**N01639939**

**STEVEN MARTY M. CES**

**ASSIGNMENT#2**

Complete the following steps and take screenshot of the outcome of each step:

**Step1:** **Create new Node/Express app as follow:**

• Create new folder named “Asn2-yourname”.

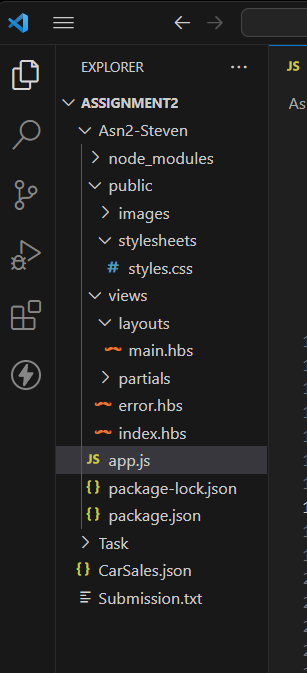
• Change directory to “Asn2-yourname”:

• Open terminal and run npm init

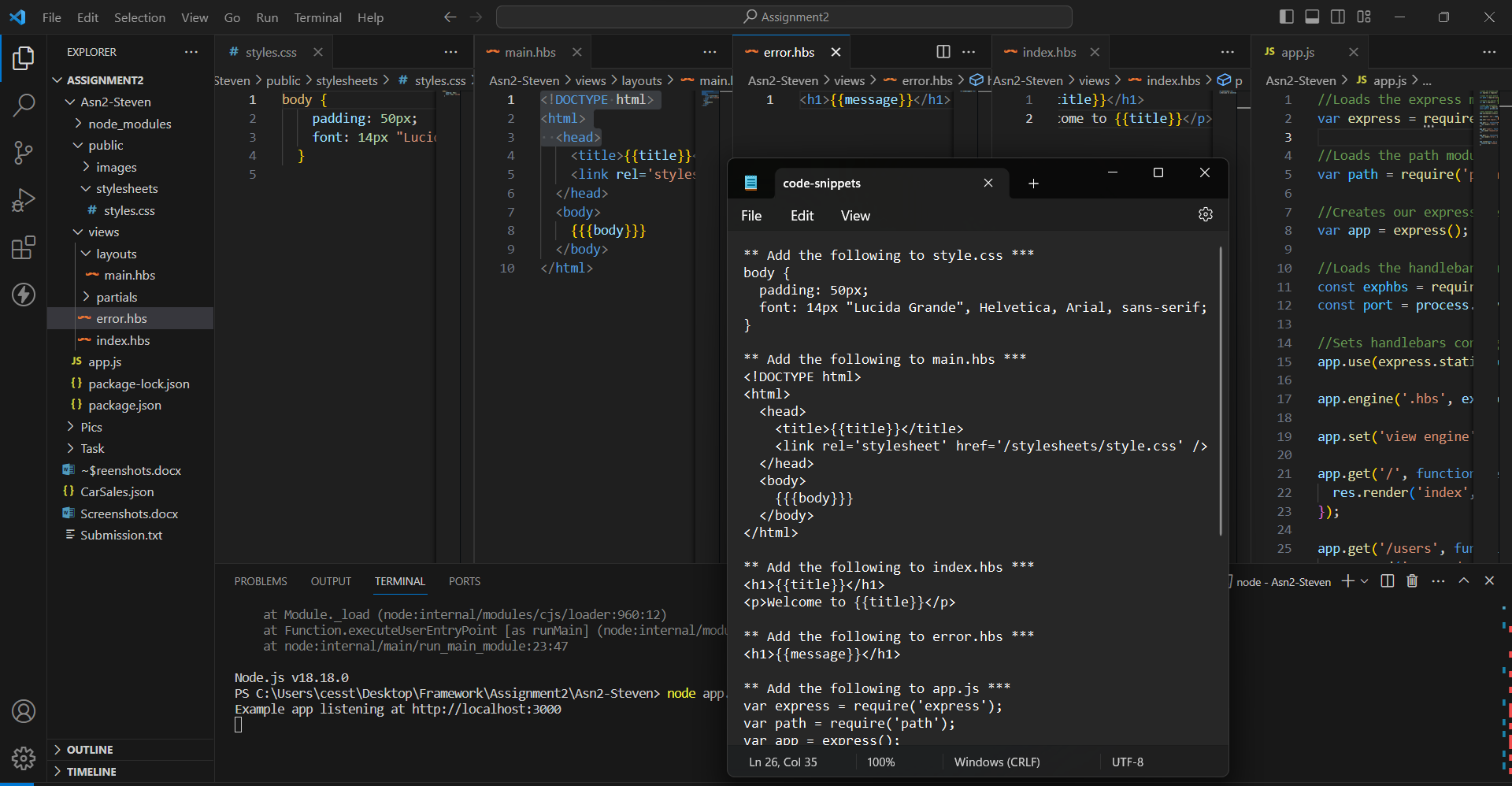
• In the terminal run: npm install express --save

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

• Make sure to have the following project structure:



**Step 2:** Using the attach file (codenippet.txt), copy/paste the related code to “style.css”,

“main.hbs”, “index.hbs”, “error.hbs” and “app.js”.

