Contents

[Aims and Objectives 2](#_Toc90471028)

[Design 3](#_Toc90471029)

[ERD 3](#_Toc90471030)

[Data dictionary 3](#_Toc90471031)

[Testing 4](#_Toc90471032)

[Developmental testing 4](#_Toc90471033)

[Testing 4](#_Toc90471034)

[Evaluation 5](#_Toc90471035)

## Aims and Objectives

Your aim should be an opening paragraph, explain what you are doing and why

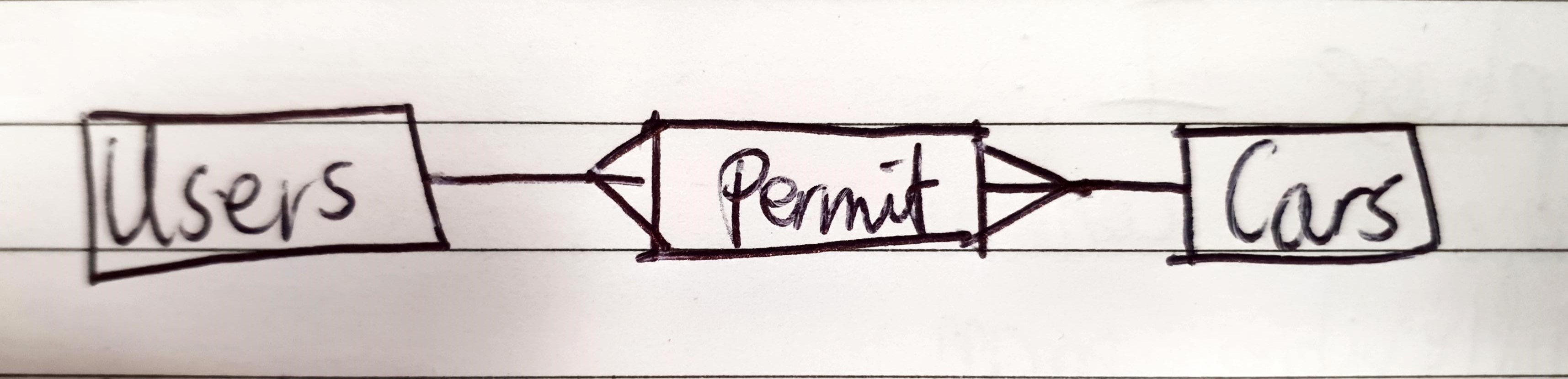
Objectives should be a numbered list identifying exactly what you will be achieving, these should all have measurable success criteria. Can you check it has been achieved?

I want to create a simple yet efficient flask app that can allow the user to easily parse the collyers car park database in search of a specific car or user to see if they have a permit. I would also like to have some toggle buttons which will make it easier to do so and will automatically act as data validation check marks. I want to make the gui collyers themed, using their signature colour pallet – gold and blue.

* Have a results table – make it take up the majority of the screen
* Use mainly gold and blue for any gui elements
* Have a selection of filters such as:
  1. Staff
  2. Student
  3. Visitor
  4. Has permit
  5. Permit valid start date
  6. Permit valid end date
* Text input fields such as:
  1. Name
  2. Car Reg plate
  3. Car make
  4. Car model
* An option to enter in new data
* Display the cars photo and the owner's photo

## Design

You have been given an outline of the user interface so you don’t need to repeat that here, you need to show the design for your database and processing



Function query\_database(data)

User\_query = ‘SELECT SYNumber, FirstName, LastName, Age, UserType, PhotoPath FROM Users WHERE’, data[user\_filters]

Car\_query = ‘SELECT NumberPlate, Colour, PhotoPath FROM Car WHERE’, data[car\_filters]

permit\_query = ‘SELECT SYNumber, NumberPlate, HasPermit, ValidFrom, ValidUntil FROM Permit WHERE’, data[permit\_filters]

sql = f"SELECT u.FirstName, u.LastName, u.Age, c.NumberPlate, c.Colour, p.HasPermit, u.UserType, p.ValidFrom, p.ValidUntil, u.PhotoPath, c.PhotoPath FROM ({user\_query}) u JOIN ({permit\_query}) p ON u.SYNumber = p.SYNumber JOIN ({car\_query}) c ON p.NumberPlate = c.NumberPlate;"

rows = Database.execute(sql)

data = []

for i in range(len(rows)) then

data.append({

'first\_name': rows[i][0],

'last\_name': rows[i][1],

'age': rows[i][2],

'number\_plate': rows[i][3],

'colour': rows[i][4],

'permit': rows[i][5],

'usertype': rows[i][6],

'valid\_from': rows[i][7],

'valid\_until': rows[i][8],

'user\_photo': rows[i][9],

'car\_photo': rows[i][10],

'Id': i

})

Endfor

Return data

### ERD

Draw your ERD here and write the standard notation under to fully identify the keys you will be using.

### Data dictionary

Each table you have identified above needs a separate data dictionary.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data type | Required Yes/No | Validation | Description | Key |
| SYNumber | int | Yes | Length, type, uniqueness | The persons id | yes |
| First name | str | Yes | Length, type | The persons first name | no |
| Last name | str | No | Length, type, existence | The persons last name | no |
| Age | int | Yes | Range, type | The persons age | no |
| UserType | str | Yes | Format | If the user is a student, staff or visitor | no |
|  |  |  |  |  |  |
| NumberPlate | str | Yes | Format, type | The cars reg plate | Yes |
| Make | str | Yes | Length, type | The cars maker | no |
| Model | str | Yes | Length, type | The cars model | no |
| Colour | str | Yes | Length, type | The cars colour | no |
|  |  |  |  |  |  |
| SYNumber | int | yes | Length, type, uniqueness | The persons id | yes |
| NumberPlate | str | yes | Format, type, | The cars reg plate | yes |
| haspermit | boolean | yes | type | If the user + car have a permit or not | no |
| validfrom | date | no | Range, existence | When the permit if valid from | no |
| validuntil | date | no | Range, existence | When the permit is valid until | no |

## Design review

You should add some pseudocode as there wasn't much to judge and it doesn't really describe how the system should function internally

You should also add some more details about the validations used such as the length check of a user's first name and the format check of a car numberplate.

(corrected version)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| NumberPlate | str | yes | Format, type, length, uniqueness | The cars reg plate. The format check will check for 8 characters of length and if so remove the 5th character (which would be a space) or if its 7 characters long then leave it | yes |

Function query\_database(filter\_data) then

if filter\_data['name'] not "" then

words = filter\_data['name'].split(' ')

if len(words) >= 2 then

firstname = words[0]

lastname = ""

for i in range(length(words)-1) then

lastname += words[i+1] + ' '

endfor

lastname = lastname[ then-1]

user\_query = f"SELECT SYNumber, FirstName, LastName, Age, UserType, PhotoPath FROM Users WHERE FirstName = '{firstname}' AND LastName = '{lastname}' AND ("

else then

user\_query = f"SELECT SYNumber, FirstName, LastName, Age, UserType, PhotoPath FROM Users WHERE FirstName = '{filter\_data['name']}' AND ("

if filter\_data['staff'] == 'on' then

user\_query += " UserType = 'staff' OR"

if filter\_data['student'] == 'on' then

user\_query += " UserType = 'student' OR"

if filter\_data['visitor'] == 'on' then

user\_query += " UserType = 'visitor' OR"

if user\_query[-1:] == '(' then

user\_query = user\_query[:-6]

else then

user\_query = user\_query[:-3] + ')'

else then

user\_query = ""

if filter\_data['staff'] == 'on' then

user\_query += " UserType = 'staff' OR"

if filter\_data['student'] == 'on' then

user\_query += " UserType = 'student' OR"

if filter\_data['visitor'] == 'on' then

user\_query += " UserType = 'visitor' OR"

if len(user\_query) > 1 then

user\_query = 'SELECT SYNumber, FirstName, LastName, Age, UserType, PhotoPath FROM Users WHERE' + user\_query

user\_query = user\_query[:-3]

else then

user\_query = 'SELECT SYNumber, FirstName, LastName, Age, UserType, PhotoPath FROM Users'

car\_query = “”

if filter\_data['reg'] not “” then

car\_query += f" NumberPlate = '{filter\_data['reg']}' AND"

if filter\_data['make'] not “” then

car\_query += f" Make = '{filter\_data['make']}' AND"

if filter\_data['model'] not “” then

car\_query += f" Model = '{filter\_data['model']}' AND"

if len(car\_query) > 1 then

car\_query = 'SELECT NumberPlate, Colour, PhotoPath FROM Car WHERE' + car\_query

car\_query = car\_query[:-4]

else then

car\_query = 'SELECT NumberPlate, Colour, PhotoPath FROM Car'

permit\_query = “”

if filter\_data['from'] not 'Valid From' then

permit\_query += f" ValidFrom >= '{ilter\_data['from']}' AND"

if filter\_data['until'] not 'Valid Until' then

permit\_query += f" ValidUntil <= '{filter\_data['until']}' AND"

if filter\_data['permit'] == 'active' then

permit\_query += " HasPermit = '1' AND"

elif filter\_data['permit'] == 'inactive' then

permit\_query += " HasPermit = '0' AND"

if len(permit\_query) > 1 then

permit\_query = 'SELECT SYNumber, NumberPlate, HasPermit, ValidFrom, ValidUntil FROM Permit WHERE' + permit\_query

permit\_query = permit\_query[:-3]

else then

permit\_query = 'SELECT SYNumber, NumberPlate, HasPermit, ValidFrom, ValidUntil FROM Permit'

sql = f"SELECT u.FirstName, u.LastName, u.Age, c.NumberPlate, c.Colour, p.HasPermit, u.UserType, p.ValidFrom, p.ValidUntil, u.PhotoPath, c.PhotoPath FROM ({user\_query}) u JOIN ({permit\_query}) p ON u.SYNumber = p.SYNumber JOIN ({car\_query}) c ON p.NumberPlate = c.NumberPlate;"

rows = Database.execute(sql)

data = []

for i in range(len(rows)) then

data.append({

'first\_name': rows[i][0],

'last\_name': rows[i][1],

'age': rows[i][2],

'number\_plate': rows[i][3],

'colour': rows[i][4],

'permit': rows[i][5],

'usertype': rows[i][6],

'valid\_from': rows[i][7],

'valid\_until': rows[i][8],

'user\_photo': rows[i][9],

'car\_photo': rows[i][10],

'Id': i

})

