

**Bachelor of ICT Assessment Cover Sheet**

Complete and attach this cover sheet to your assessment before submitting

**Course Code and Title:**

IT7405 – Web development using Non-relational Databases

**Assessment Title:**

**Learning Outcomes:**

LO1 – Demonstrate critical knowledge and understanding of the fundamental concepts and techniques of Data Mining

LO2 – Utilize data mining software to solve defined and undefined problems

LO3 – Interpret and evaluate obtained numerical and graphical data to recommend the best model to be used for prediction purposes

Individual Project

201901558

**Student ID**

Ali Husain Annan

**Student Names:**

**Tutor:**

**Due Date:**

Sini Raj Pulari

Wednesday, 20th Dec 2023 by 11:55 p.m.

**Late Rule:**

**The maximum grade granted for late submission is 60 % for up to 2 calendar days. A grade of 0 will be allocated for submission after 2 days**

***By submitting this assessment for marking, either electronically or as hard copy, I confirm the following:***

* *This assignment is* ***my own work***

#### Any information used has been properly referenced.

* *I understand that a copy of my work may be used for moderation.*

#### I have kept a copy of this assignment

Do not write below this line. For Polytechnic use only*.*

**Assessor:**

**Date of Marking:**

**Grade/Mark:**

Table of Contents

[Task1: Problem Statement Formulation and definition 3](#_Toc153919049)

[Task2 – Creating the No-SQL MongoDB Database and Data Modeling 4](#_Toc153919050)

[Creation of a No-SQL MongoDB 4](#_Toc153919051)

[CRUD operations 4](#_Toc153919052)

[Create 4](#_Toc153919053)

[Read 6](#_Toc153919054)

[Update 8](#_Toc153919055)

[Delete 9](#_Toc153919056)

[Usage of MongoDB Index 10](#_Toc153919057)

[Query Diagnosis and Analysis 10](#_Toc153919058)

[Task3 - Using Django to build the Web Application using Bootstrap 13](#_Toc153919059)

[Creation of VirtualEnv for Django 13](#_Toc153919060)

[Files structure 13](#_Toc153919061)

[Project settings used 15](#_Toc153919062)

[Django connectivity with MongoDB 16](#_Toc153919063)

[Django Template Language 17](#_Toc153919064)

[Model –View-Template 18](#_Toc153919065)

[Django Admin site 19](#_Toc153919066)

[Django forms 21](#_Toc153919067)

[Incorporation of Bootstrap 22](#_Toc153919068)

[Task4 – Overall GUI and working, Report, GIT hub, Video and Reflection 24](#_Toc153919069)

[Overall Navigational GUI 24](#_Toc153919070)

[Working of Web Application meeting all Functionalities 24](#_Toc153919071)

# Task1: Problem Statement Formulation and definition

From young age my love to the JDM cars was increasing specially the late 90’s generations as there were built with absolute perfection the design , engine , and exterior but unfortunately not everyone now the worth of these masterpieces and how valuable they are and sometimes the owner want to sell their cars but do not have the best place even some of the car enthusiastic would be grateful if they just took a photo with these cars as they are extremely rare and expensive even more expensive that a range rover , Mercedes or even a Porsche. As a car enthusiastic myself a website such as this will make a huge difference as many of us may do not buy or even book a filming session just seeing these cars would make me happy, moreover the websites to list and book a car shot appointment are rare and even rarer in our region, therefore I have chosen this idea

The aim of this website is to create a user-friendly website to give the cars there real worth when listing and it is not confused with the rest of low value cars as well as helping the car enthusiastic to have the opportunity to book a session take images with these vintage masterpiece by booking an appointments in addition to search and buy this cars from one location with no need to search in multiple websites or account to buy it with opportunity to test drive and inspect the can in person concluding the website have multiple features and functionality such as:

* Listing all available cars
* Buying one of the cars
* Booking an appointment to film with the car
* View the in-depth details and specs of the vehicle
* Login / Logout
* Register

# Task2 – Creating the No-SQL MongoDB Database and Data Modeling

## Creation of a No-SQL MongoDB

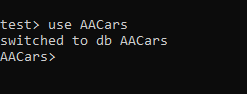


Figure Create the mongo database

## CRUD operations

### Create

* Insert One

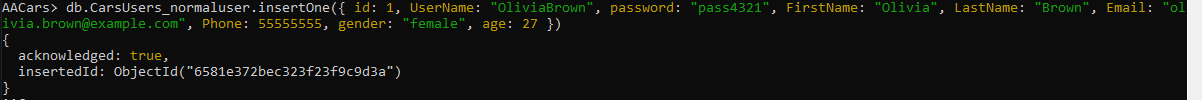


Figure Insert one document

* Insert Many



Figure Insert Two documents in one time

### Read



Figure Using the find command to read the data in the database

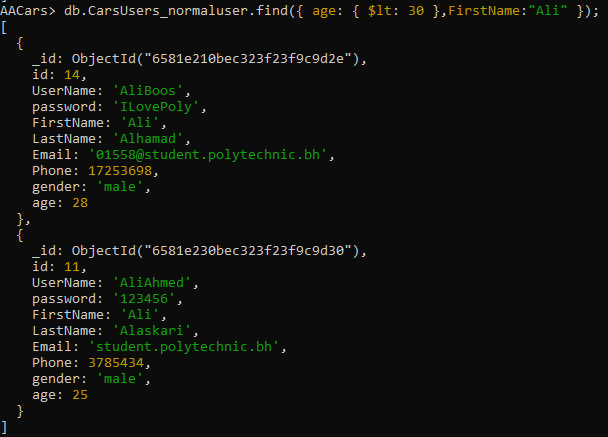


Figure This is a find command to search for age less than 30 and have the name Ali

