

CMPE273: Enterprise Distributed Systems

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

Due: March 10th, 2019 11:59 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 (**No teamwork is allowed**)

Prerequisite

- You must have carefully read the Environment Setup document. You should be able to run the “Hello World” node.js application discussed in class.
- You must know programming language basics, JavaScript.

Grading

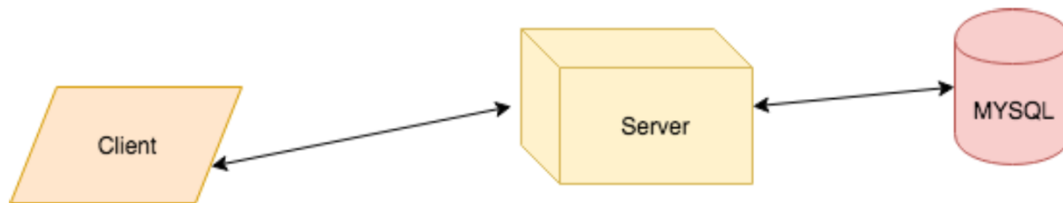
Part 1 - Calculator - 5pts

Part 2 – Canvas Application - 18 pts

Questions - 7pts

Total - 30pts

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.



Part 1 - Calculator

Server - demonstrate RESTful Services [2 pts]

The first Node.js based server you need to develop is the “Calculator”. This server should perform the following tasks:

1. Addition
2. Subtraction
3. Multiplication
4. Division

Client - [1 pts]

Make attractive and simple User Interface to perform above operation.

Note: Client can take values from users and send to a server to perform the required operation. A Server will process the inputs and send output back to a client. A Client will display output to users.

Testing should be done using JMeter: [2 pts]

Automate the processes using JMeter and print the average time required for below operations:

1. Start a timer
2. Invoke 1,000 calculator calls on randomly selected tasks.
3. Stop the timer and print out the average time to perform each operation

1. Start a timer
2. Invoke 5,000 calculator calls on randomly selected tasks.
3. Stop the timer and print out the average time to perform each operation

1. Start a timer
2. Invoke 100 concurrent users with 1000 calls each to calculator on randomly selected tasks.
3. Stop the timer and print out the average time to perform each operation

Your output should be one single client run displaying all the requirements. You need to display results of calls to server. ***Draw a graph with average time and include it in the report.*** Compare the performance from server and discuss results.

Part 2 - Application

Server - demonstrate RESTful Services (8 pts)

The next node.js based server you need to develop is the “Prototype of **Canvas** application”. Everyone should refer the Canvas application and see how it functions.

This server should perform the following tasks:

a) Basic **Users (Student & faculty)** functionalities:

1. Sign up new user (Name, Email and password)
2. Sign in existing user
3. Sign out.
4. Profile (Profile Image, Name, Email, Phone Number, About Me, City, Country, Company, School, Hometown, Languages, Gender)
5. Users can update Profile anytime.

To use the system, a user must login first to the system. Password must be encrypted.

b) **Faculty:**

1. Faculty should be able to create a course with following fields
 - a. CourseId
 - b. CourseName
 - c. Course Dept
 - d. CourseDescription
 - e. CourseRoom
 - f. CourseCapacity
 - g. Waitlist capacity
 - h. courseTerm
2. Faculty should be able to give permission codes for the waitlisted students. (A provision must be made for the faculty to generate random codes which can be onetime use only)

c) **Student:**

1. Student should be able to search for all the courses by term, by id or by course name. Should have filter like id greater than a particular value (Eg: All courses greater than id: 200 for CMPE dept.)
2. Student should be able to enroll for a course, drop a course and waitlist a course when full.

c) **Home**

1. Student should be able to view all the courses he/she has registered.
2. Faculty should be able to view all the courses created by them.

d) **Course details:**

Student:

- Student can view his/her grades in the course.
- Student should be able to submit his/her assignment.
- Student should be able to view all his submissions per assignment.
- Student should be able to take quiz on .
- Should be able to view announcements
- Should be able to view all the people registered for that course.
- Should be able to download lecture notes and files

Faculty:

- Faculty should be able to create Assignments and Quizzes.
- Faculty should be able to view and download submissions from students.
- Faculty should be able to make announcements
- Faculty should be able to view all students registered for that course.
- Faculty should be able to grade assignment submitted by students.
- Should be able to upload lecture notes and files
- Should be able to remove a student from the course.

e) Should perform connection pooling (in-built in MySQL) for database access.

Sample Screenshots for faculty pages (No need to implement filter, just show list of announcements)

FA18: CMPE-273 Sec 01 - Ent Dist Systems > Announcements

Fall 2018

Home

Announcements

Assignments

Discussions

Grades

People

Pages

Files

Syllabus

Outcomes

Quizzes

All

Search


🔒

🗑️

+ Announcement

External feeds

☐




Steps to clean up your repository

[All Sections](#)

Hope you are doing well. With regards to your Git Repository please follow the followin...

Posted on:
Dec 19, 2018 at 1:01pm

☐




Final Exam Tips

[All Sections](#)

What to expect? The focus will be on application knowledge that you have gained throu...

