**Name: Taranjeet Singh**

**Humber No: N01539902**

**Assignment 2**

**Step 1:**

Create new Node/Express app as follow:

* Create new folder named “Asn2-yourname”.
* Change directory to “Asn2-yourname”:
* Open terminal and run npm init

A screenshot of a computer program

Description automatically generated

* In the terminal run: npm install express --save

A screen shot of a computer

Description automatically generated

* In the terminal run: npm install express-handlebars --save

A black background with text

Description automatically generated

* Make sure to have the following project structure:

A screenshot of a computer

Description automatically generated

**Step2:** Using the attach file (codenippet.txt), copy/paste the related code to “style.css”, “main.hbs”, “index.hbs”, “error.hbs” and “app.js”.

**Style.css**

A screenshot of a computer

Description automatically generated

**Main.hbs**

A screen shot of a computer program

Description automatically generated

**Index.hbs**

A screenshot of a computer

Description automatically generated

**Error.hbs**

A screenshot of a computer

Description automatically generated

**App.js**

**A screen shot of a computer code

Description automatically generated**

**Step 3:** Run the application and test it using the following routes:

* http://localhost:3000/

A screenshot of a computer

Description automatically generated

* http://localhost:3000/users

A screenshot of a computer

Description automatically generated

* http://localhost:3000/notExist

A screenshot of a computer

Description automatically generated

**Step 4:** Base on your observation, answer the following questions:

1. What is the role of “main.hbs”? How does it being used/called in the app?

main.hbs serves as the main layout template for the application. It provides the base structure for the application's HTML pages. Specific parts of this template, like {{title}} and {{{body}}}, are placeholders that can be dynamically replaced with content:

* {{title}}: A placeholder for the page title.
* {{{body}}}: A placeholder for the main content of the page. The triple curly braces {{{ }}} indicate that the content should be rendered without escaping, allowing HTML to be inserted directly.

1. What is the role of “index.hbs” and “error.hbs”? How do these files being used/called inthe app?

* **index.hbs:**

Role: This template seems to represent the main or home page of the application.

Usage: When the template is rendered, the placeholder {{title}} will be replaced by the actual title provided. For example, in the app.js code you provided earlier, the line res.render('index', { title: 'Express' }); will render this view, replacing {{title}} with "Express".

* error.hbs:

Role: This template is used to display error messages.

Usage: The placeholder {{message}} will be replaced with the actual error message provided when rendering the template. In the app.js code, the line res.render('error', { title: 'Error', message:'Wrong Route' }); will render this view when a user tries to access a route that doesn't exist, replacing {{message}} with "Wrong Route".

Both of these templates (index.hbs and error.hbs) will be wrapped within the main.hbs layout when rendered, meaning the content of these views will replace the {{{body}}} placeholder in the main.hbs template.

1. In “app.js”, what is the role of the following line? What happen if you commend it out?

app.use(express.static(path.join(\_\_dirname, 'public')));

This line serves static files (like CSS or JS) from the **public** directory. If you comment it out, these static files won't be accessible, potentially breaking styles or scripts on web pages

1. Try to add proper comments to each line/block in app.js

A computer screen shot of a program code

Description automatically generated

**Step 5**: Modify the layout of the page and add “header”, “content” and “footer” as follow.

* Add your name/std\_id to the header

A screen shot of a computer

Description automatically generated

* Add your email address to the footer

A computer screen shot of a program

Description automatically generated

* The other dynamic contents/messages should be displayed in the “content” area.

A screenshot of a computer

Description automatically generated

* Use your creativity to add proper css style/format. Add

*(hint: you need to redesign main.hbs and style.css*)

**Step 6**: Add the Assignment1 code into this template. So the assignment1 form and output should be displayed in the content area of the layout.

* For each of the route in assignment 1 (like /data, /data/invoiceNo/{index}, /search/invoiceNo/, /search/ Manufacturer/), you need to create a related “page” in “/views” folder. So, when the “app.get” of the related route is called, you gonna use the proper “res.render(‘page’, {data})”. (*hint: use similar ideas like index.hbs*) Moreover, you also should redesign the way that you generate output in Assingment2 by displaying the output using Handlebars *expression/helpers*.

A screenshot of a computer

Description automatically generated

**All Data:**

A screen shot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

Search Invoice ID:

A screenshot of a computer

Description automatically generated

A screenshot of a car

Description automatically generated

Search Manufracturer:

A screenshot of a computer

Description automatically generated

**Step 7**: Design a new route “/allData” to display all Sales info in html table output using handlebars helpers/expression.

A screenshot of a computer

Description automatically generated

**Step 8**: Modify Step 7 (change the endpoint/view name) by removing those records which their “class” is blank (*hint: use “#if” helper* ☺)

A computer code on a black background

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

**Step 9**: Design a custom helper for changing “blank” to “unknown” for records which their “class” is “blank”. Apply this on Step7&8 in order to display all data but highlight those rows that their “class” is “blank”!

A computer screen shot of code

Description automatically generated

A screen shot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

**Step 10: Any idea how to do any of the followings:**

* You want to change table row color for those records which “class” is “blank”. Any way that you can implement it using Template Engine? Explain your answer

With Handlebars, we can use the **{{#if}}** helper to check the value of the "class" property and conditionally apply a CSS class to the table row.

**Implementation:**

1. In your Handlebars template:

{{#each data}} <tr {{#unless class}}class="highlight-row"{{/unless}}> <td>{{InvoiceNo}}</td> <td>{{Manufacturer}}</td> <td>{{class}}</td> </tr> {{/each}}

1. Add CSS:

.highlight-row { background-color: #f2f2f2; /\* Gray color for blank class \*/ }

* In this app, do you find any use cause for utilizing “Partial” Templates? Explain your answer

**Explanation:**

Partials in Handlebars allow for template reusability. Given the repetitive nature of certain parts of a web page, like headers or footers, partials can be useful. In your app, the navigation header is a good candidate for a partial since it's likely to appear on multiple pages.

**Implementation:**

1. Create a **header.hbs** partial:

<header> <div class="header-title">Taranjeet Singh / N01539902</div> <nav class="navbar"> <a href="/">Home</a> <!-- ... other links ... --> </nav> </header>

1. Register the partial in your Express setup:

hbs.registerPartials(path.join(\_\_dirname, 'views/partials'));

1. Use the partial in your main templates:

{{> header}}

<!-- Rest of the template -->

* Bonus: implement your idea of A and B!