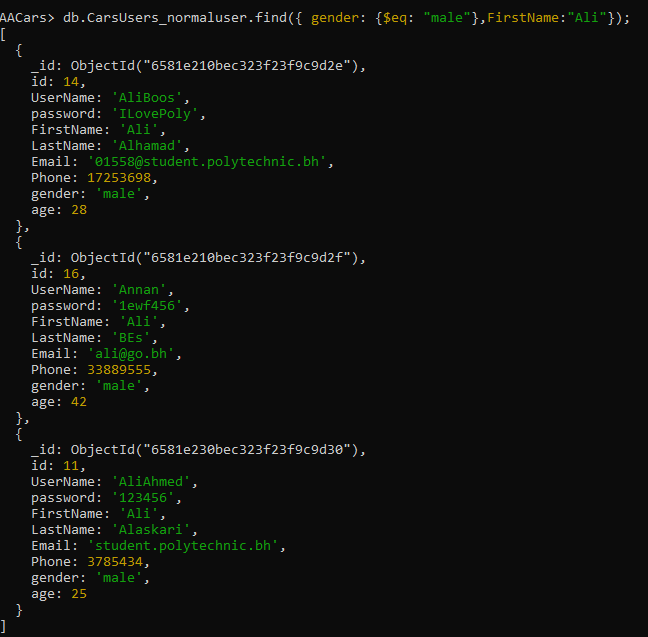


Figure Find command that gender must be male and FirstName must be Ali

### Update

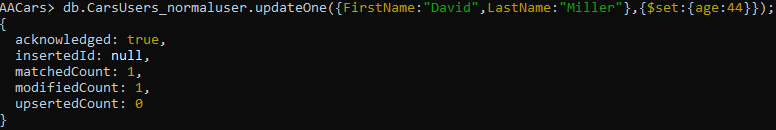


Figure Update one row where the first name is David and the last name is Miller and set the age to 44

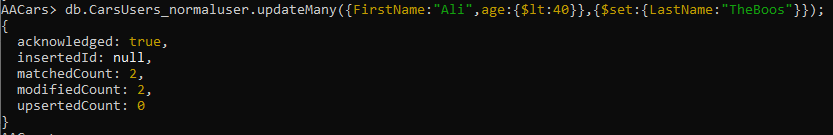


Figure Update many for all documents with the conditions

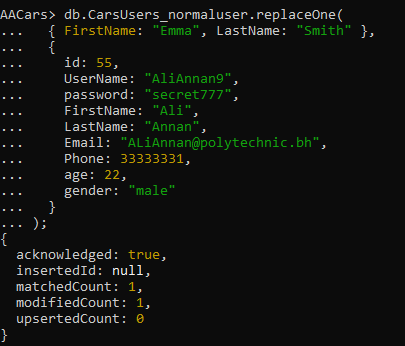


Figure Using the replace one command to place new document

### Delete



Figure Delete the document that match the username

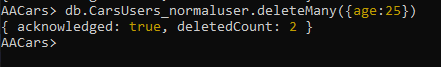


Figure Delete All the documents the have age 25

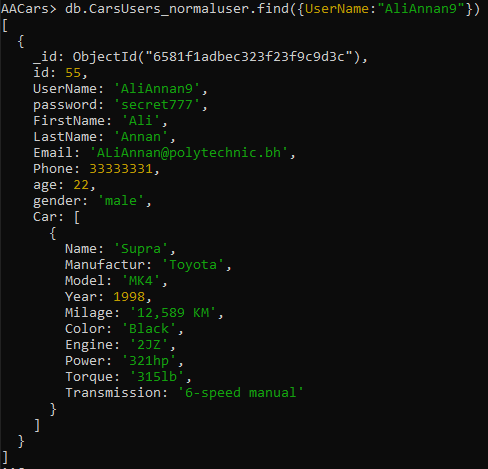


Figure The data modelling concept is embedded and the relationship between the car and the user is one-one

## Usage of MongoDB Index



Figure Creation of the index in the model field inside the car

## Query Diagnosis and Analysis

The "executionStats" option in the explain method is used to perform the diagnosis. This allows MongoDB to run the query optimizer to select the winning plan, carry out the winning plan through to completion, and return statistics detailing the execution of the winning plan.



Figure This is the analysis before creating the index



Figure This is the analysis after creating the index

# Task3 - Using Django to build the Web Application using Bootstrap

## Creation of VirtualEnv for Django

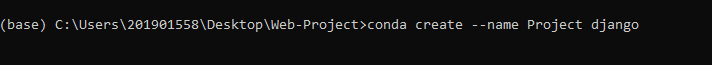


Figure Creating the virtual Env

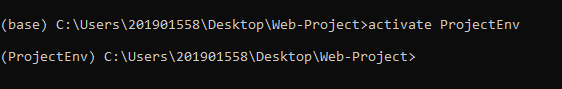


Figure Activating the virualEnv



Figure Create a Project







Figure Create App Cars.