Endfor

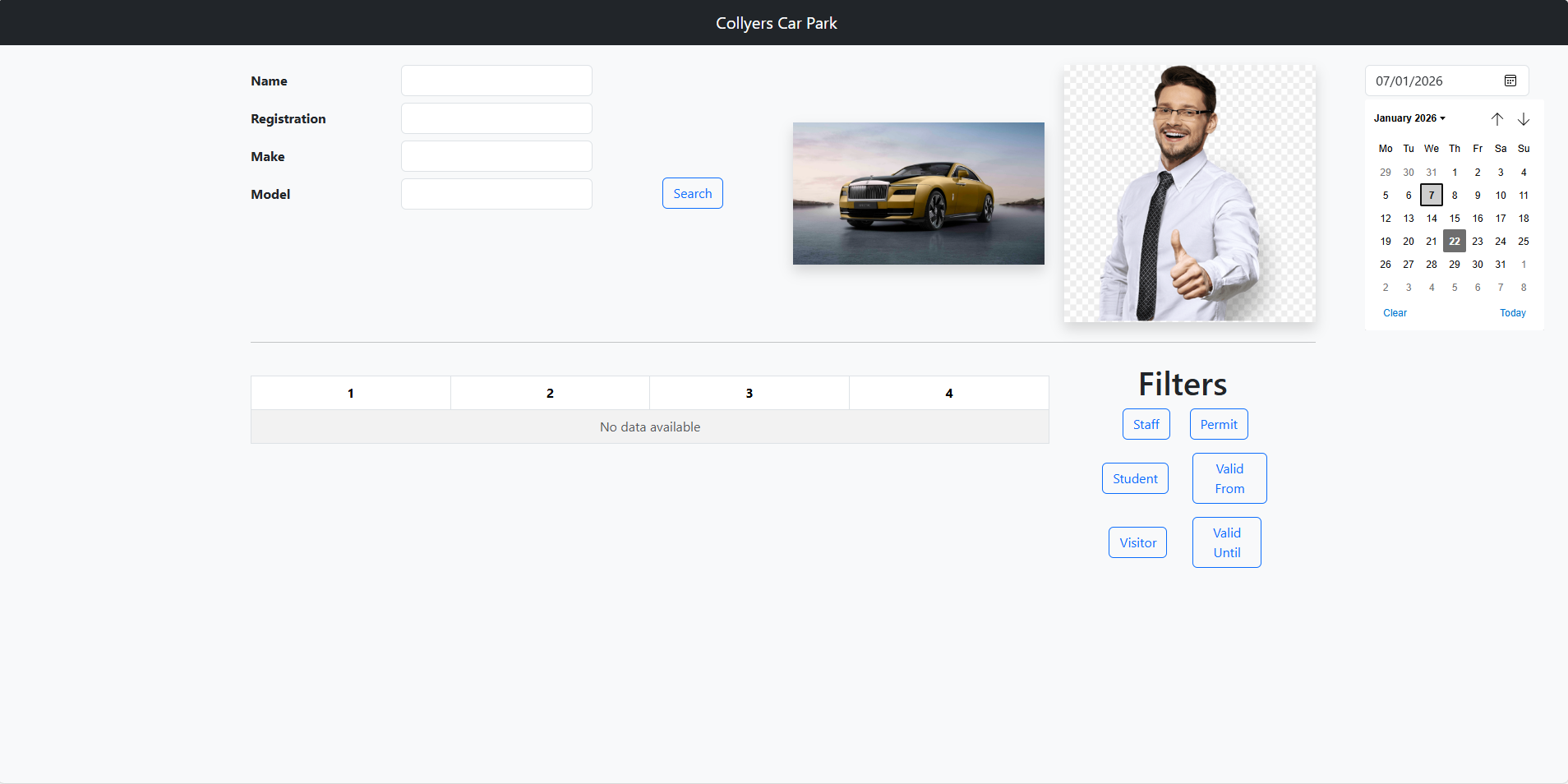
Return data

## Testing

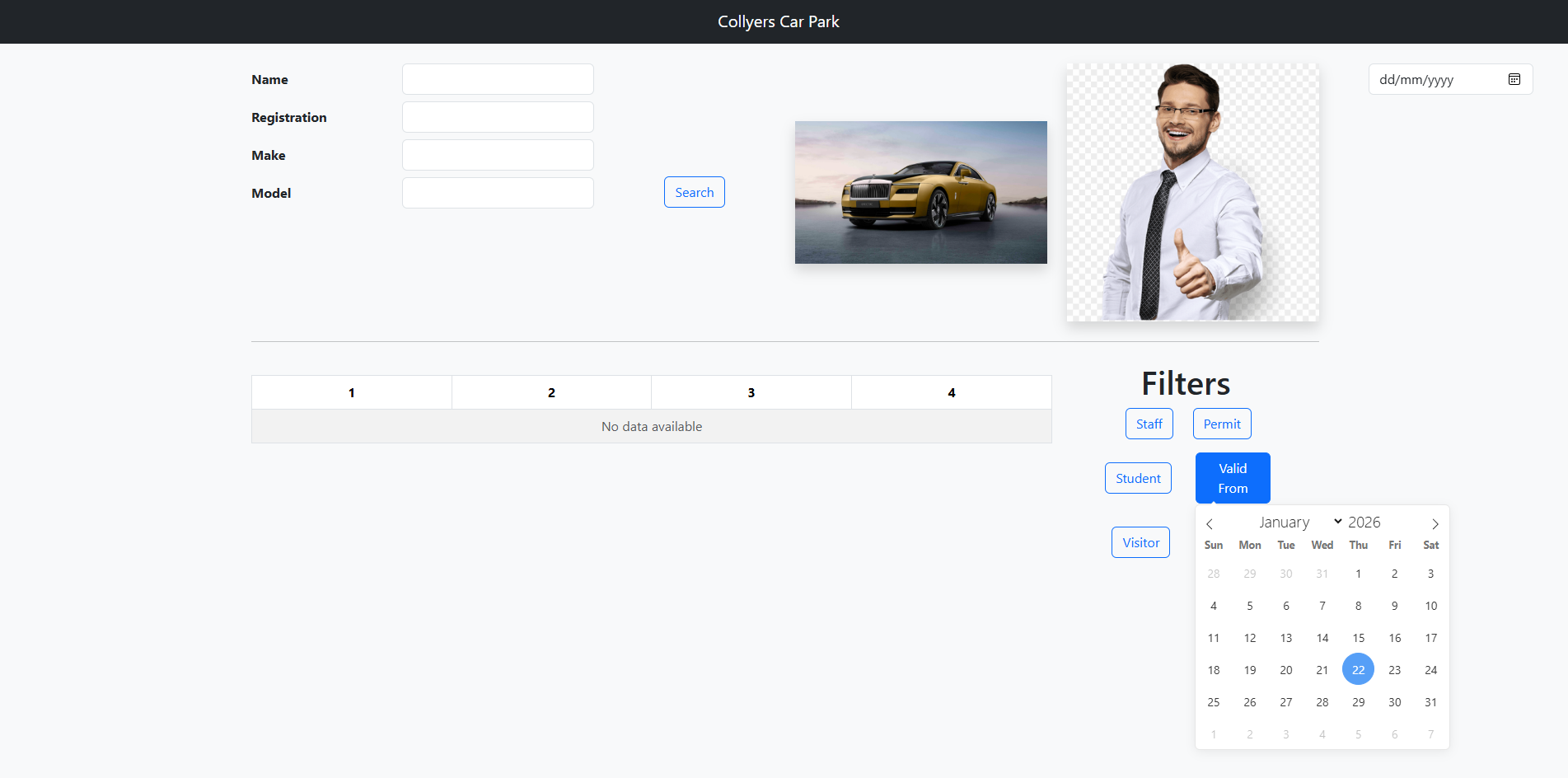
### Developmental testing

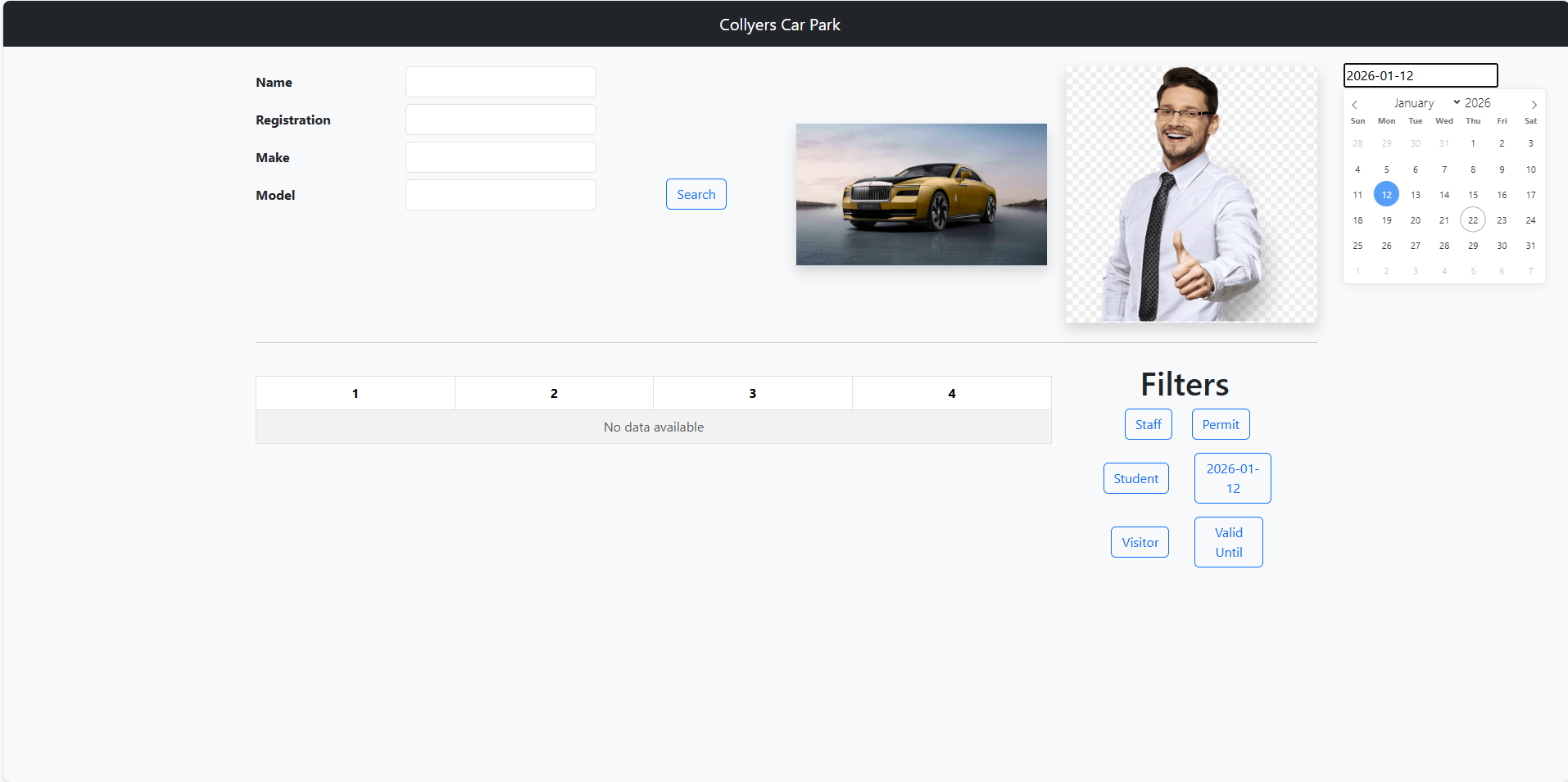
You should show your problem solving of at least 5 problems you have encountered. This should be evidenced with before and after images of the problem and an explanation of how you solved it.

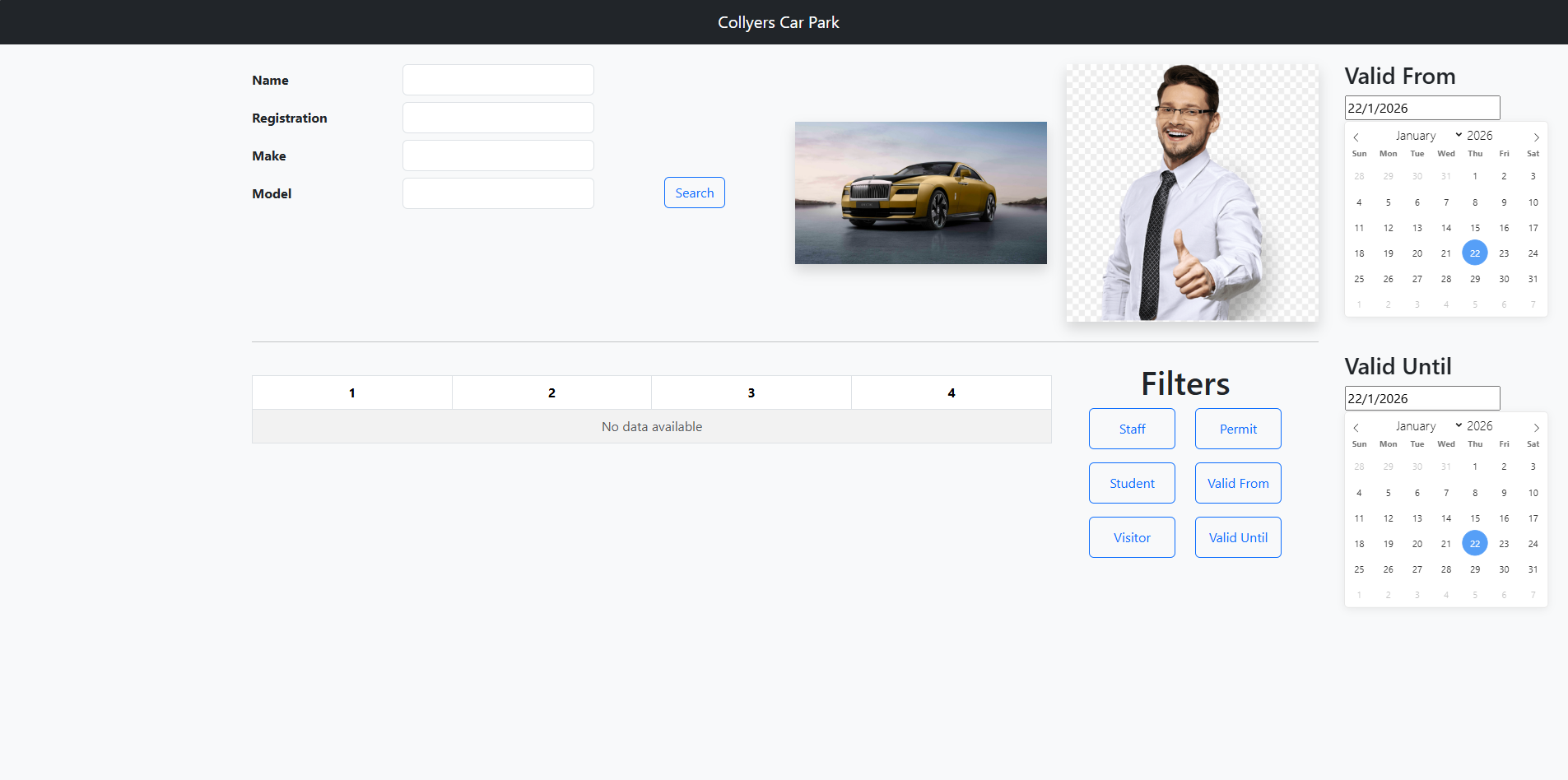
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



