Faculty of Science, Engineering and Technology

COS30043
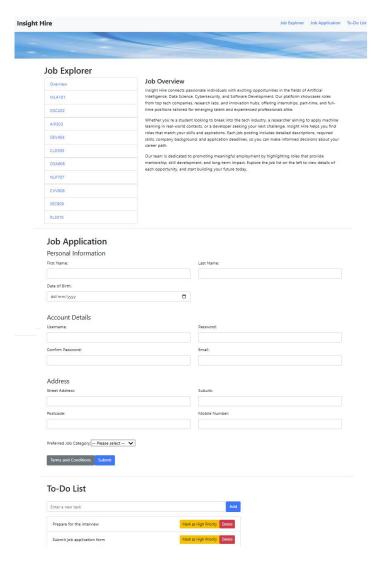# Interface Design and Development

Assignment 2. Insight Hire

1. Overview

The purpose of this assignment is to develop a web application named *Insight Hire* using Vue.js 3 and Bootstrap. This application allows users to browse for jobs, apply for jobs, and manage their tasks. The web app must be deployed to the university web server (Mercury).

You are required to implement a web application that consists of three sections: Job Explorer, Application Form, and To-Do List. A sample screenshot is provided below to illustrate these sections and give you an overall understanding of the website. Please use your own design while ensuring that all requirements are met.

2. Components

In this assignment, all resources (Bootstrap, Vue and Vue Router) must be stored under your own web server directory. This ensures stability and prevents unexpected behavior due to external framework updates. Therefore, make sure you are using Bootstrap and Vue from your web server, not through a CDN.

In this application, you must use Vue components for the root component and other required components, placing each in a separate file (.js or .vue). You should have the following components: App (root component), JobList, JobDetail, JobOverview, ApplicationForm, and ToDoList. Please organise your folders appropriately according to best practices.
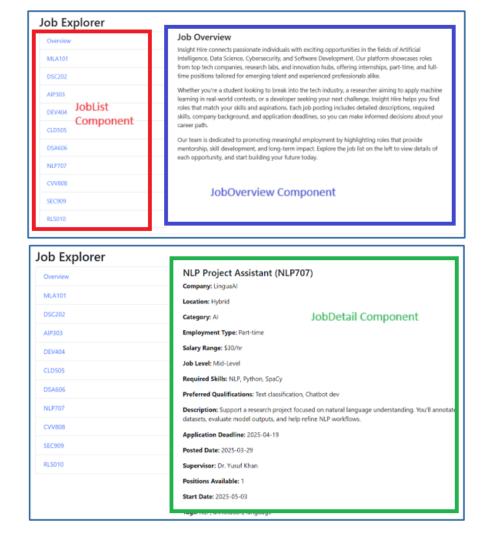
3. Job Explorer

This section needs to be implemented using Vue Router. It should display a list of job IDs on the left, shown as links using <router-link>. An 'Overview' link should appear at the top of the list. When the 'Overview' or a job ID link is clicked, the corresponding overview description or job details should be displayed on the right, including all relevant job fields in the detail view.

You need multiple components for this section, and place each in a separate file.

- App – the root component
- JobList – the job list component
- JobDetail – the job detail component
- JobOverview – the job overview component
- router – the router configuration component

The job list is in the jobs.txt file, and you can hard-code it as a JavaScript array in your code. You can also make minor improvements to the jobs.txt file while keeping the main structure.

4. Application Form

Implement this section using a component named ApplicationForm.

Create an application form that validates user inputs and displays appropriate error messages. The form must not be submitted if there are any validation errors. Group related input fields using <fieldset> where appropriate.

Required Fields:

1. First Name – Letters only, required
2. Last Name – Letters only, required
3. Username – Minimum 3 characters, required
4. Password – Minimum 8 characters, must include at least one special character ($, %, ^, &, *), required
5. Confirm Password – Must match the password
6. Email – Must be a valid email format, required
7. Street Address – Optional, maximum 40 characters
8. Suburb – Optional, maximum 20 characters

9. Postcode – Exactly 4 digits, required (some postcodes may start with 0)
10. Mobile Number – Exactly 10 digits, must start with 04
11. Date of Birth – Required, must be a valid date and the applicant must be at least 16 years old
12. Preferred Job Category – Required, must be selected from a dropdown (e.g., AI, Data Science, Web Development)
13. Submit Button
14. Terms and Conditions Button – Toggles the visibility of a brief terms and conditions text (any placeholder text is acceptable)

Use the following attributes in the <form> tag to submit your form data for testing:

<form method="post" action="http://mercury.swin.edu.au/it000000/formtest.php">

Upon submission, all form data should be displayed with their corresponding values on the server response page.

Job Application

Personal Information

First Name:

Last Name:

Date of Birth:

dd/mm/yyyy

Account Details

Username:

Password:

Confirm Password:

Email:

Address

Street Address:

Suburb:

Postcode:

Mobile Number:

Preferred Job Category: -- Please select --

Terms and Conditions   Submit

5. To-Do List

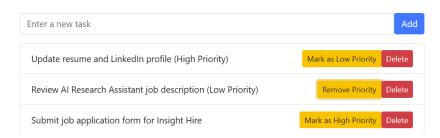Implement this section using a component named ToDoList.

This section allows users to manage a list of to-do tasks. They can add tasks, delete tasks, and update the priority status of each task (e.g., mark as High Priority or Low Priority).

There should be an Add button that allows users to enter a task. Added tasks should appear in a list (e.g., in a list or table), with new tasks inserted at the beginning of the array.

Each task should have a Delete button that allows users to remove the task, and a Toggle Priority button that cycles between High Priority and Low Priority. When a task is marked as

Faculty of Science, Engineering and Technology

High Priority, append the text '(High Priority)' after the task. When marked as Low Priority, append the text '(Low Priority)'. The button caption should also toggle between 'Mark as High Priority' and 'Mark as Low Priority' accordingly.

To-Do List

Enter a new task	Add
Update resume and LinkedIn profile (High Priority)	Mark as Low Priority Delete
Review AI Research Assistant job description (Low Priority)	Remove Priority Delete
Submit job application form for Insight Hire	Mark as High Priority Delete

6. Submission Requirements

- Submit a ZIP file containing all your source code to Canvas.

- Upload your web app to Mercury, using the specified folder structure.

- Include your Mercury server URL in the submission comment. Example: http://mercury.swin.edu.au/cos30043/s1234567/A2/

7. Marks Breakdown (20 Marks Total)

Marks are based on your web application and your understanding of the required knowledge and techniques, according to the following criteria:

Job Explorer	6 marks
Application Form	6 marks
To-Do List	6 marks
Folder Structure & Deployment	1 mark
Overall Web App Appearance	1 mark