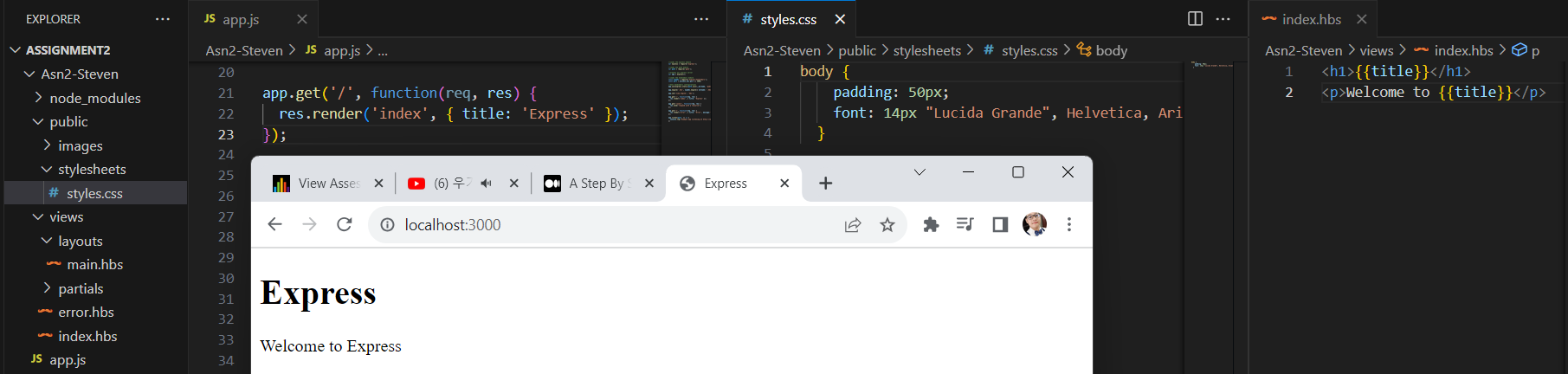
**Step 3:** Run the application and test it using the following routes:

• <http://localhost:3000/>

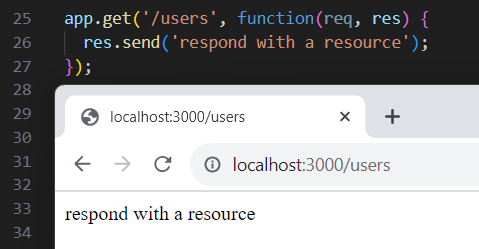
• <http://localhost:3000/users>

• <http://localhost:3000/notExist>

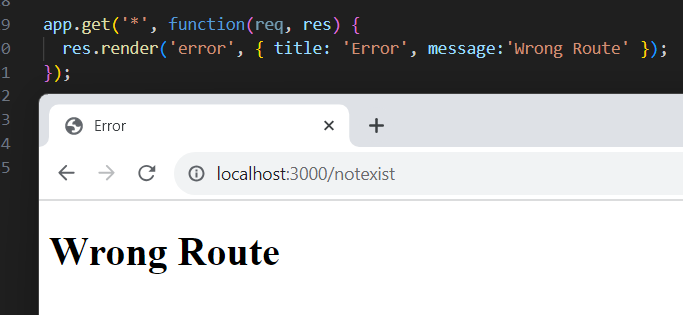
**Step 3.1:** Running the application at <http://localhost:3000/>.



**Step 3.2:** Running the application at <http://localhost:3000/users>



**Step 3.3:** Running the application at <http://localhost:3000/notExist>



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

**a)** **What is the role of “main.hbs”?**

* it handles the HTML file along with its link of the css showing the output of ‘/’ or index titled ‘Express’ using the placeholders for dynamic content. Basically, handles the main contents and be used in every page related to it.

**How does it being used/called in the app?**

* through res.render, the {{title}} and {{{body}}} for the placeholders and using the Handlebars templating engine as well as the code : app.set('view engine', 'hbs') that sets the handlebars configurations be enabled.