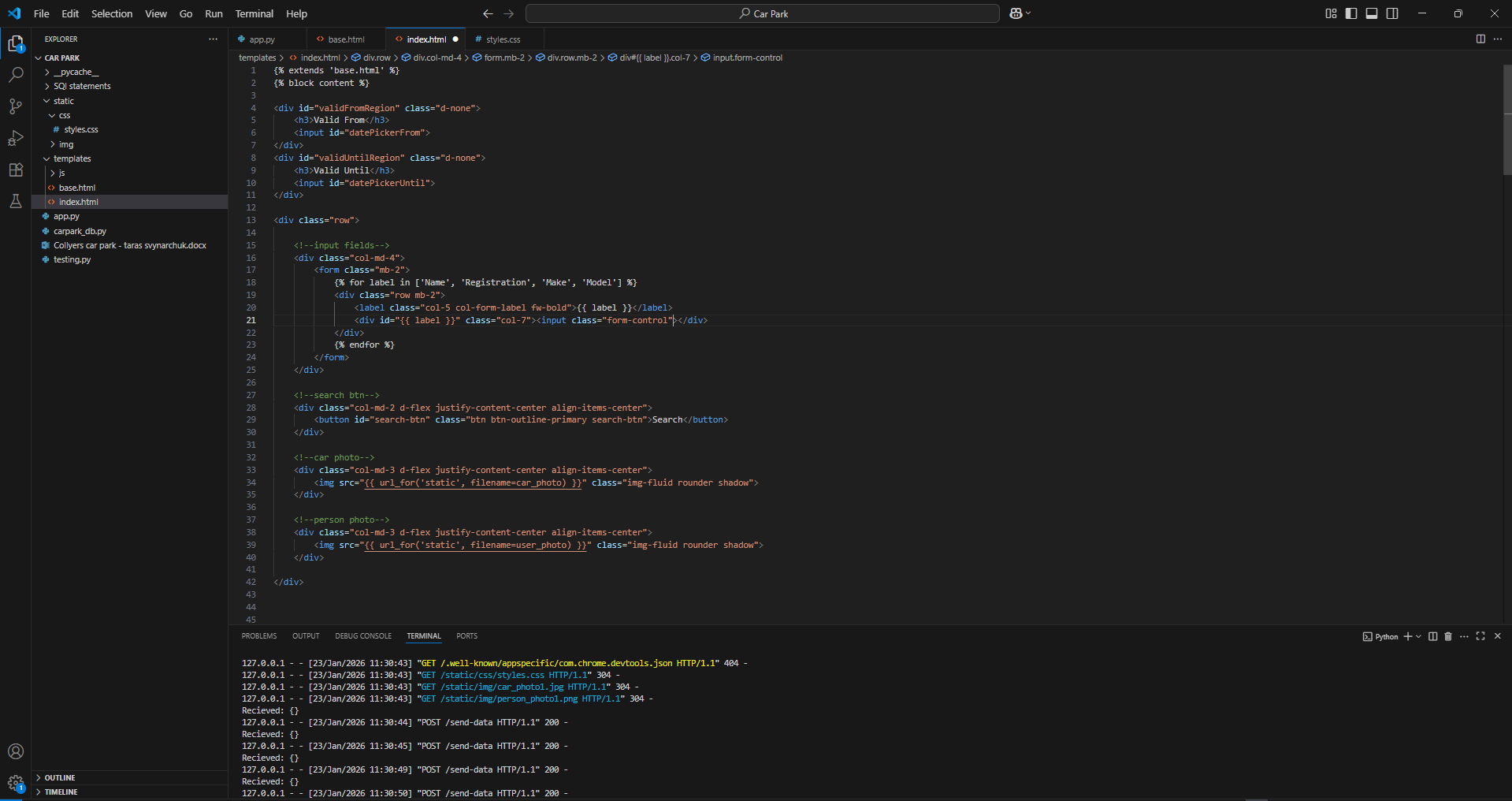
Ive managed to get a dropdown date selector to show up on the right hand side where there was lots of space being wasted however i dont like the design of the selector as it looks a bit too bulky so i will try to find another more modern drop down and implement it.

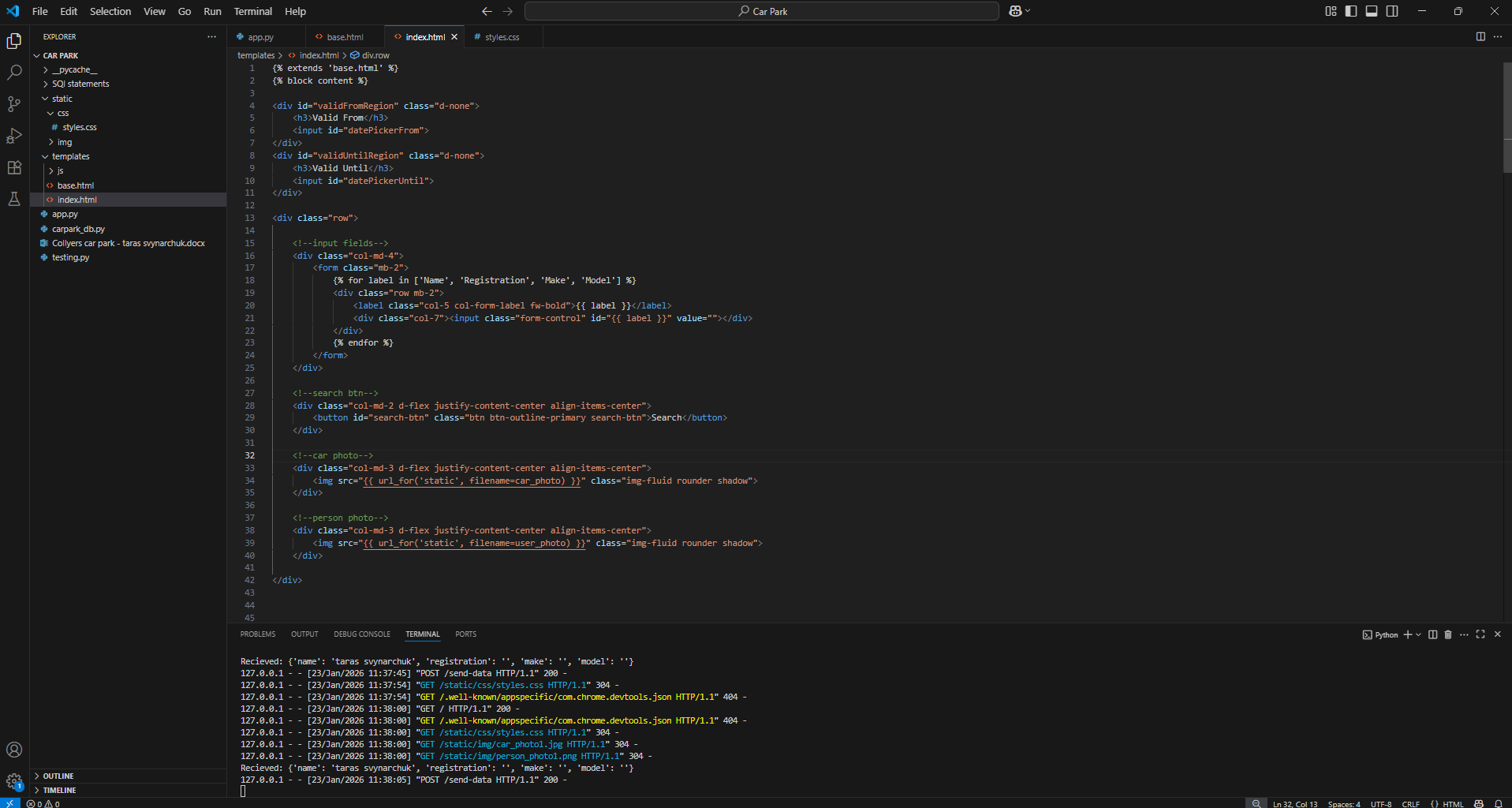
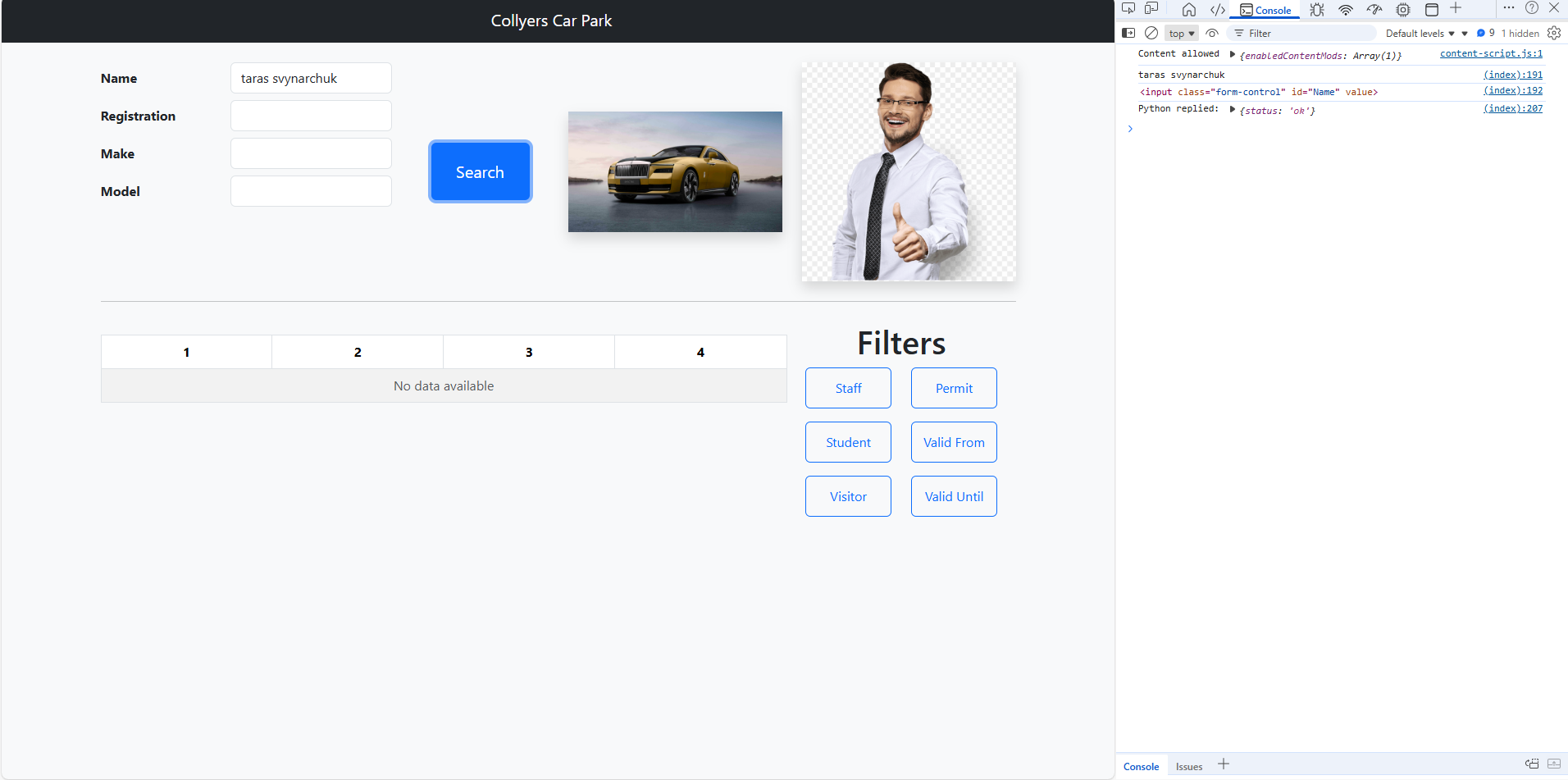
Ive found a more modern design which i think will suit this project much better however I couldn't get it to popup in the corner where the previous one was.

Now ive managed to place it in the top corner where i want it and also made it do the date selected would be the valid from box filters’ name.

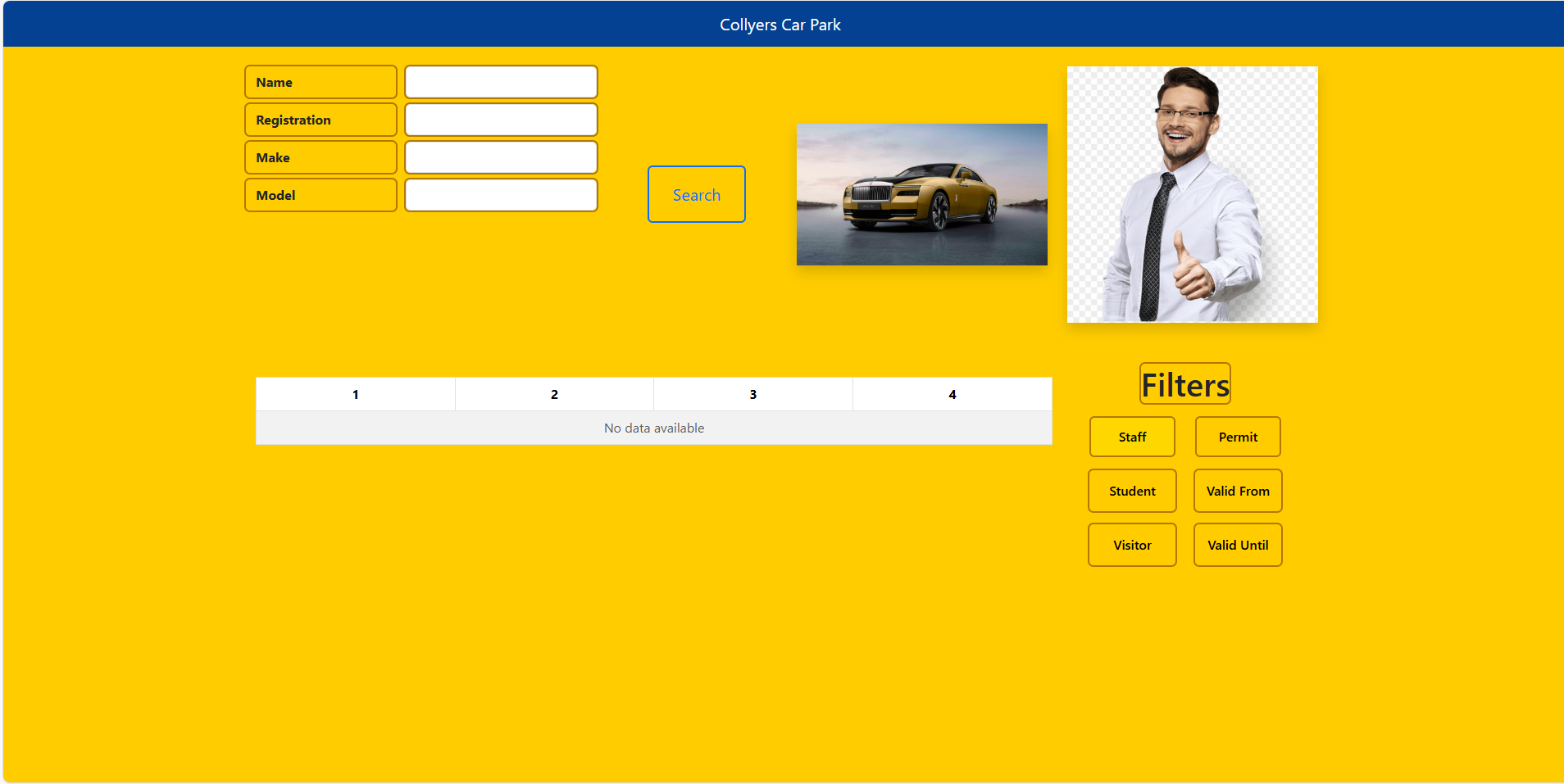
Ive essentially completed the date drop down and added some customization. Using a date drop down ensures no miss inputs can be made especially with the date selected being shown in three separate locations on the webpage and the search button must still be selected.