## Files structure

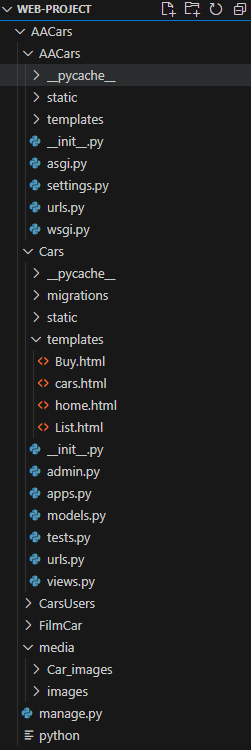
The project name was AACars and the files contained three apps in addition to the main app “AACars” that have two folders static where all the static images such as the logo, the second folder template folder which has the base.html file Moreover the three other folders are “Cars, CarsUsers, FilmCar” and in each app there is template folder that have all the html files required as well as all initial files, the only addition was the forms.py and the urls.py.

Figure The Files Structure

## Project settings used

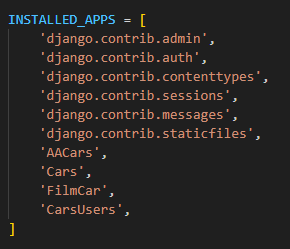


Figure The installed apps in settings.py adding the apps created

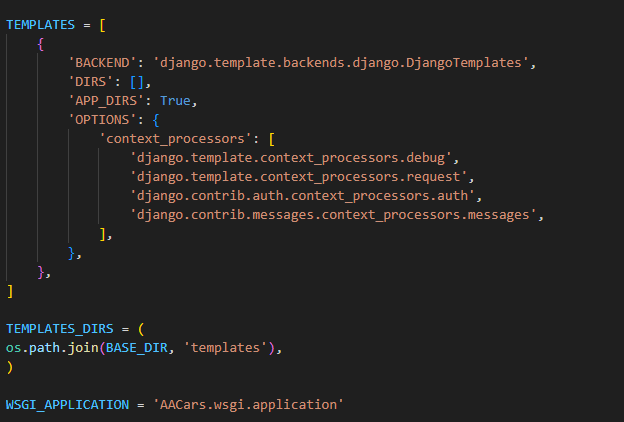


Figure The Templates implemented in the project

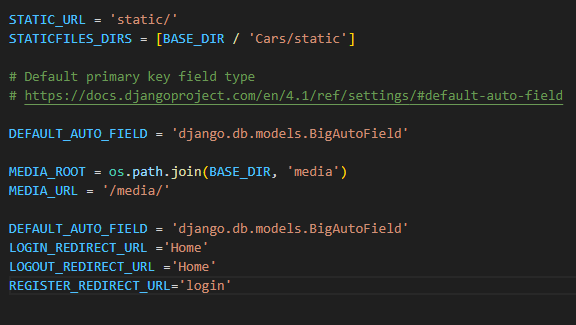


Figure Added codes for the functionalities such as the login/logout redirect and uploading the images in the static

## Django connectivity with MongoDB

As mentioned, and used Djongo serves as a middleware between Django and MongoDB, functioning as an Object Document Mapping (ODM) tool. It facilitates the mapping of Python objects to MongoDB documents, enabling seamless integration between the two frameworks. Moreover, Djongo enforces data integrity by validating and ensuring that only valid and sanitized data is stored in the MongoDB database.

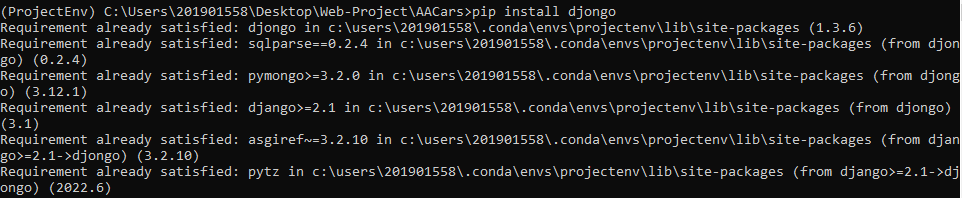


Figure Installing Djongo to connect both Django with the MongoDB

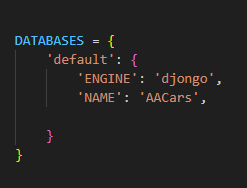


Figure Connecting the database with Djongo as an engine



Figure makemigrations must be applied after any modifications in the project

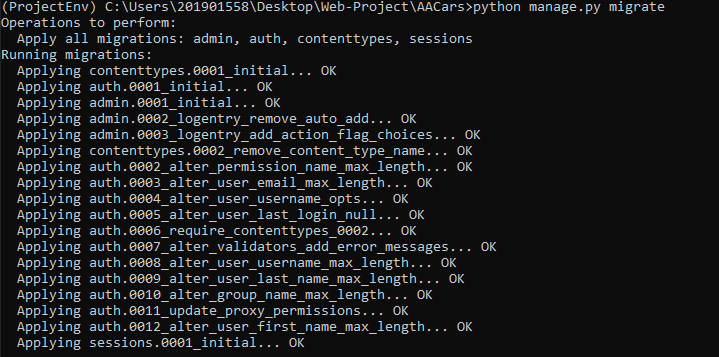


Figure To apply the makemigrations command we must run the migrate command

## Django Template Language

The Django template language is one of the main aspects of the project as it connects between the base.html and the other pages by using its variables and tags, however the variables are calling the objects such as {{Car.Year}} as the car is the model’s name and the year is the attribute of it, on the other hand the tags are used to generate the inherited code on text using for loop for example or to lead the code from base.html {% extends ‘base.html’ %}

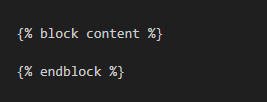


Figure The base.html tag that will load the other html codes overriding the plain text using the Djongo