Posted on:
Dec 16, 2018 at 5:01pm

☐



Team Project

[All Sections](#)

The grades received for your team project is not only dependent on test-cases. There ar...

Posted on:
Dec 9, 2018 at 9:46pm

Fall 2018

Home

Announcements

Assignments

Discussions

Grades

People

Pages

Files

Syllabus

Outcomes

Quizzes


Search for Assignment

+ Group

+ Assignment


⋮

▼ Assignments




Refresher Homework

Closed | Due Sep 7, 2018 at 11:59pm | 25 pts




Node Hom

Closed | Due Sep 11, 2018 at 2pm | 10 pts




Quiz 2

10 pts



ReactJS Homework

Due Sep 18, 2018 at 2pm | 10 pts



MySQL Homework

SJSU

Fall 2018

Home

Announcements

Assignments

Discussions

Grades

People

Pages

Files

Syllabus

Outcomes

Quizzes

Account

Dashboard

Courses

Groups

Calendar

📄

FA18: CMPE-273 Sec 01 - Ent Dist Systems > Files

Search for files

0 items selected

+ Folder

Upload

▼ FA18: CMPE-273 Sec 01 - Ent Dist Systems

► Demos

► Group Project

► Homework

► Honesty Pledge

► Lab 1

► Lab 2

► Lab 3

► Lecture Notes

Name	Date Created	Date Modified	Modified By	Size	
Demos	Aug 28, 2018			--	✓
Group Project	Oct 23, 2018			--	✓
Homework	Sep 4, 2018			--	✓
Honesty Pledge	Aug 21, 2018			--	✓
Lab 1	Sep 4, 2018			--	✓

No need to add any extra feature like submission comments. Just stick to the requirement.

The screenshot shows a web application interface for 'Refresher Homework'. The top navigation bar is blue and contains the title 'Refresher Homework', a progress indicator '82/83 Graded', and a '1/83' page indicator. Below the navigation bar is a toolbar with various icons for navigation and editing. The main content area is white and displays the title 'Refresher Homework' in bold. Below the title, there is a section for 'JavaScript' with a description of functions and a problem statement: 'Create JavaScript function to find product of two number using functions.' The solution is provided in a code block with the following HTML code:

```
<!DOCTYPE html>
<html>
<head><title>
  Refresher Functions
```

 On the right side of the interface, there is a sidebar with submission information, including the submission date and time, the student's name, and the grade out of 25. There is also a section for 'Assignment Comments' with a text input field and a 'Submit' button.

The Service should take care of exception that means validation is extremely important for this server. **Proper exception handling and prototype similar to actual application would attract good marks.**

Client - [7 pts]

A client must include all the functionalities implemented by the web services. Develop the Client using HTML5 and ReactJS. A Simple, attractive and Responsive client attracts good marks.

Note: Every field in an entire project must have validation. User's Name (Navigate to Profile) and Property Name (Navigate to Property Details view) must have hyperlinks.

Testing of the server should be done using JMeter and Mocha.

Mocha is a node.js testing backend APIs.

Enzyme for testing at least 3 views/pages.

1. Following tasks to be tested using JMeter: (2 Points)

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.**

2. Following tasks to be tested using Mocha: (1 Point)

Implement five randomly selected REST web service API calls using Mocha. **Display the output in the report.**

Create a private repository on the GitHub or bitbucket to manage source code for the project. Add a description for every commit. A description should consist of a one-line overview on what is committed. Include GitHub/bitbucket project link with access credentials in your report. Regular commits to your repository are mandatory. Include GitHub /bitbucket commit History to your report.
(Penalty for not including commit history would be 3 points).

Questions (7 pts)

1. Explain the encryption algorithm used in your application. Mention different encryption algorithms available and the reason for your selection of the algorithm used.
2. 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.
3. What is SQL caching? What all types of SQL caching are available, and which suits your code the most. You don't need to implement the caching, write pseudocode or explain in detail.
4. Is your session strategy horizontally scalable? If **YES**, explain your session handling strategy. If **NO**, then explain how you can achieve it.

Deliverables Required (Git Deliverables):

- Create a private repo with the format "Lab1 - <Student ID>" (eg. Lab1 - 123456789)
- Inside the repository create two folders, one for Calculator and One for Canvas
- Inside each of these folders create two sub-folders, one for Frontend-Code and one for Backend-Code. Place all your source code in respective Folders.
- Do not submit binaries, .class files, or supporting libraries (e.g., junit.jar, javaee.jar) (including them would be **3 points** deduction).
- Include the Readme file to document the steps to run the application.
- **All the dependencies should be added into package.json file.**

Submission (Report Submission)

- **On-line submission:** shall include your report (smith_lab1_report.doc). Submissions shall be made via canvas.
- **Project report**
 - Introduction: state your goals, purpose of system,
 - System Design: Describe your chosen system design
 - Results: Screen image captures of each client/server pair during and after running.
 - Performance: What was performance? Analyze results and explain why you are getting those results.
 - The answers to the questions.