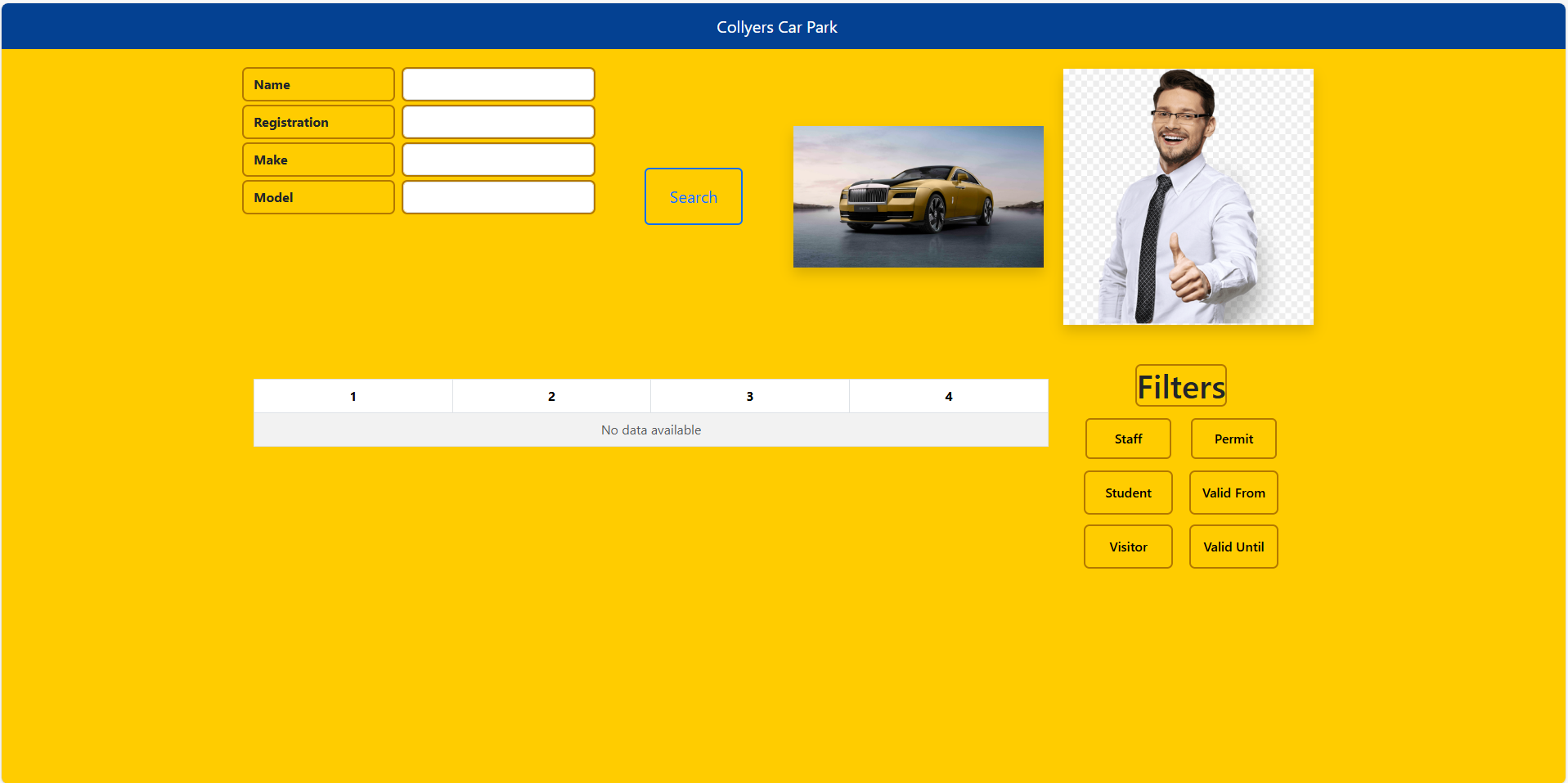
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

im trying to get the inputted fields to return back to flask through a json file but im getting nothing back but no error either

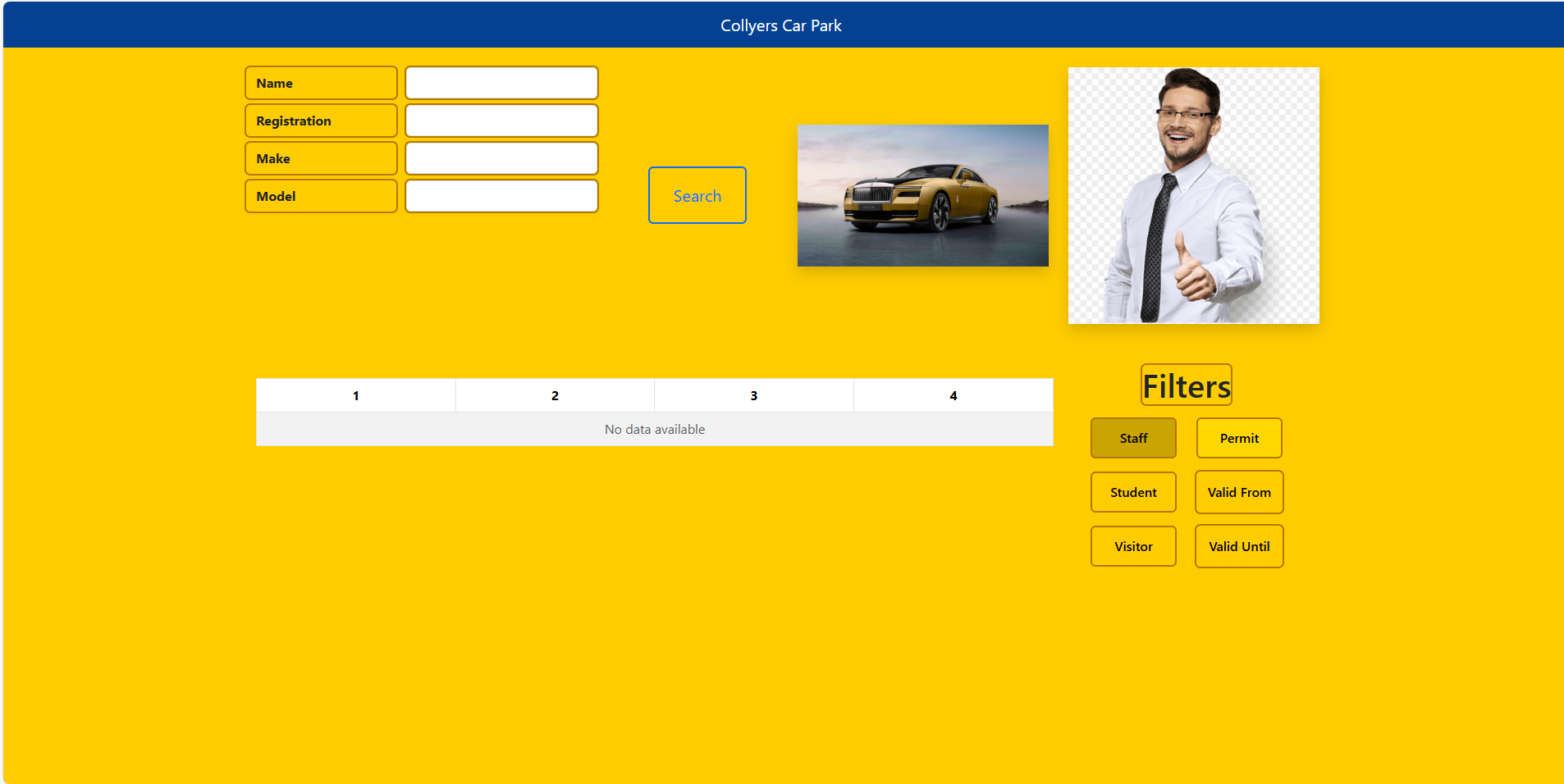
not ive gotten it to successfully return back the data – turns out I placed my id attribute in a div rather than in the input itself

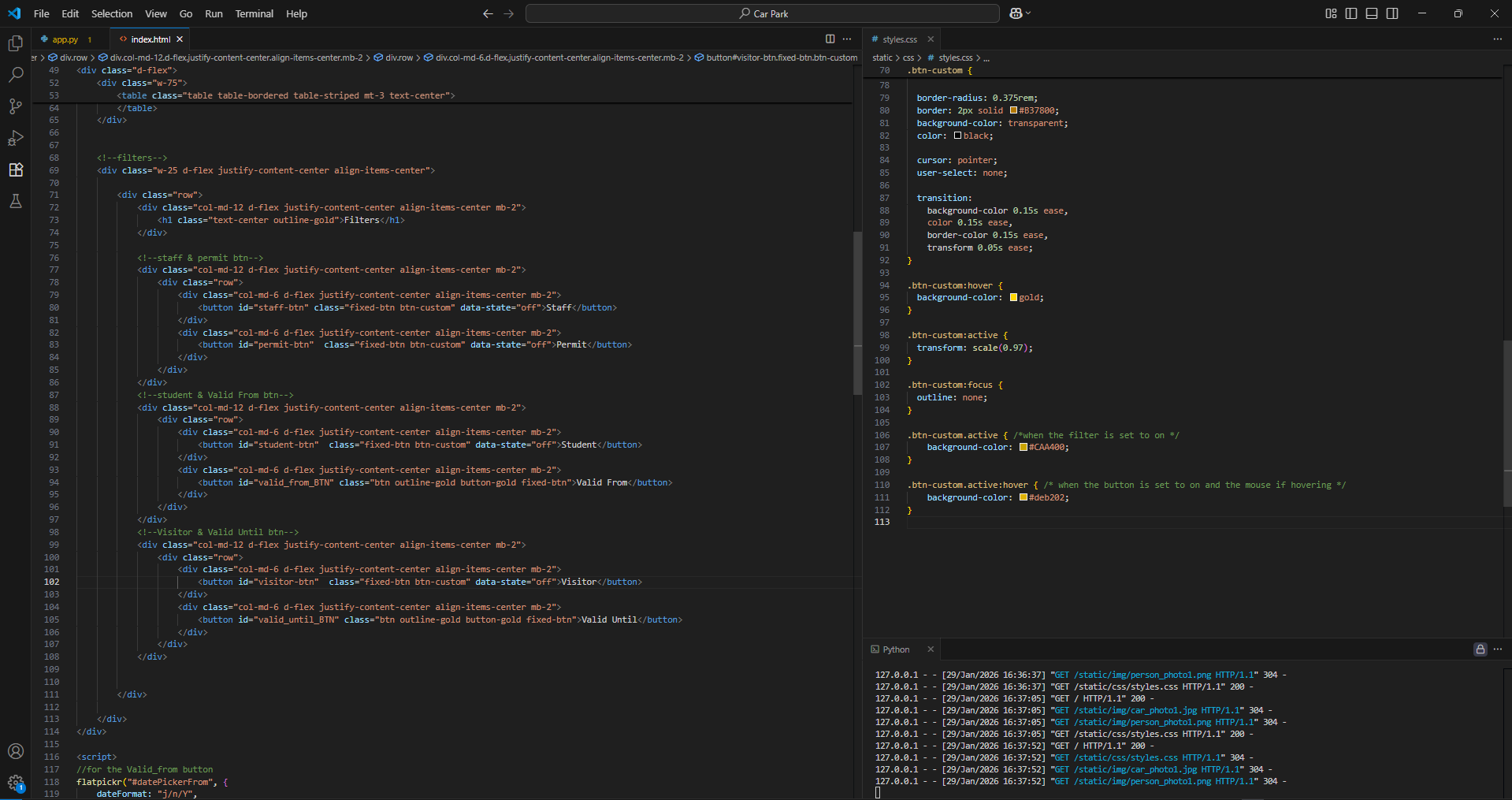
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

You can see that the staff button is a different colour when hovering over it but once clicking on it, it will never change colour again when hovering over.