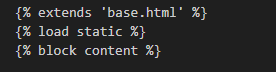


Figure The tags that will inherits the base.html code first and second will load the static folder to take the images from it

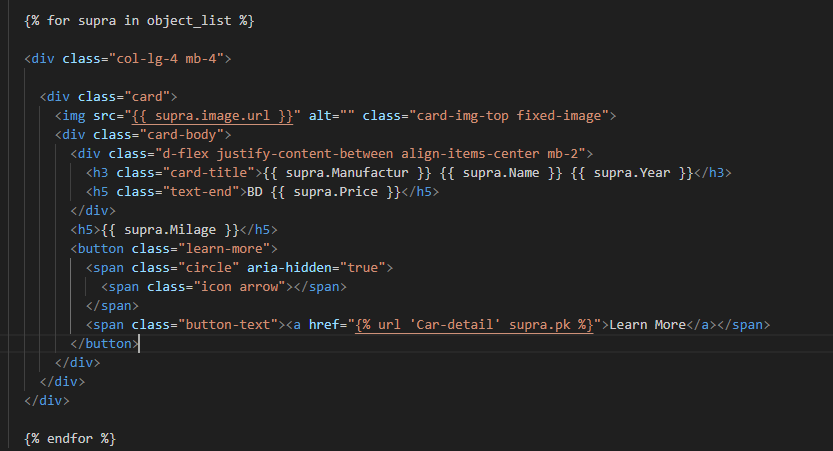


Figure in this tag we basically load the data using the for loop from the object\_list that we have passed as a model to the page from the view

## Model –View-Template

The Model-View-Template (MVT) architectural pattern is used by the Python web framework Django to create online applications. The MVT pattern for software design consists of three fundamental elements:

* Model: The application's logic and data structure are represented by the model. It manages data manipulation operations including searching, storing, updating, and deleting records in addition to defining the database schema.
* View: The view communicates with the user and manages the presentation logic. It takes requests from the user, gets information out of the model, and sends it to the template so that it can be rendered. It also receives input from users and processes form entries.
* Template: The application's user interface and presentation are handled by the template. It specifies how the user sees the data displayed by the application. Templates make use of filters, variables,

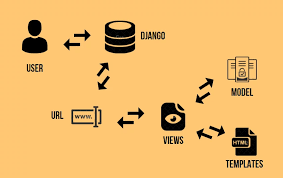


Figure The structure of MVT

An example of it:

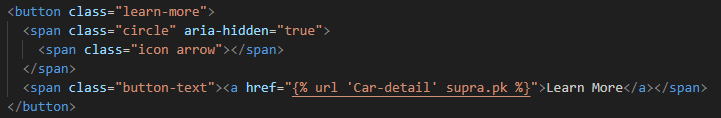


Figure This is a button and inside it there is a link in a tag that ask the urls upon clicking to navigate to the detailed information of the car chosen

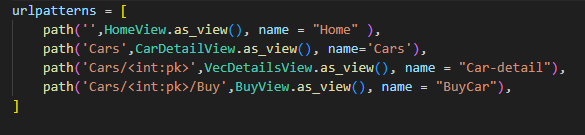


Figure The urls import the view CarDetailView from views that have the car model to link between html pages

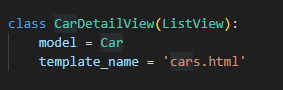


Figure The view class import the model object and pass it to the urls so it could be passed to render the html pages

