on the company name to view company profile (description and contact information).
5. Apply for a Job opening by uploading Resume document (PDF) and clicking Submit Application button

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

Events Tab

1. View list of upcoming events in the order of increasing date
2. Search for event (using event name)

3. Click and view event details
4. Register for an event (only if eligible)
5. View list of registered events

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. Click on Student details to view Student Profile

- Every service should have proper exception handling and every input field should have proper validation.
- Password values should be encrypted.
- 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 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. 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 the results of graphs with and without in-built mysql connection pooling of database. Explain the result in detail and describe the connection pooling algorithm if you need to implement connection pooling on your own. (2 pts)
2. What are the ways to improve SQL Performance? List at least 3 strategies. Explain why those strategies improve the performance. (2 pts)
3. Is your session strategy horizontally scalable? If **YES**, explain your session handling strategy. If **NO**, then explain how you can achieve it. (1 pts)

Git Repository
o In your Git repository, create two sub-folders, one for Frontend and one for Backend. Place all your source code in respective Folders.
o Add a proper description for every commit describing what code changes are added.
o Regular commits to your repository are mandatory. (Penalty of 3 marks if missed).
o Do not submit dependencies or supporting libraries (e.g. node_modules) (including them causes deduction of 2 marks).
o All the dependencies should be included in package.json file.
o Readme file of your repository should contain the steps to run the application.

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

Submission
Please upload your report (John_Lab1_Report.doc) on Canvas before deadline. (Number of pages in Report should be below 30)