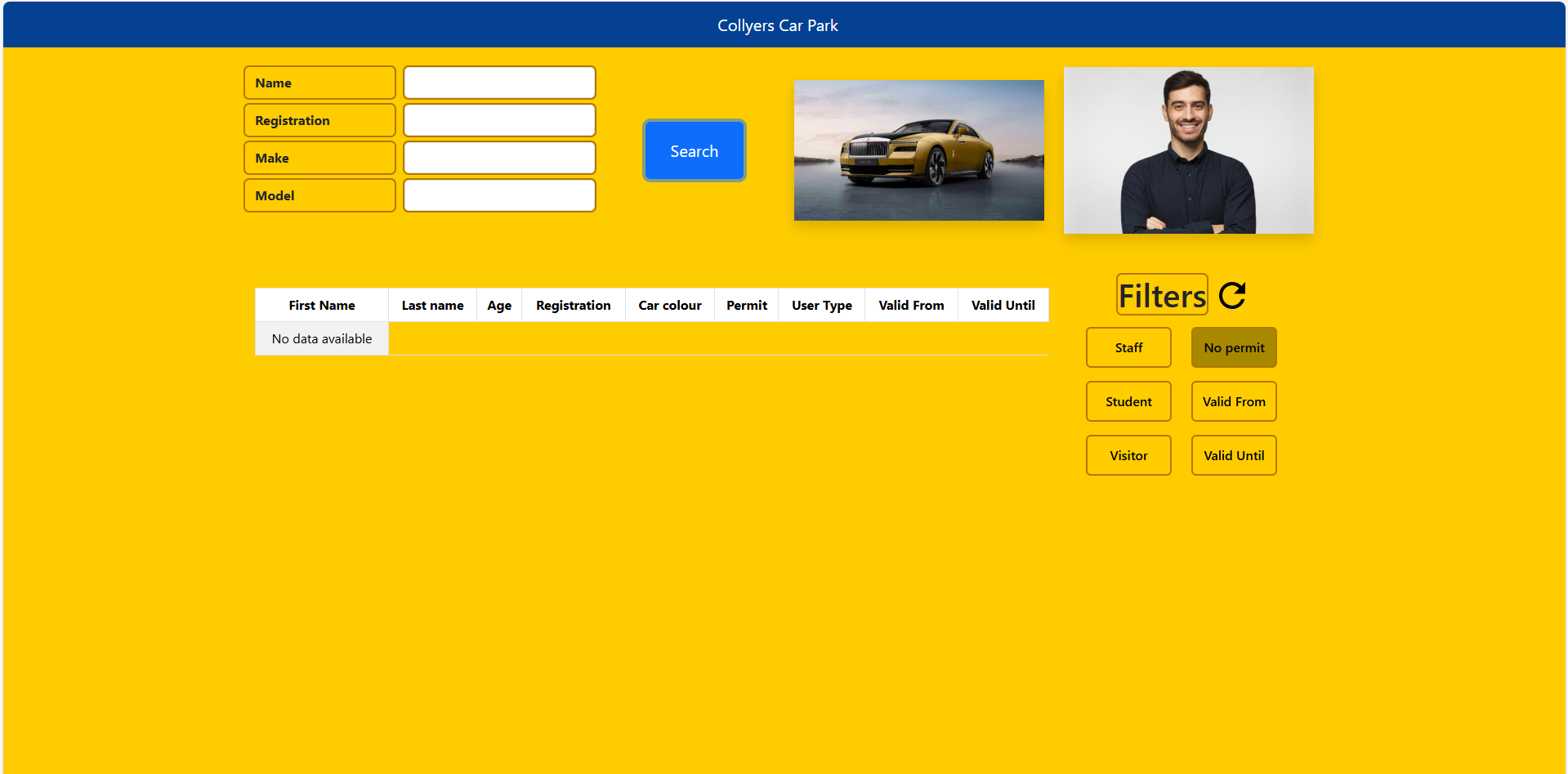
Heres after clicking on it.

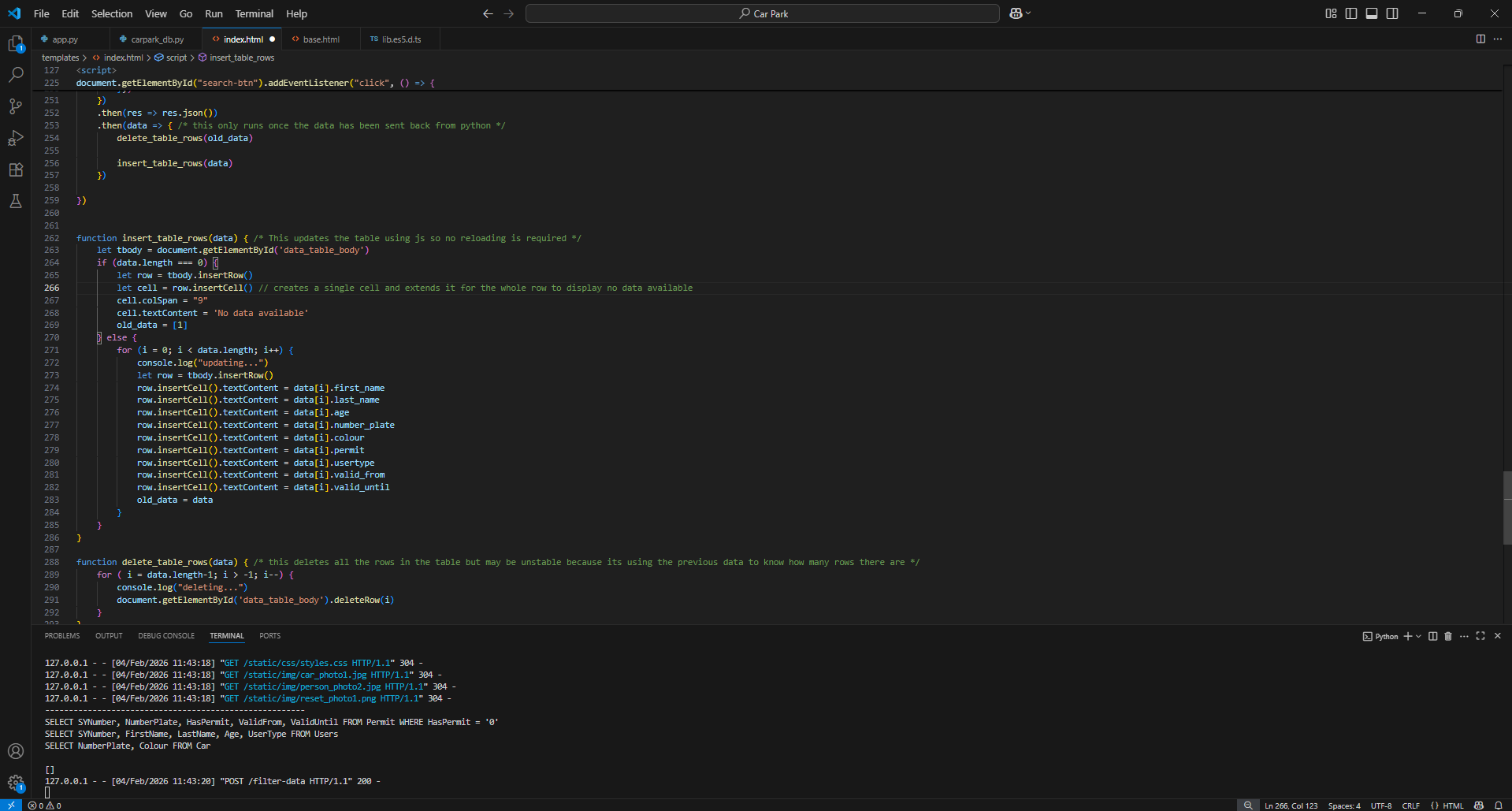
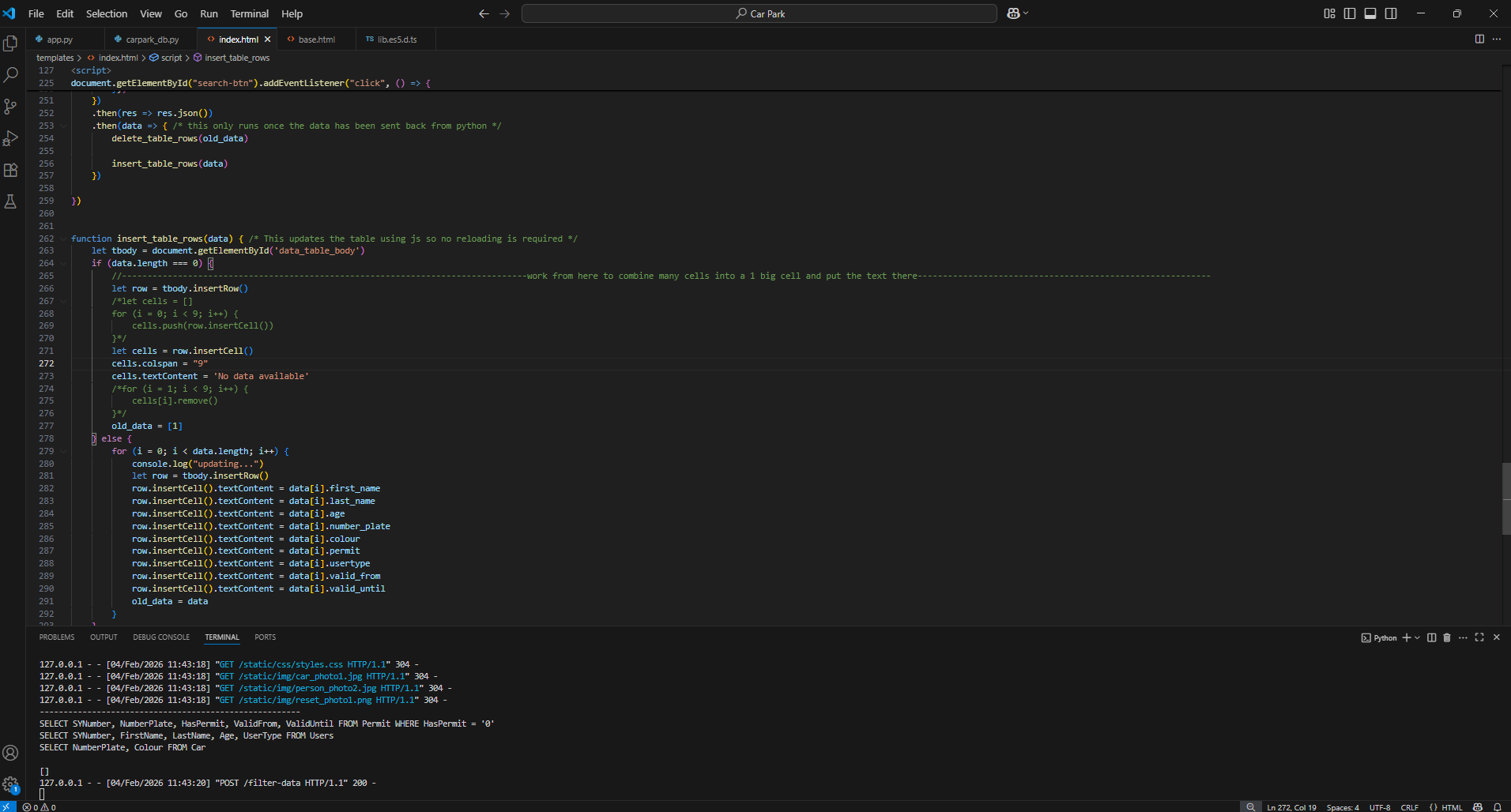
To solve this i think i will need to make new css classes to handle all the on/off states of the buttons.

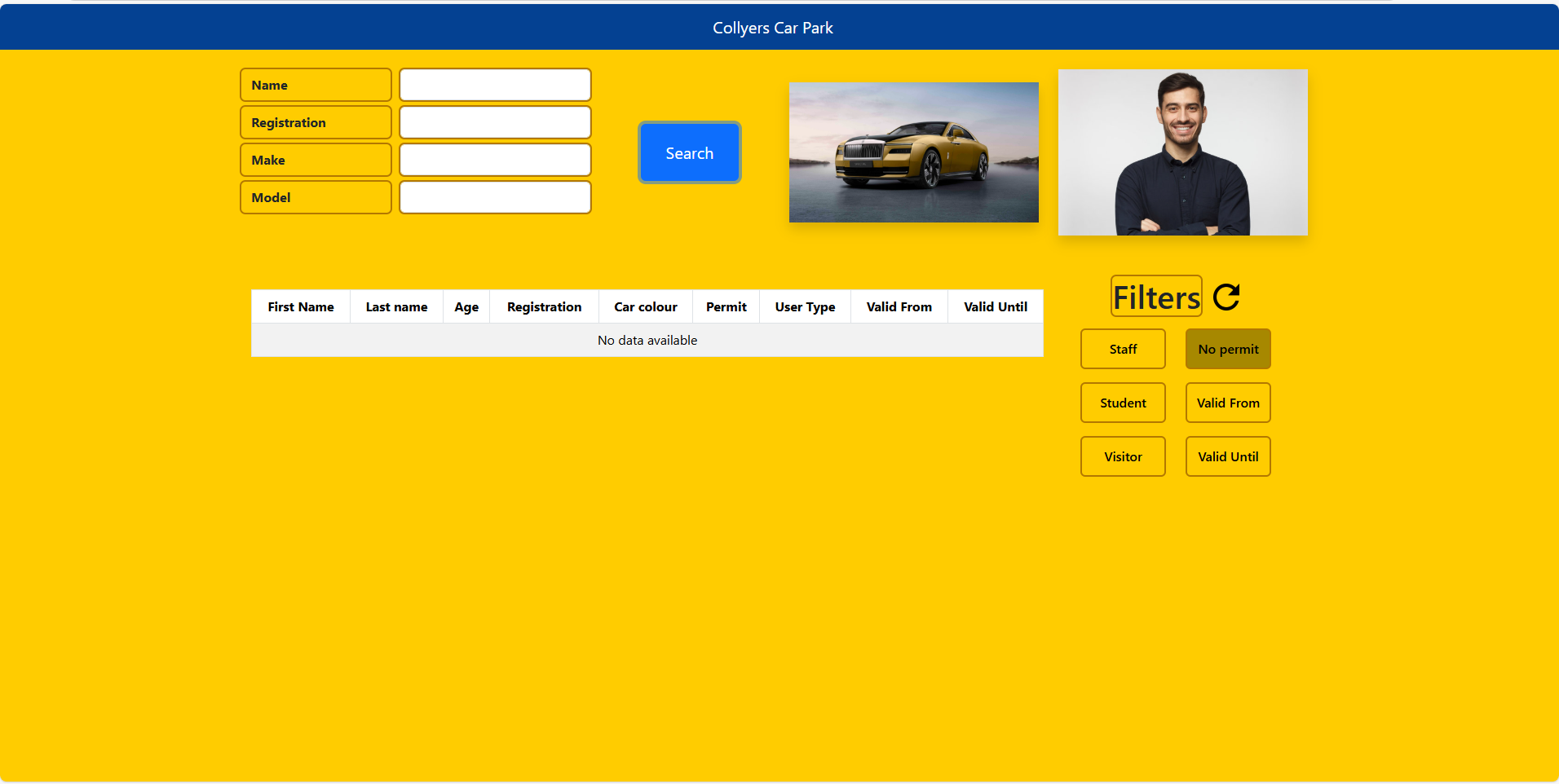
Now it works as intended and all i have to do is fix the colour scheming.