## Django Admin site



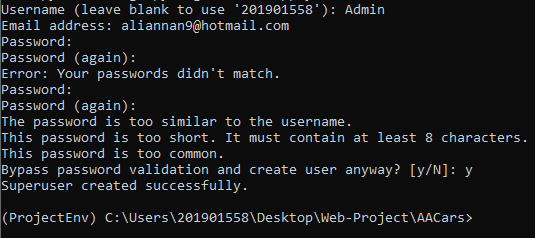


Figure Creating the admin User

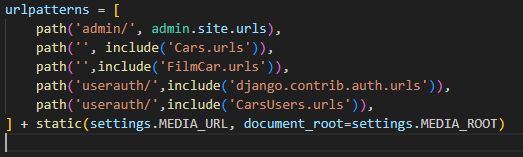


Figure Declaring the urls that the admin access to the admin page

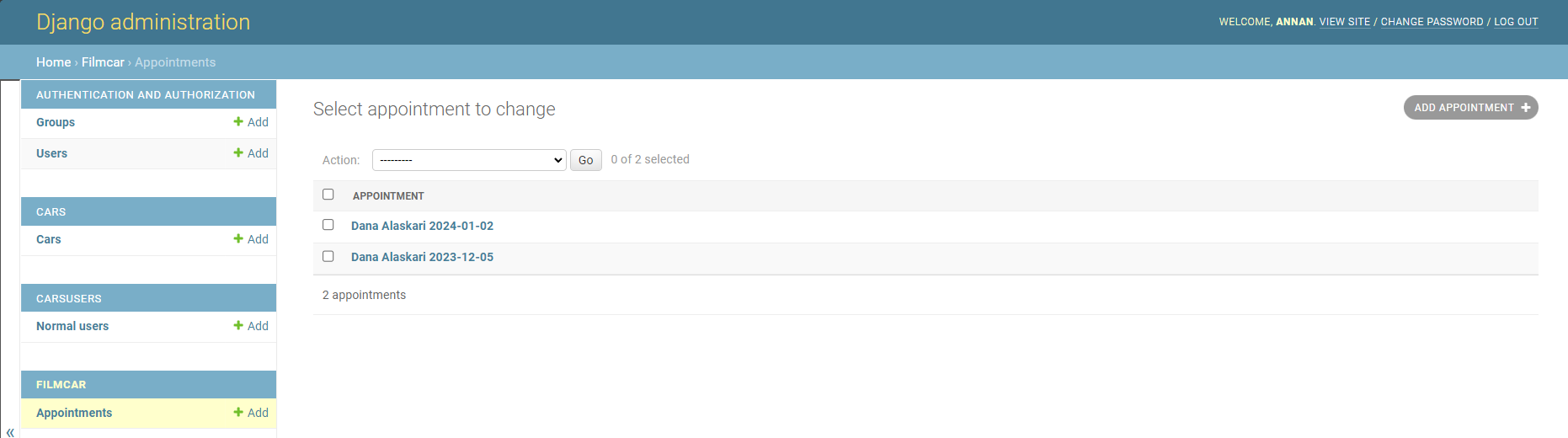


Figure Admin GUI dashboard site containing all model objects and could create edit or delete a document inside each one of them

## Django forms

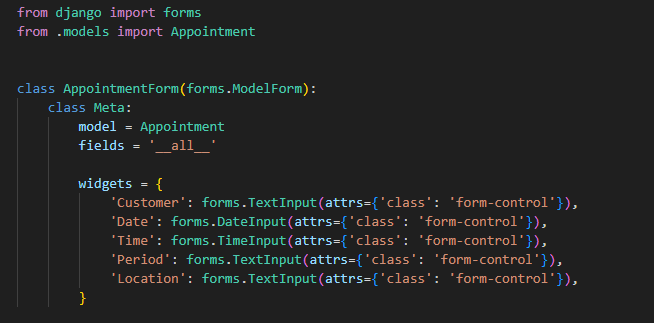


Figure We have created a file named forms.py and created a class named "AppointmentForm" so we could use it in the html pages and pass it through the views

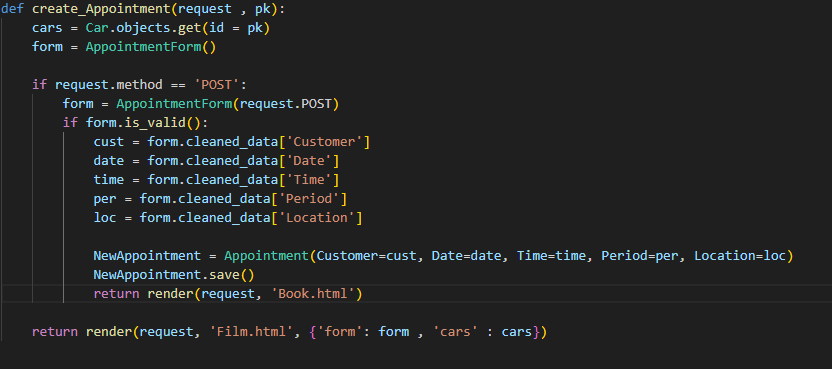


Figure The related view contained a function that render the html page and pass the form inside it along with the model, however in this function we are validating the fields before saving it in the appointment model and save it and after saving it redirect to another page named book.html



Figure This html file has the form and used the widgets declared in the form.py to connect the input with hte fields in the form and using {% csrf\_token %} to secure the page from malicious arracks

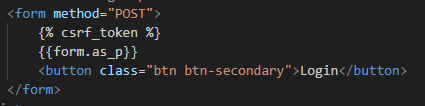


Figure This page is the same as above but the only difference that we do not have widgets the form is generated by using {{forms.as\_p}}

## Incorporation of Bootstrap

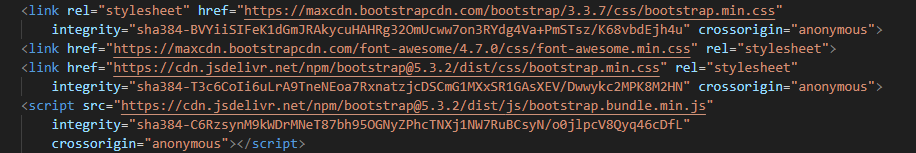


Figure These are the links used to import the bootstrap and connect it with the classes, ids, and the scripts in all html pages to ensure to create a responsive page