**b)** **What is the role of “index.hbs” and “error.hbs”?**

* for index.hbs, it is rendered and called using the res.render() function the same as the main.hbs, but can only be used/called to its respective route. While the error.hbs, is the same format, being rendered and called as well using res.render() function, the difference was that this handlebar will be used/called whenever there’s an error in the route handler.

**How do these files being used/called in the app?**

---- same as others, by a templating engine, such as Handlebars.

**c) In “app.js”, what is the role of the following line?**

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

* this line serves a static directory of files named ‘public’ to the client thru path.join(\_\_dirname,’public’). The app.use confirms this line as a middleware Express. While the express.static allows the client to include assets like images, css and javascripts in the web application.

**What happen if you commend it out?**

* then I won’t be able to use any static files within the public directory like my css, images or javascript files.

**d) Try to add proper comments to each line/block in app.js *(COMMENTS ARE ADDED AT APP.JS)***

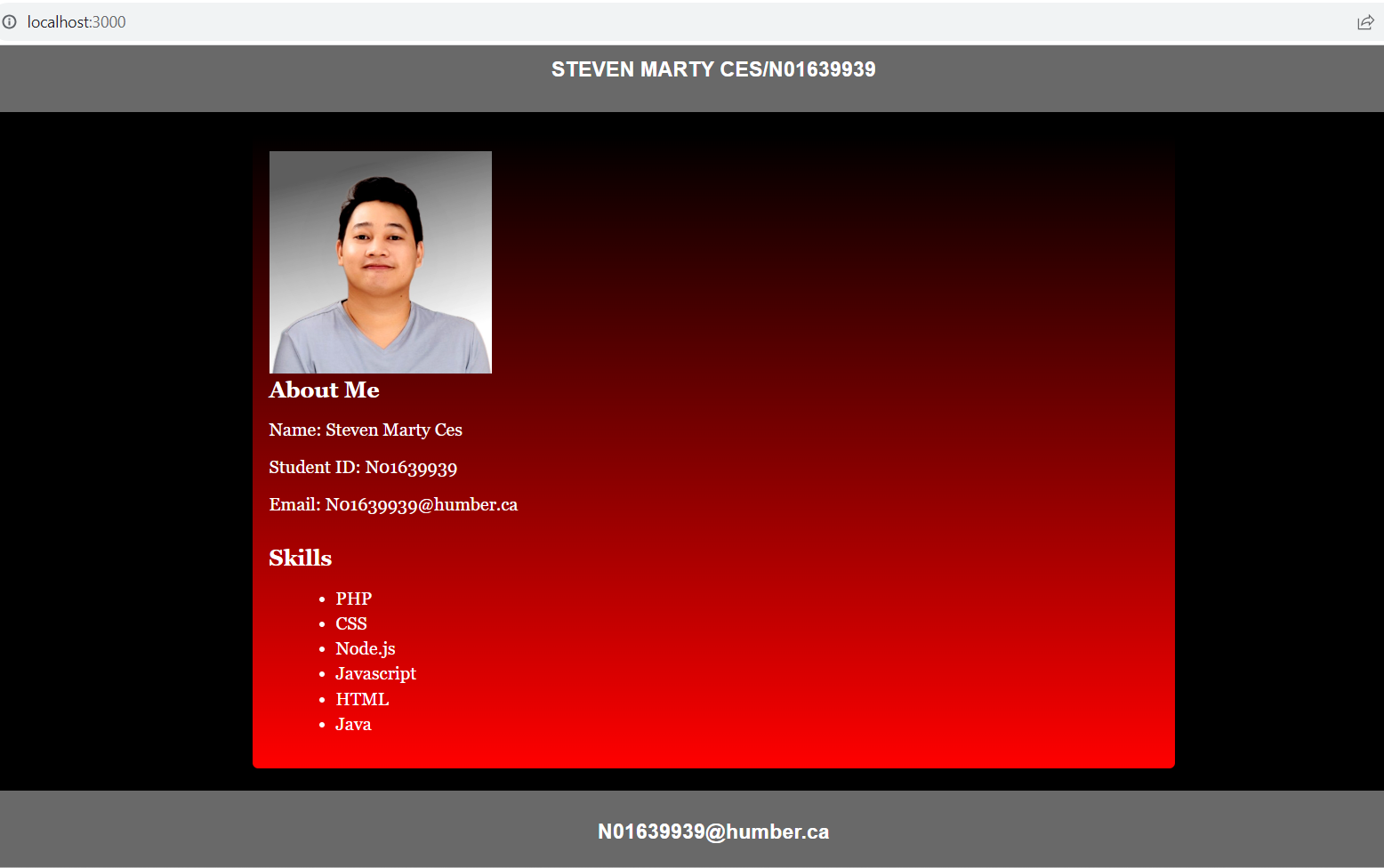
**Step 5: Modify the layout of the page and add “header”, “content” and “footer” as follow.**

• Add your name/std\_id to the header

• Add your email address to the footer

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

• 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. Show 1 data each search bar.