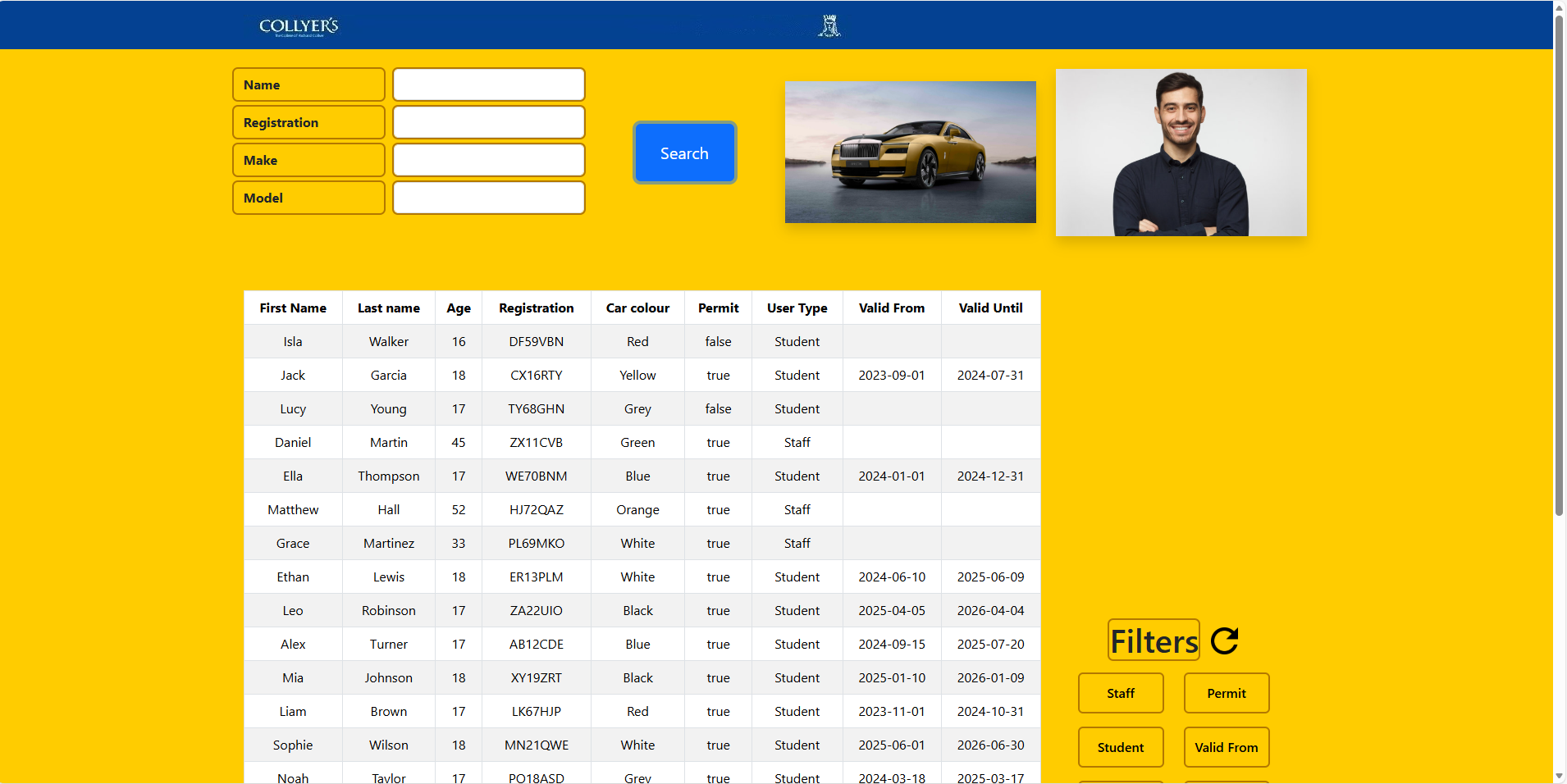
To make it work i have an active class which is toggled on/off by my javascript on the click of the button

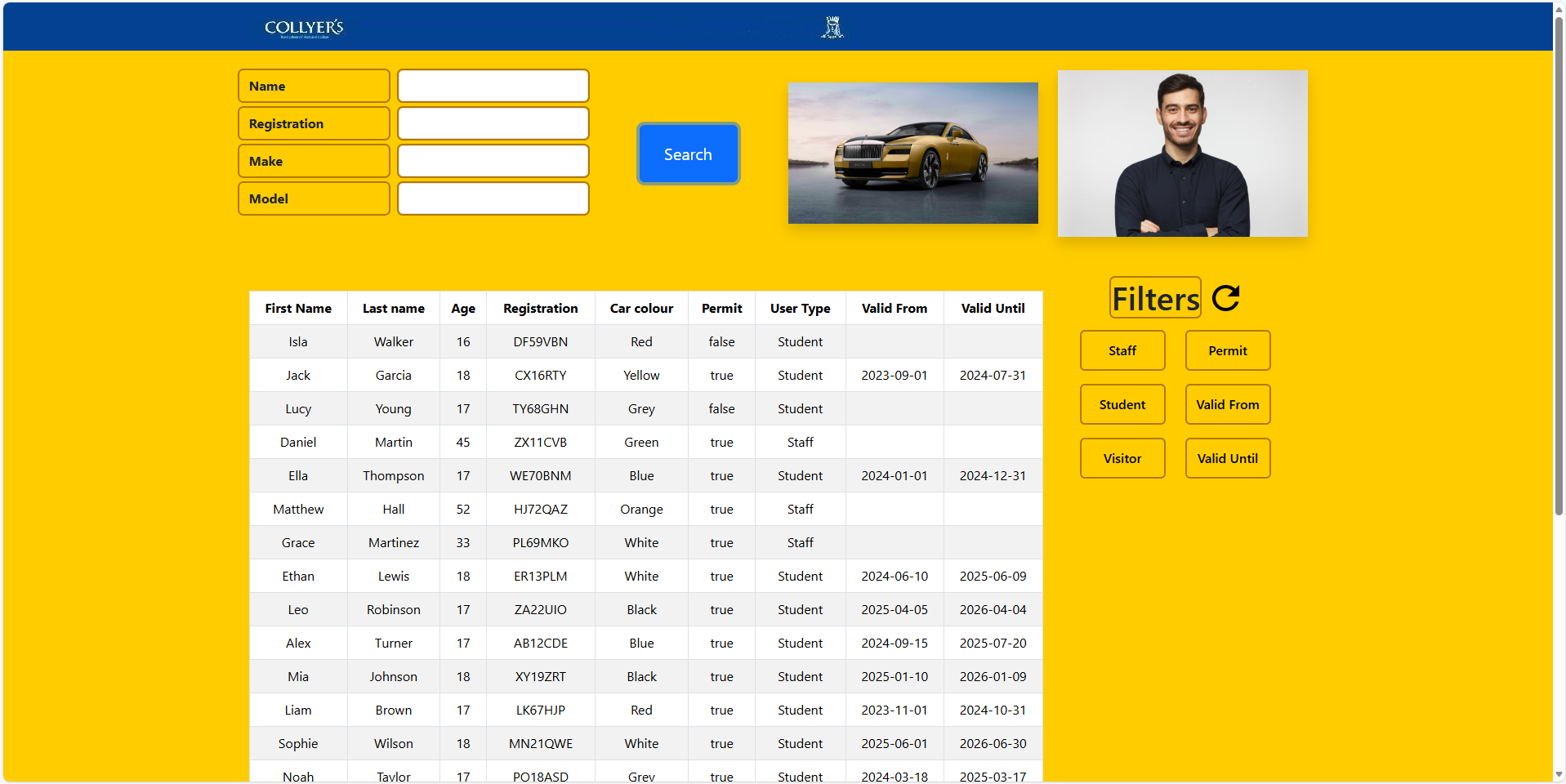
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

The cell displaying no data available is not extended for the whole row

I solved it by using camelCase for the colSpan property of the cell element as i was using colspan earlier which doesnt exist.

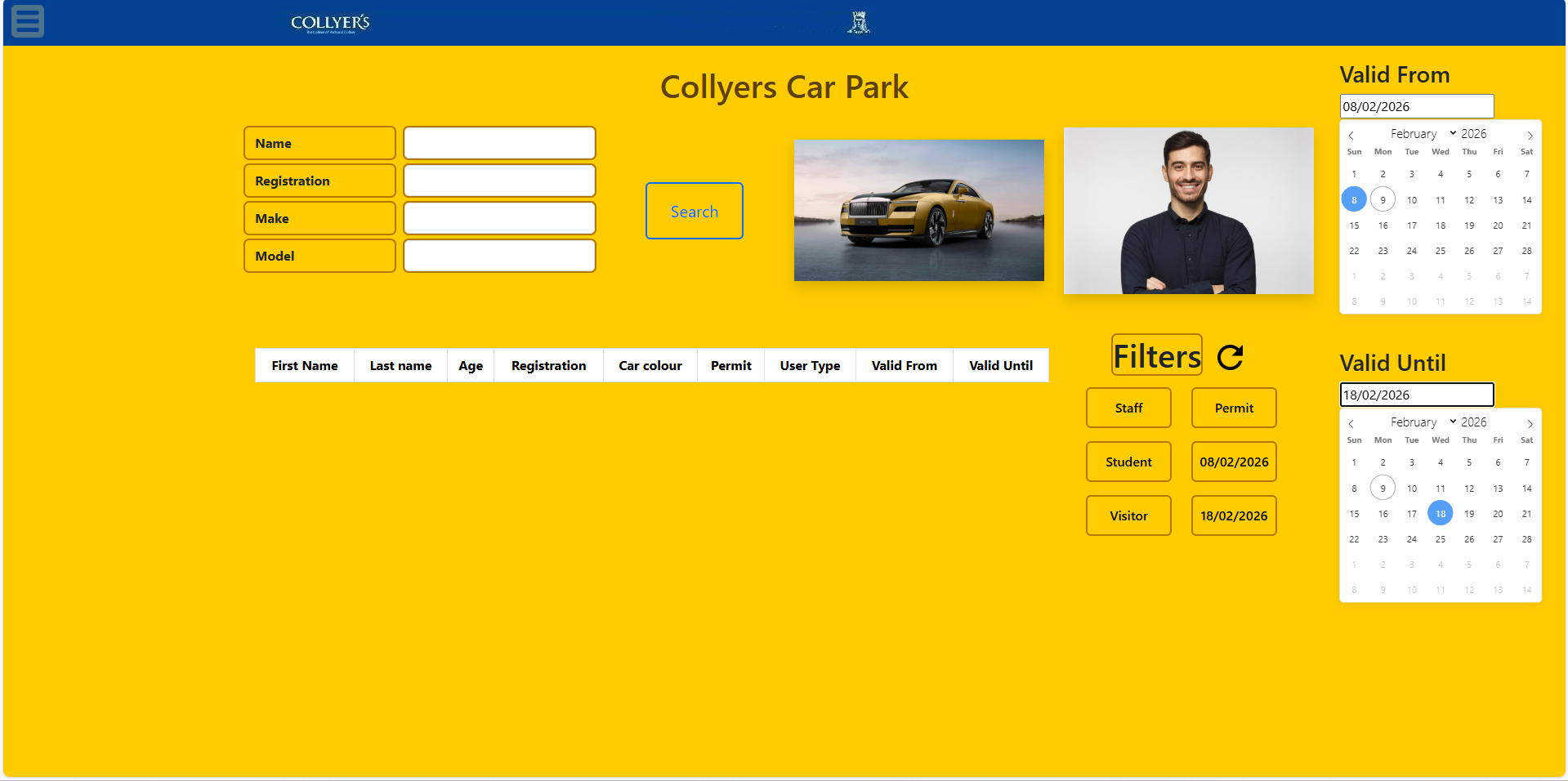
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