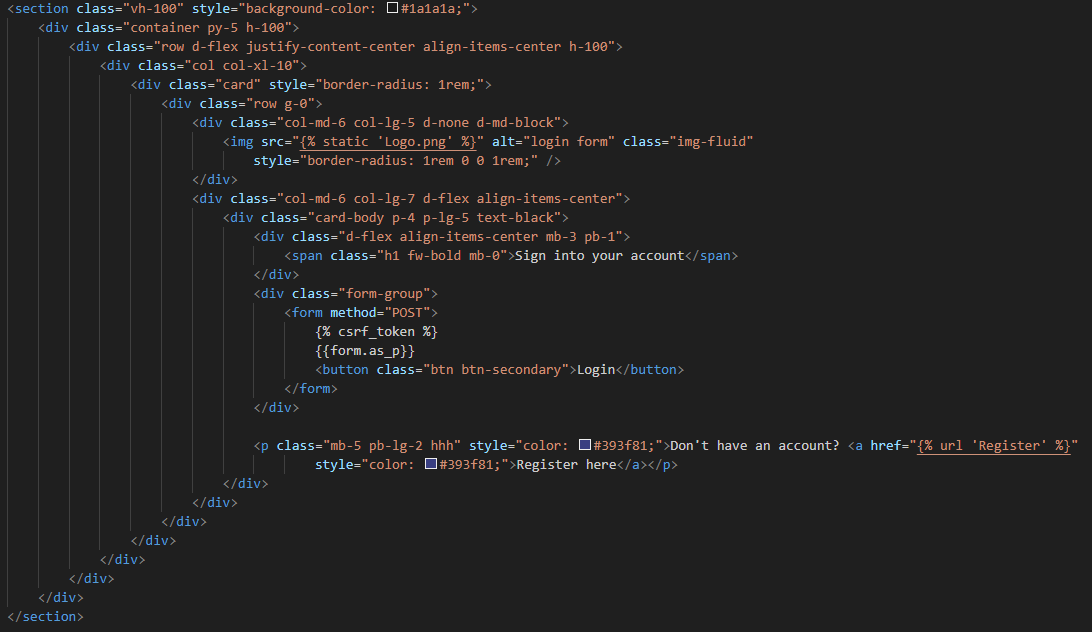


Figure The login page is crated using bootstrap predefined code

# Task4 – Overall GUI and working, Report, GIT hub, Video and Reflection

## Overall Navigational GUI



Figure This is the home page and the user could navigate in any of the pages in the navbar or to see the cars from the button in the middle of the page in addition to having a welcome sentence with his/her name along with the logout button