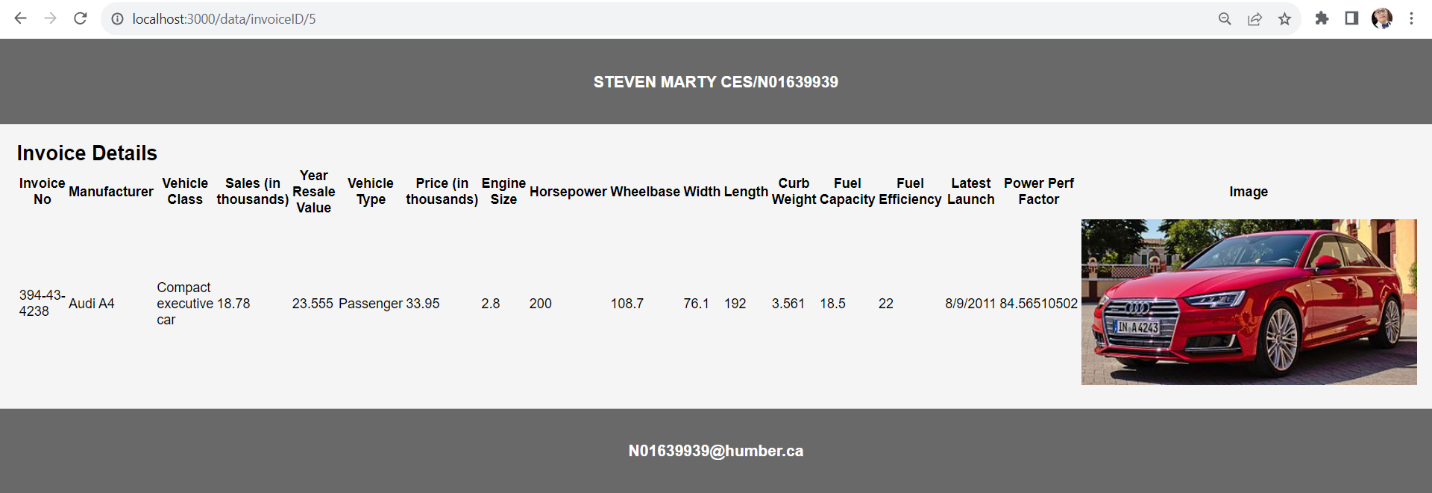
So, when the “app.get” of the related route is called, you gonna use the

proper “res.render(‘page’, {data})”. (hint: use similar ideas like index.hbs) Moreover,

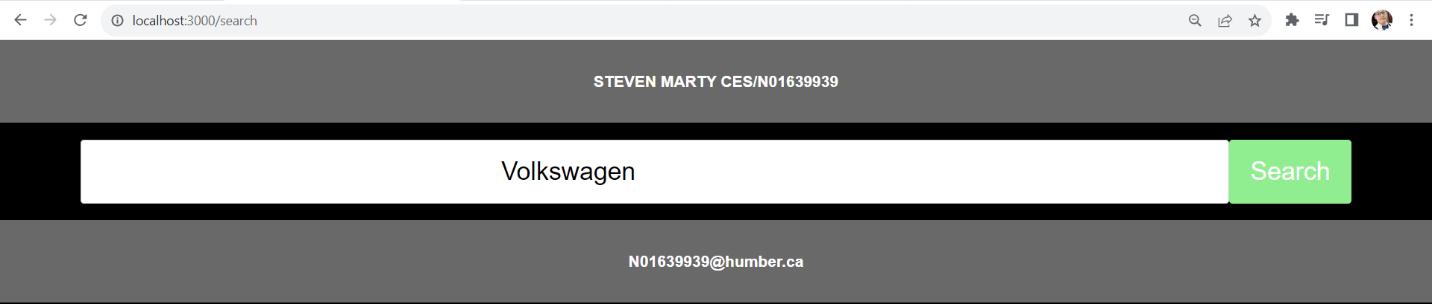
you also should redesign the way that you generate output in Assingment2 by displaying

the output using Handlebars expression/helpers.