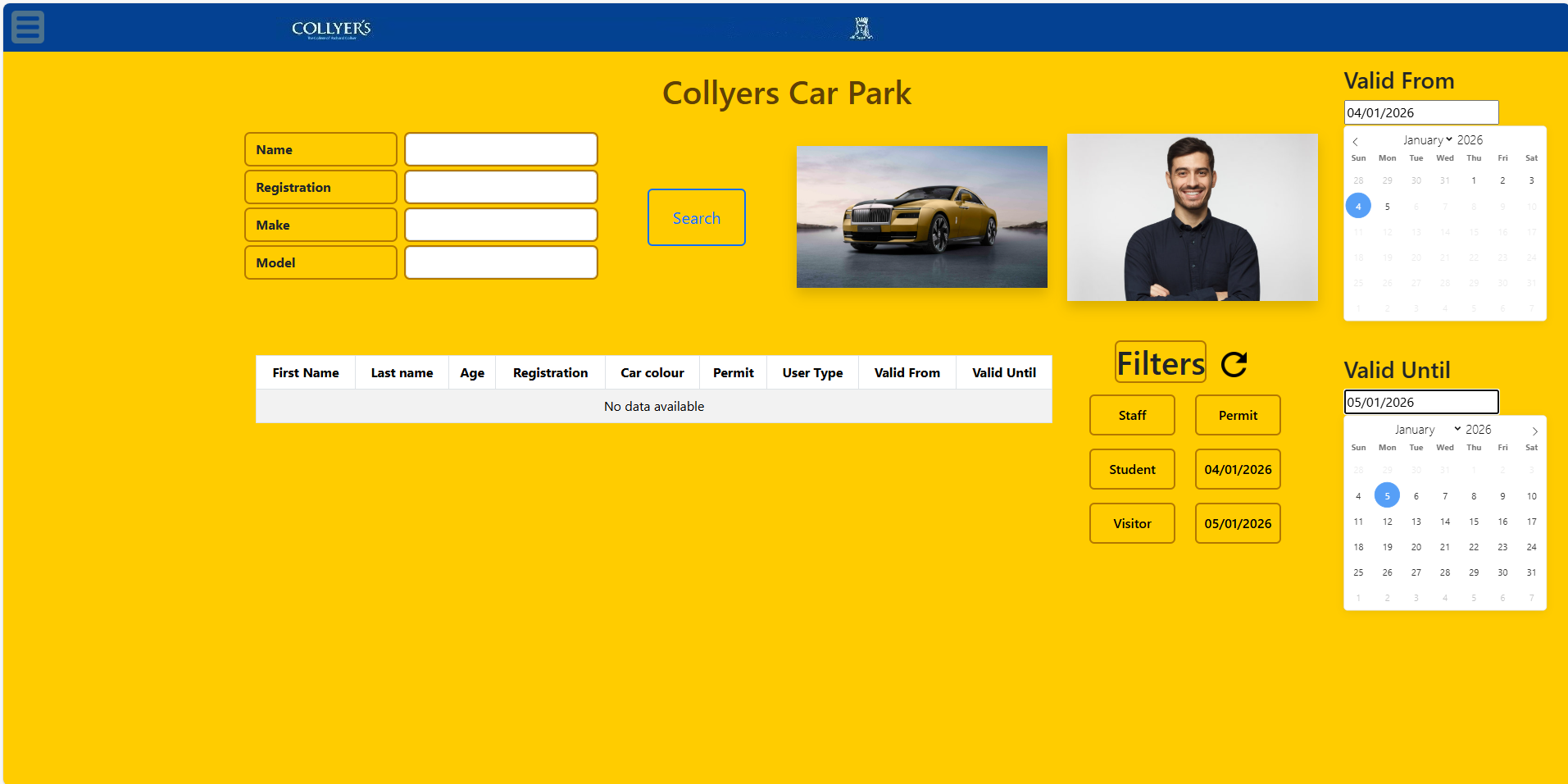
After adding more data to test for bugs in the ui i found that my filters shift down the page a lot – to halfway of the height of the results table. To solve this i will try either lifting the filters and keep them stuck to the top of their column/region or i will limit the height of the results field so the filters will be in the screen and so that user can scroll through the data within the smaller region

Is was a quick fix due to bootstraps large capabilities – i used the first option here and made the filters stick to the top of the region. However i dont really like how the table is going out of the page so i might implement a fusion of both ideas so that the filters do not keep changing location based on the amount of results in the table and keep them limited to a max of say 10 records and the user can scroll further if needed.

Ive managed to get both features to work now.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

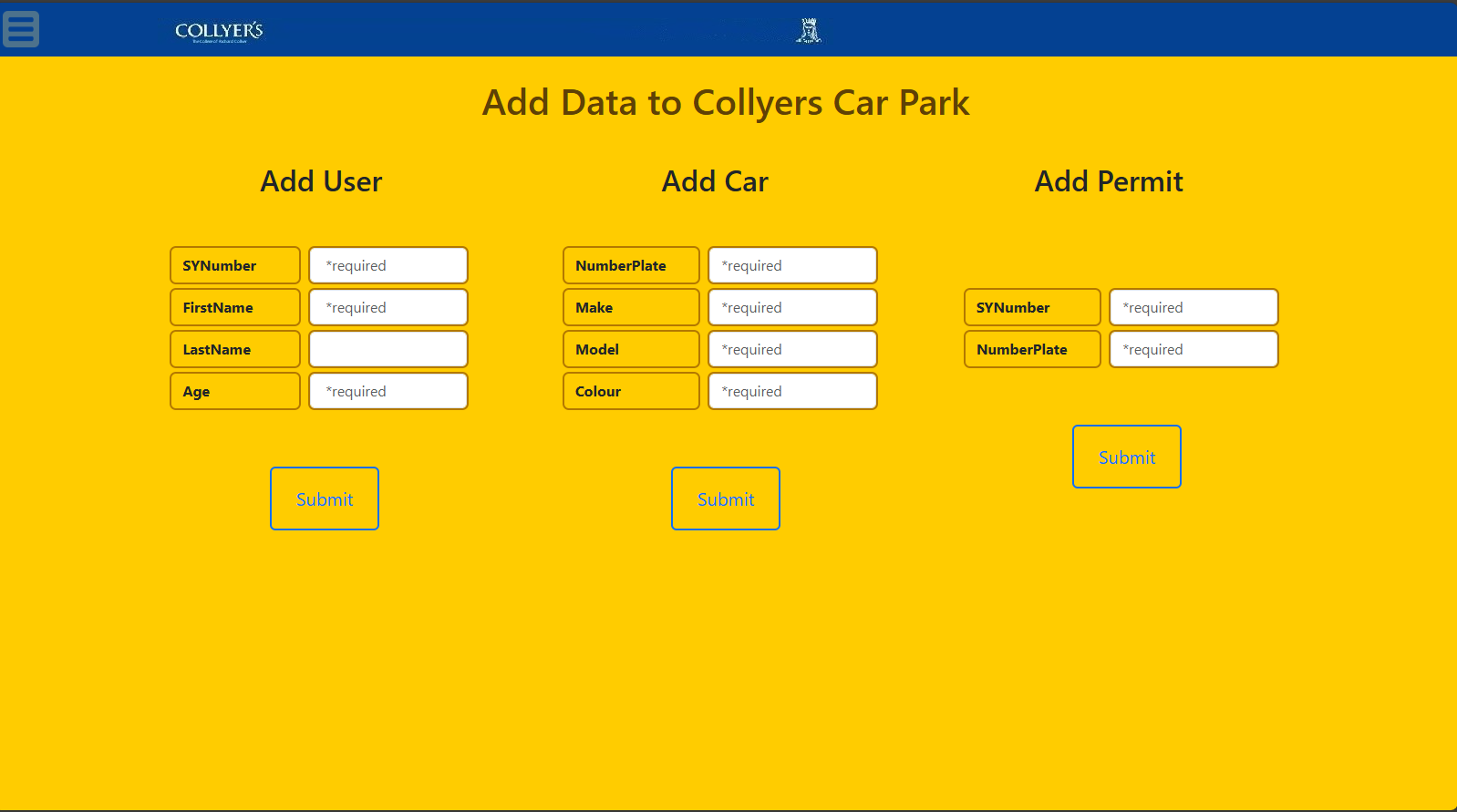
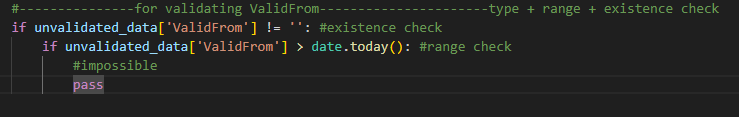
I wish to make the date select drop downs restrict dates which would be logically impossible such as haveing a permit valid until a date that is before it was valid from so i will search for an inbuilt function from flatpickr since i am using their date drop down.

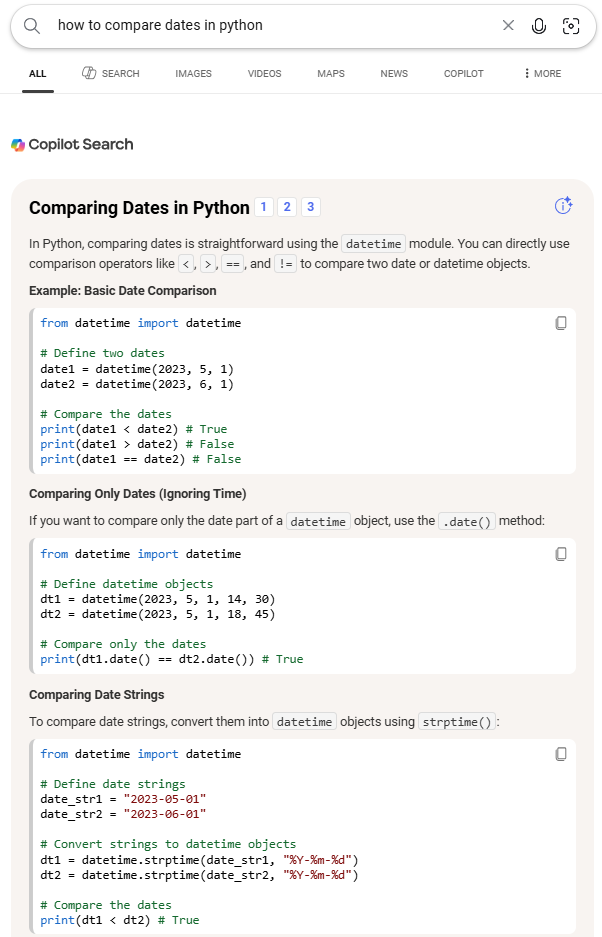


Ive managed to get it to work as intended using the code on lines 137 and 153

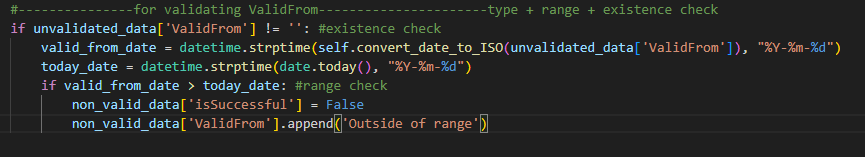
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

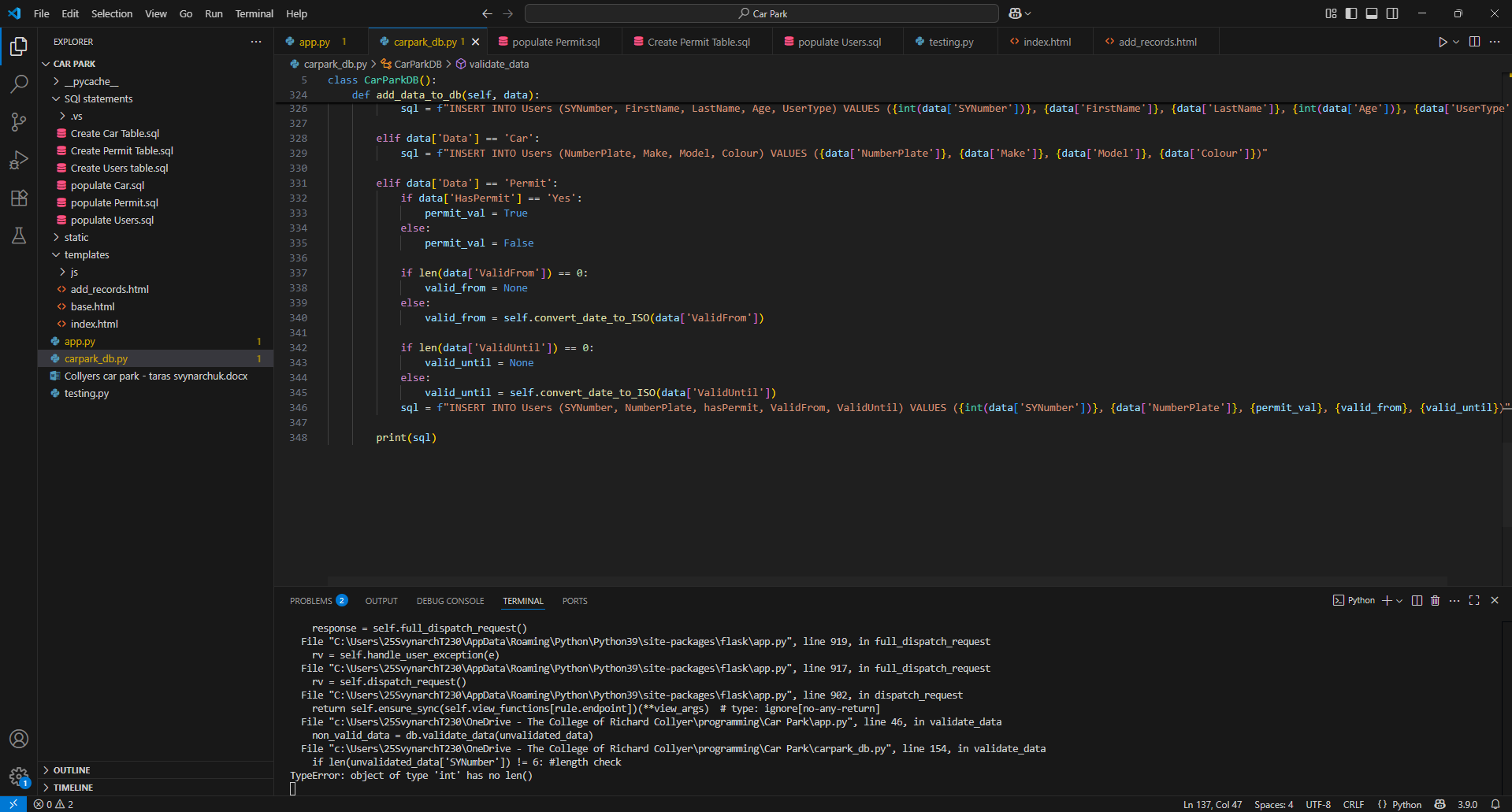
Ive managed to get the skeletal structure of the page complete but i dont like how the add user columns (which should be changed to car and permit respectively) are not in line so i will put them outside of their columns

