CS472, MUM Carpooling final project, August 2016

This is a team-based project. Every team must have 3 students. Team members should be formed and submit their names by the end of this session! If you fail to form a team, you will be randomly assigned with a group.

Application specifications and requirements:

When you first visit your web application, you will be prompted with a screen to login or create a new profile.

Profile should include the following data: (Full Name, Gender, State, City, Street, Zip code, Birth Year, Email, Password).

Password complexity should be at least 6 letters and have at least one capital letter, one small letter, one number (use RegEx).

For a safer community and rides, only members with 18+ years old are allowed to use the application.

When users finish the sign up process they should be logged in automatically.

After login, users will see latest 25 car-pooling ride-offering posts sorted by date (latest post is first).

An automatic infinite scroll pagination should be implemented: so when you scroll down the next 15 posts will fade-in using AJAX.

Every user should be able to add a new text post to the application wall. There should be an option for users to delete their own posts using AJAX.

Posts have two types: Asking for a ride, offering a ride (default display). A tab should be provided to change the view using AJAX.

When a new post is submitted by other users while you are on screen, you should see a notification on top of your page. If you click that notification it will append all the new posts to the DOM.

Users should be able to like a post, read all previous comments and add a new comment using AJAX.

Users should be able to update their profile information.

The application will provide a weather service to help users plan their trips and drive safely. A link should be located in the Navigation bar for the weather service.

The application will display two things (Current weather map, 5 days/3 hour forecast data) based on their city/zip code in their profile, if missing the app should read their current GPS location.

Users should also be able to search and display weather information about their destination cities/zip code.

Technical information:

There is a suggested MySQL database structure file in the project directory **db.sql** , you can import the file into MySQL database on your machine.

You will need to install MySQL Server:

- Standalone: MySQL Community Server http://dev.mysql.com/downloads/
- Download MySQL Workbench to manage your DB

Download **mysql-connector-java** and configure the connector with your Web Project in Eclipse.

To import the database manually from the command line:

1. mysql – u root –p (where root is your username as appropriate)
2. source d:\db.sql (assuming you have downloaded the db.sql file to the root of your d directory
3. You can find a sample of how we can connect to MYSQL using the JDBC java file **DBconnection.java** provided in the project folder (tune it to your taste).

You should format all data coming from DB into a JSON string and return that to your Ajax query.

- Use JSON-Simple https://code.google.com/archive/p/json-simple/downloads
- Example of parsing JSON http://www.mkyong.com/java/json-simple-example-read-and-write-json/

You can retrieve map information from openweathermap.org API, you will have to integrate the data coming from the service API with Google Maps. First you will need to sign up for a free account and get an API token that will allow you to query their services.

A guide on displaying weather data with the Google Maps API and OpenWeatherMap will be provided for you in the project folder (**OpenWeatherMapLayer.pdf**)

Important notes:

- You must use MVC architecture.
- You must use at least one Custom Tag to show your content.
- You must use Ajax to perform posting and commenting... etc
- You must use JSON parser and retrieve all your map data as JSON.

Notes and daily routine:

Every team member will be responsible for one part of the project, there should be a slide in your presentation showing which part of the project each one has worked on and all tasks performed.

A detailed project plan should be submitted by Monday 01:00 pm.

Before you start you better setup and use an online SVN service (Git is advised).

Every team member should be assigned on one part of the application, so they can "divide-and-conquer" that part into small tasks:

- Java MVC Backend, Login, User profile, Homepage and Custom Tags
- CSS frameworks, responsive layout, JavaScript modules and AJAX calls
- Weather Service and Google API

Despite the fact that projects will be presented in teams, every team member will be graded individually based on their contribution and completion of their tasks.

There will be a daily short meeting staring of 10:00 am in our classroom to discuss your progress and answer all your questions, please respect the following schedule and be there on time:

Time	**Monday**	**Tuesday**	**Wednesday**
10:00-10:10	group 1	group 4	group 7
10:10-10:20	group 2	group 5	group 8
10:20-10:30	group 3	group 6	group 1
10:30-10:40	group 4	group 7	group 2
10:40-10:50	group 5	group 8	group 3
10:50-11:00	group 6	group 1	group 4
11:00-11:10	group 7	group 2	group 5
11:10-11:20	group 8	group 3	group 6
11:20-12:30	Misc Questions (if needed)		

Failing to attend your daily checkup meeting on time, or an absent team/member will affect your final project grade.

You may come to your appointed time and leave after you finish. However, you may use the classroom all day long if necessary - It will be kept open for you.

Presentation schedule:

All teams should be ready and present on Thursday morning 10:00 am for the final presentation and celebration.

There will be 15 minutes' presentation time for every team as the following:

- Every team member should talk about their contribution and show their tasks and progress
- Every team member should show and explain a sample of their code
- General Demo
- Basically there will be 5 minutes' space for every member.
- There might be a space for few Q/A or comments at the end of your presentation.

Finishing all the project is definitely a plus and shows an excellent performance in the course and will guarantee a top grade. However, you don't have to finish all your tasks to get a full grade, grades will be determined by the quality of your tasks rather than quantity. Do your best and enjoy the experience of team programming.

Extra Points – 3 Points will be added to every member of the team

Update the DB to include two locations in every post: Source & Destination coordination (not names).

Add Search service to allow people:

1. Search for trips based on location using Geolocation Spatial Query to show trips near the same area. Example (Searching trips to Mt. Pleasant will show results for Burlington trips)
2. See all trips on the map (Using Google Maps API).

Note: Partial implementation to point 1 or 2 will be considered!

Project submission:

Every student must submit the following individually to Sakai by Friday 10 pm:

- A link to your SVN/Git repository (or attach the project source code).
- A report showing your project plan and your tasks and your progress.
- One-page essay explaining what you learned from the project: things that worked well, problems you faced and how you solved them.
- Advice for improving the course in the upcoming offerings.

Good luck and happy coding!