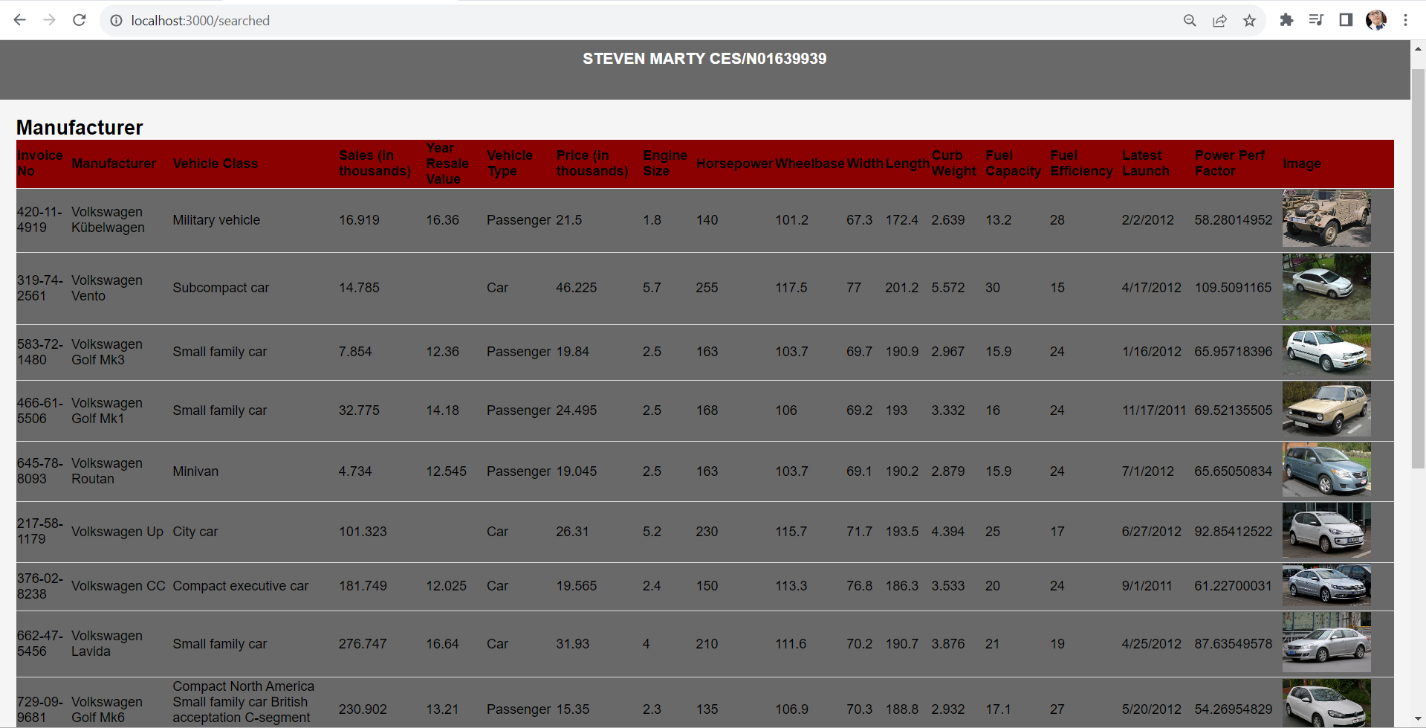
**\*\*\*/data/invoiceNo/{index}, using this route to show each dataset from their given index.**