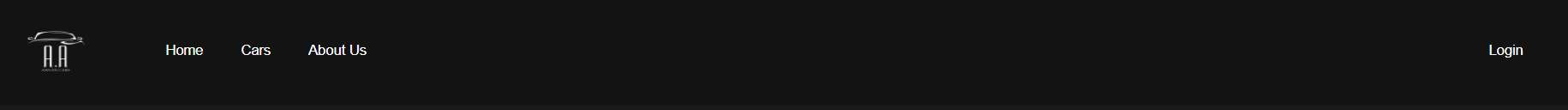


Figure If the user was not logged a login button will popup

## Working of Web Application meeting all Functionalities

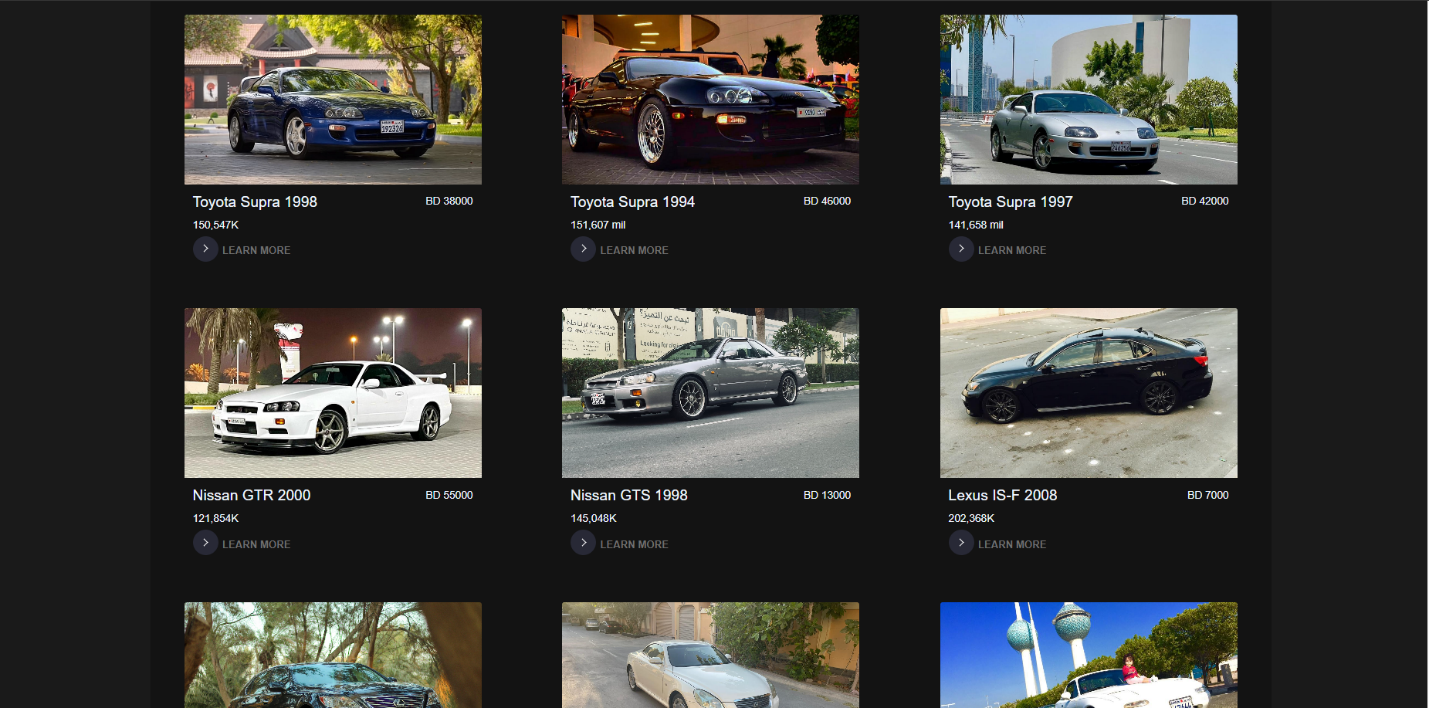


Figure Demonstrate the list of all available cars that we have in the database

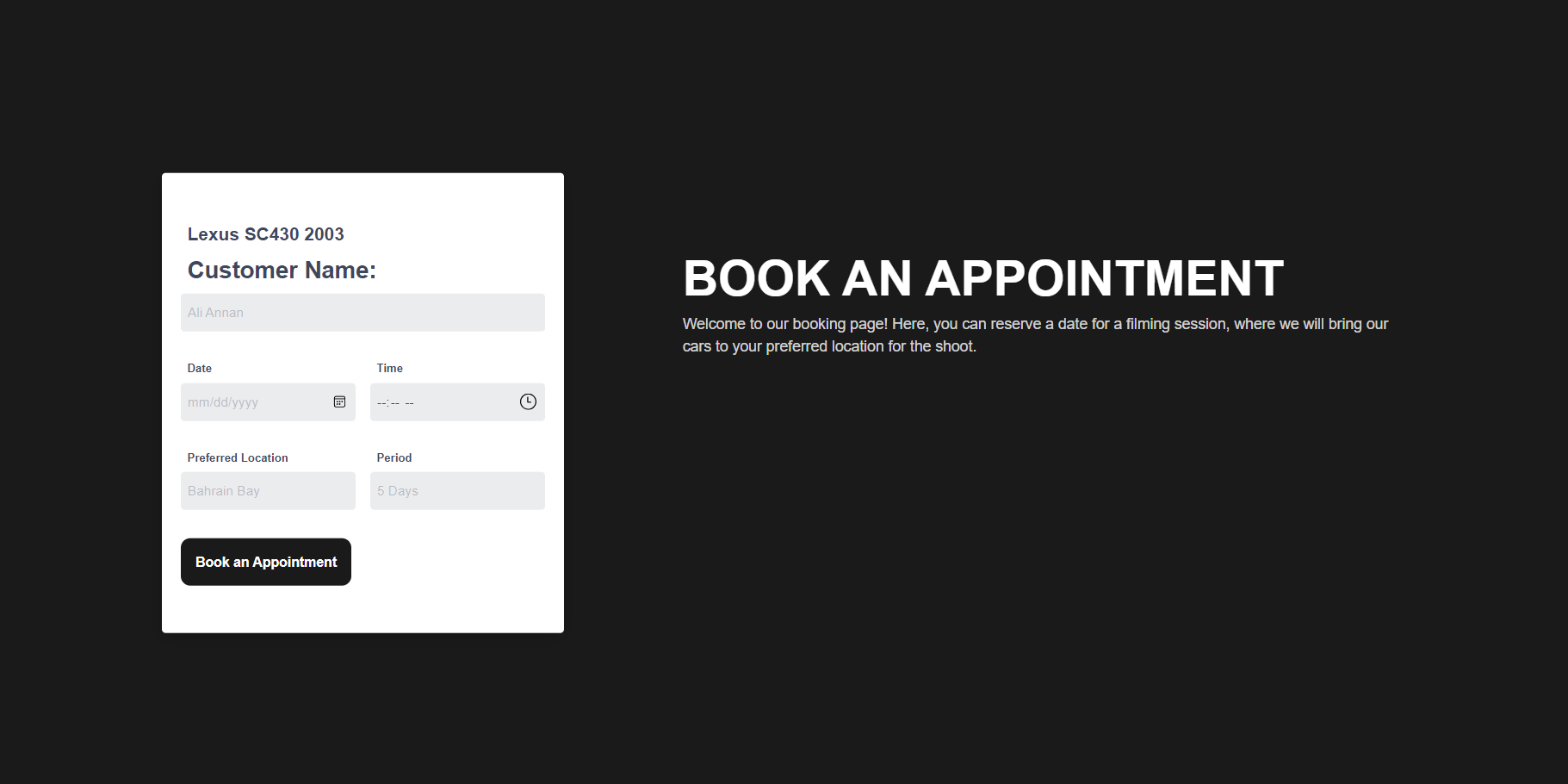


Figure Booking an appointment to film one of the cars be filling the required information