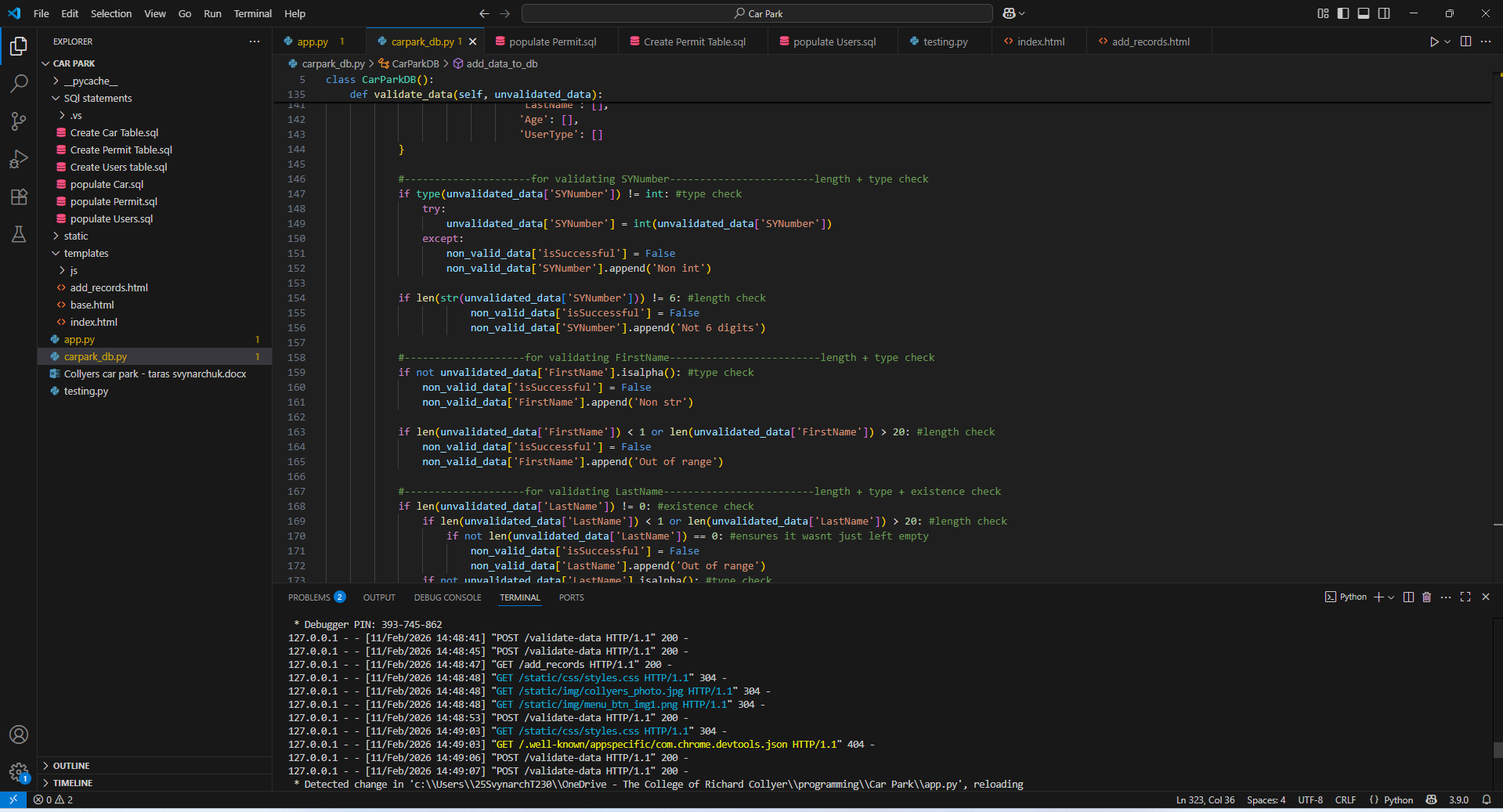
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_i need to be able to compare the date of a valid from date field and make sure it is not after today as they would not be possible – the user would not have the permit at this point in time so i will search how to compare dates



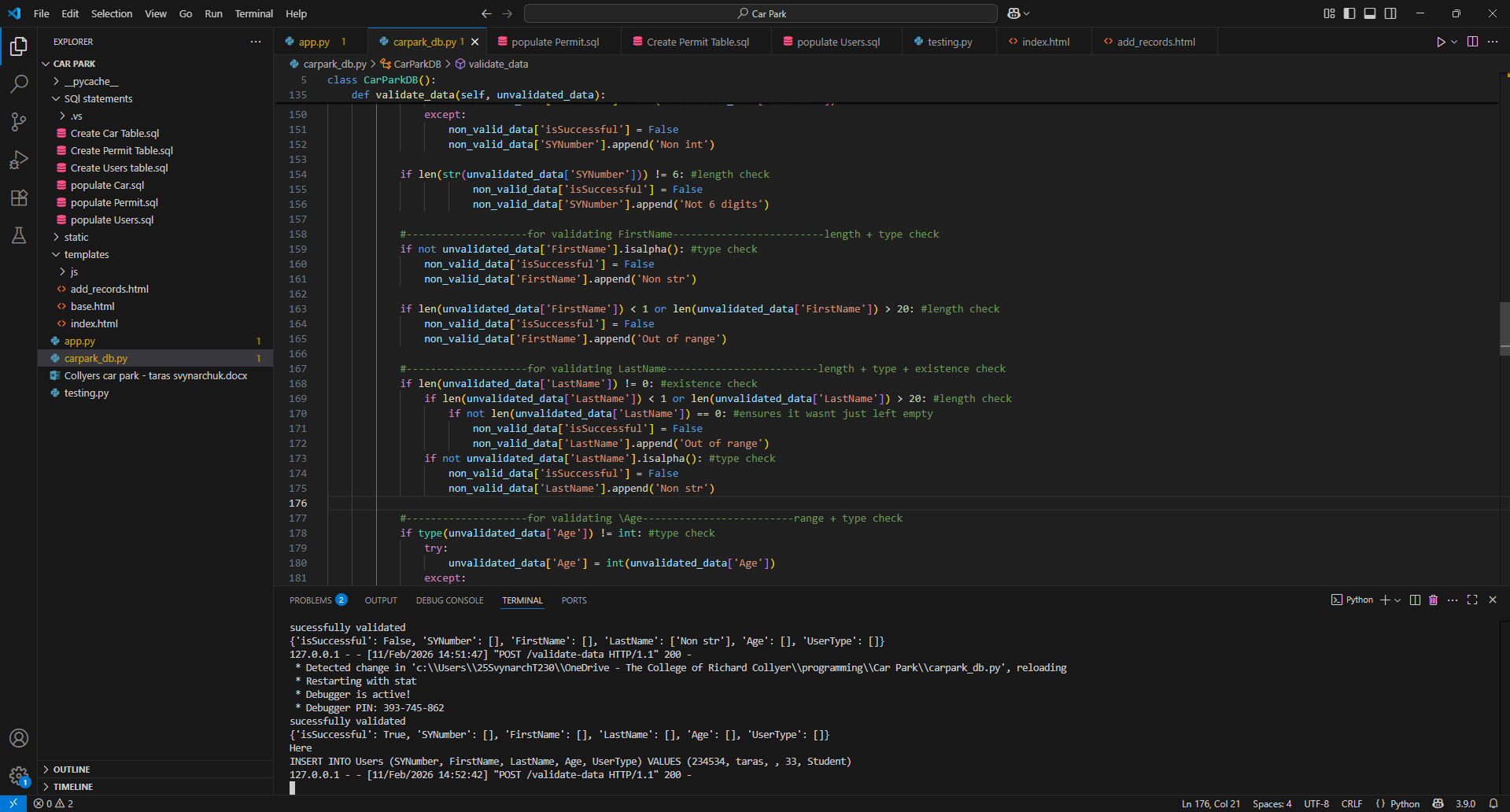
After analysing all 3 comparison methods i will implement method 3 since i have my dates already converted to ISO standard so it will be simple to get functioning

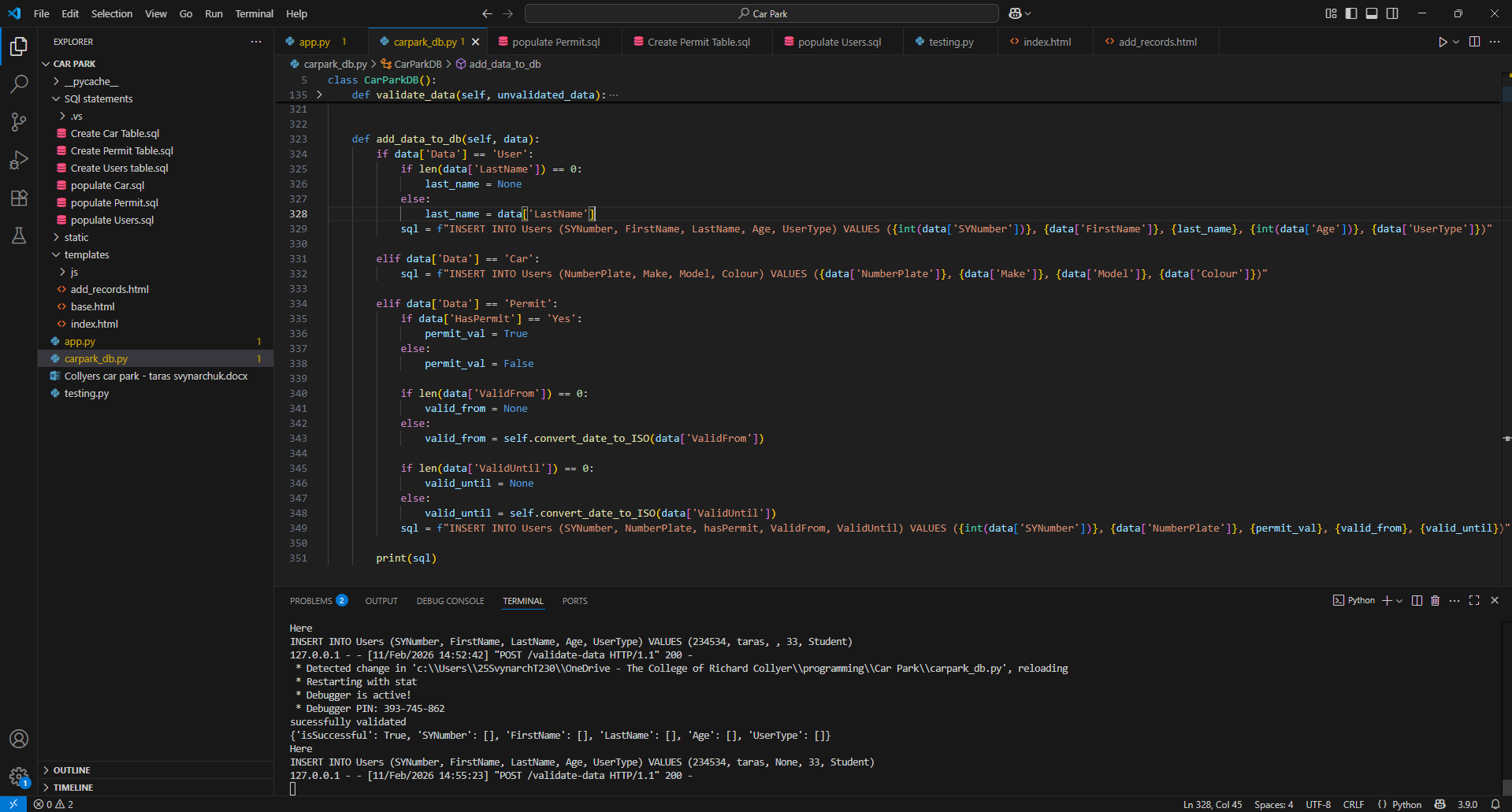
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Here i was attempting to use the len() function on and int but i should have casted the data to a str before since the len() function only works for str data types



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

My sql statement came out with a blank space for the LastName field which is because i didnt enter a last name although it is not required – so i will make a final check before creating the sql statement by checking for the last names presence

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### Testing

Once your system is complete, it needs to be fully tested. You should test the functionality of the system referring back to the objectives you set out.

You need to have evidence of each test, this can be a screen shot or video evidence.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test number | Description | Test Data | Test Type | Expected outcome | Actual Outcome |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

## Evaluation

You should consider the following points

* The effectiveness of the language and the libraries you used – PyQt5 rather than Tkinter, SQL etc.
* Identified good features and shortcomings.
* Explained any future improvements you would have liked to include.
* Evaluated personal strengths and weaknesses.
* What would you do differently for your next project.