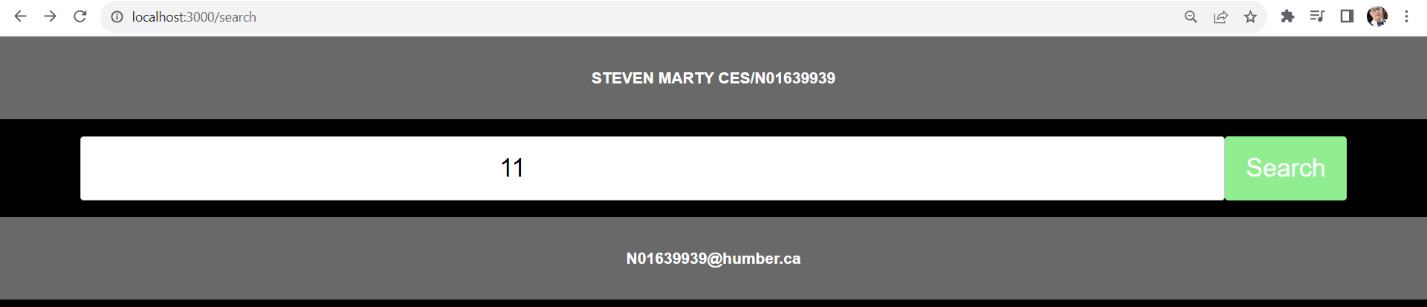
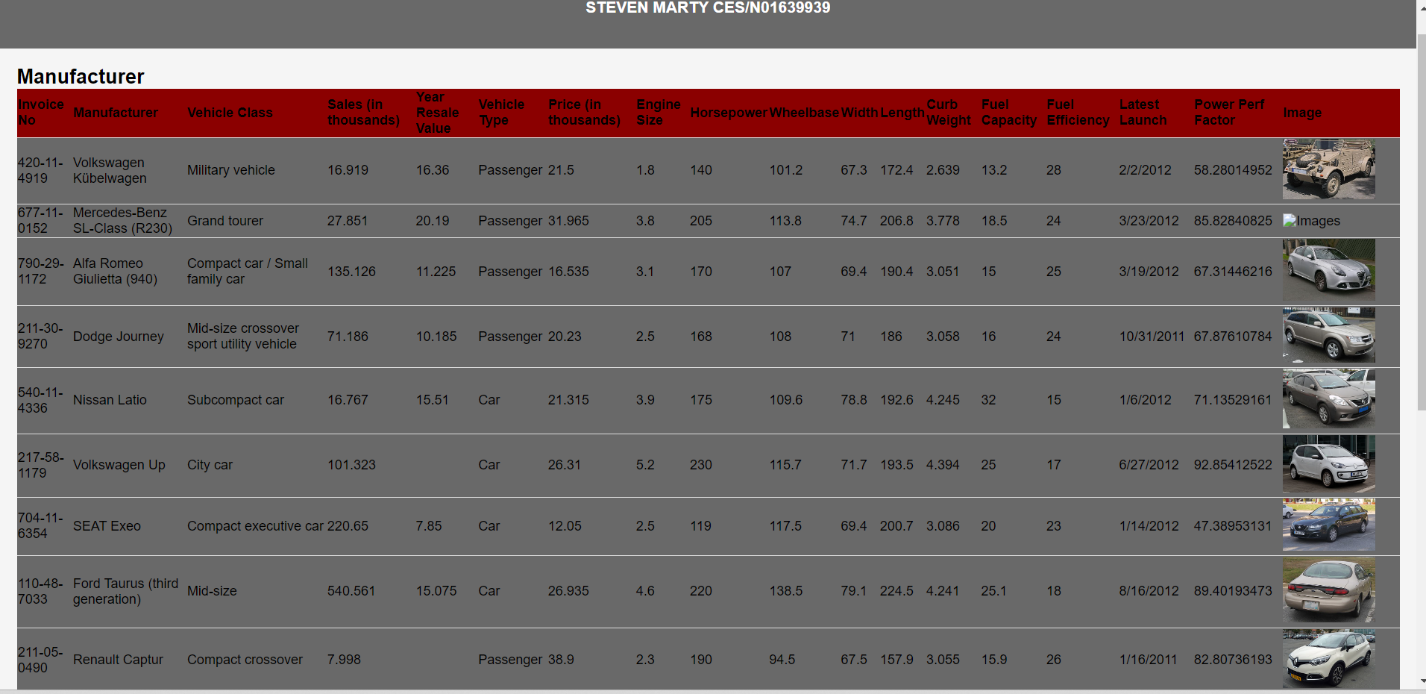


**\*\*\*/search, /search/invoiceNo/, /search/ Manufacturer/),**

 **-Using localhost:3000/search to access the page for search bar for either invoiceNo or its manufacturer. I created a search bar page to make it more user friendly.**

**-Searching for ‘Volkswagen’ Manufacturer:**

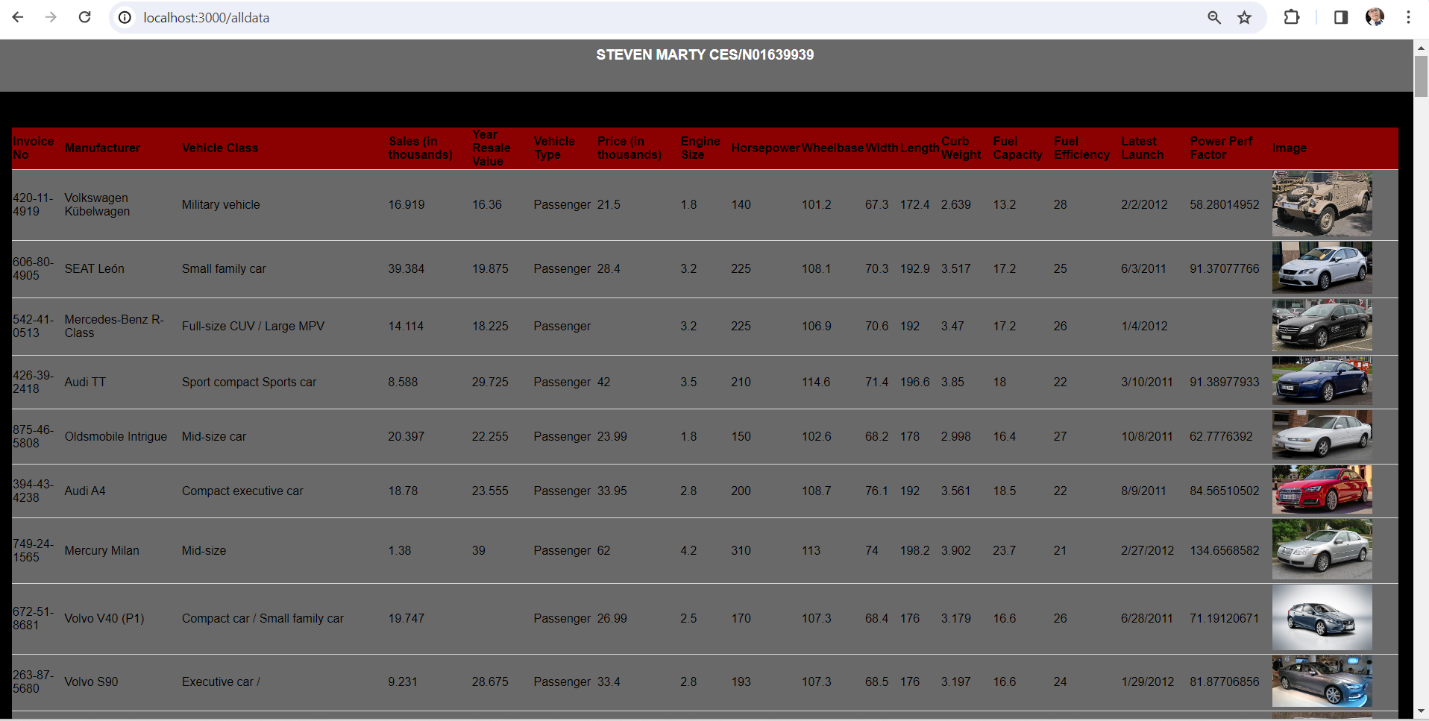
****



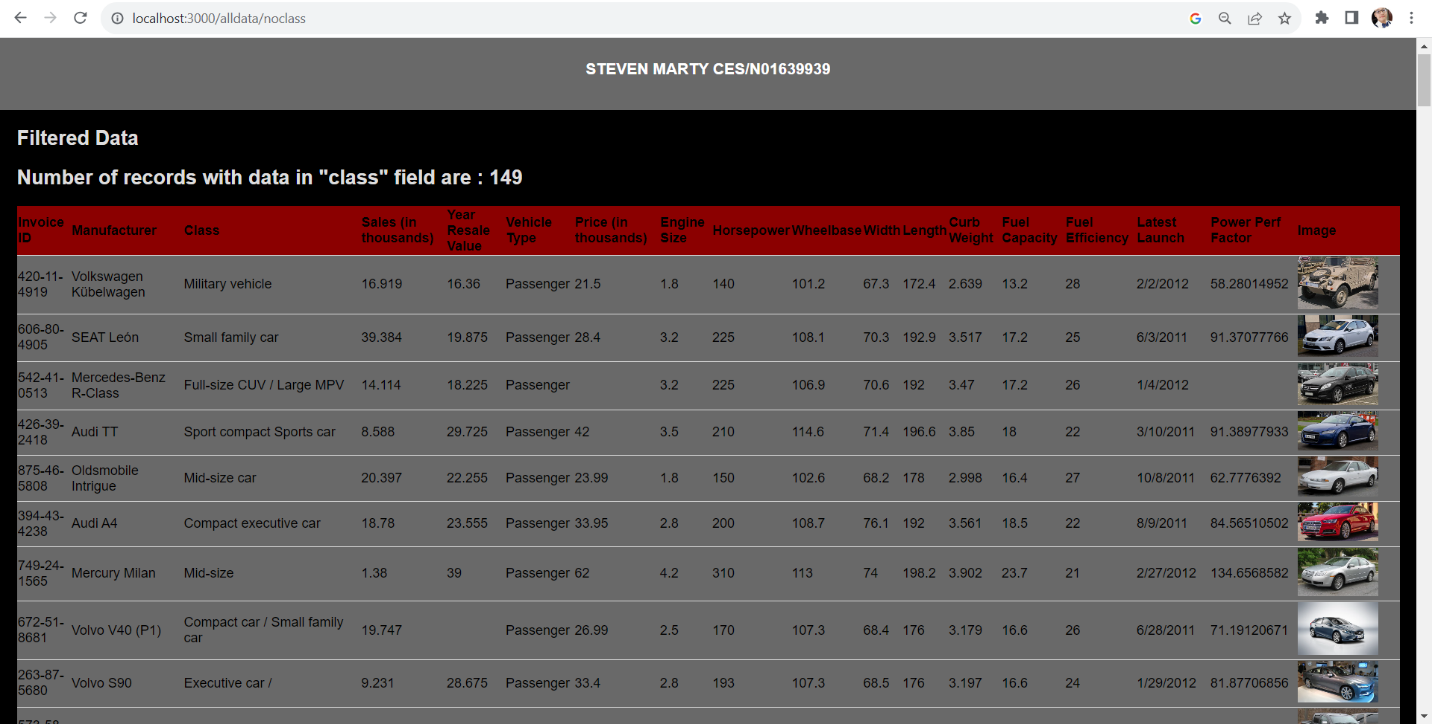
**-Searching for dataset with “11” in their InvoiceNo:**

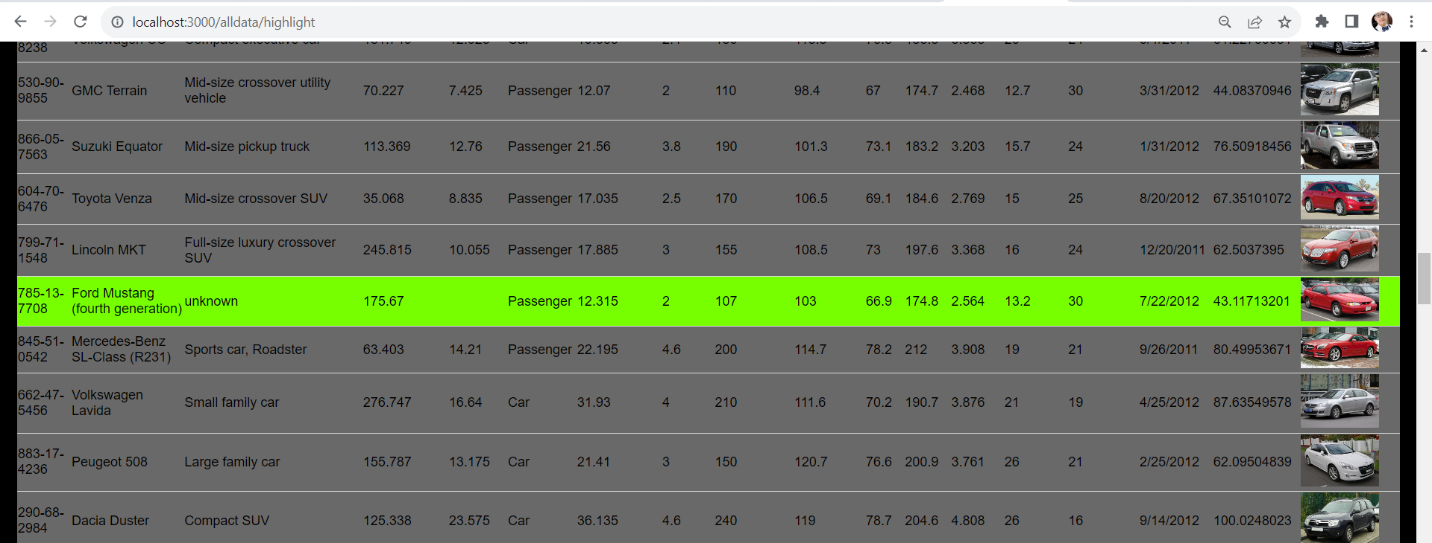
**Step 7: Design a new route “/allData” to display all Sales info in html table output using**

**handlebars helpers/expression.**



**Step 8: Modify Step 7 (change the endpoint/view name) by removing those records which their**

**“class” is blank (hint: use “#if” helper ☺). *Using localhost:3000/alldata/noclass to display the output filtered with empty data in Class field. The total number of data is 157, now less with 8 records (Empty Class Field) is equal to 149.***

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

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

A. 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.

--well first one I thought off was CSS, but then I have no idea how to implement them with my code. But then I realized, I must use handlebars package and register the "eq" helper:

<tr class="{{#if (eq class 'unknown')}}highlighted{{/if}}">

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

--No, not really, I’ve pretty much use what works in views. There was a situation where I had to revise my codes just to exclude the main.hbs from the layouts because it still displays its content in all my handlebar views which is kind of frustrating going over an over of the revision, but still, I still manage to work it all out.

• Bonus: implement your idea of A and B!

**Step 11: Deploy the app to Cycle**.