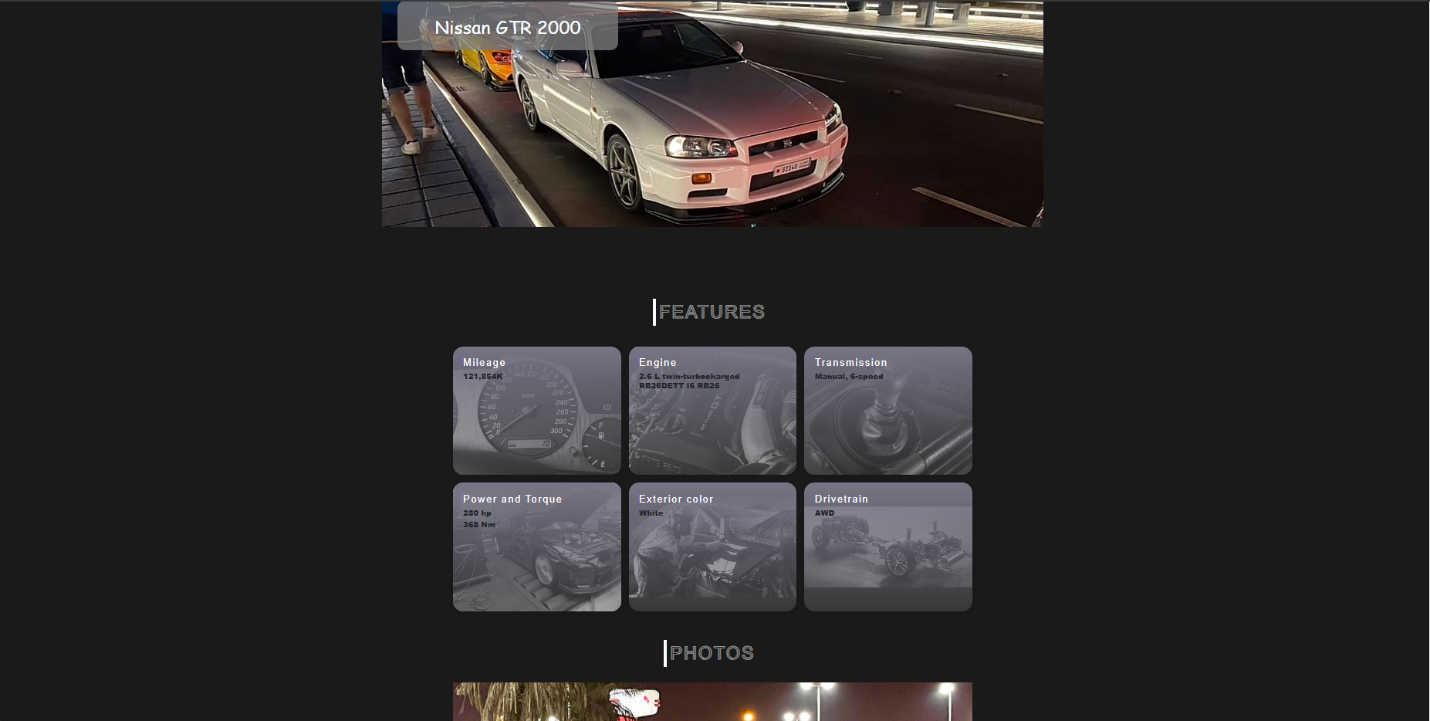


Figure Explaining the details information of the selected car

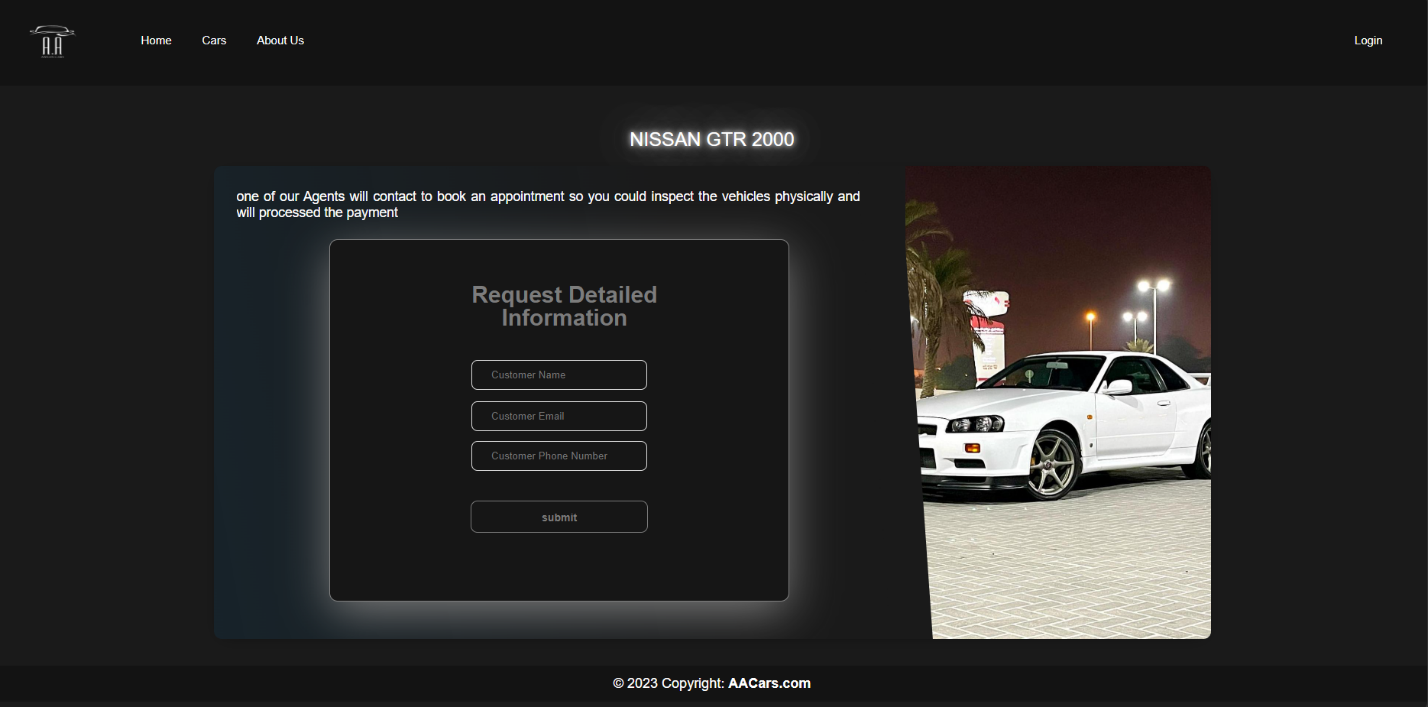


Figure Inserting the user information to buy the sleeted car

## Reflection

This experience was truly unique, combining both excitement and challenges throughout the project. Initially, before selecting database as my major, I had planned to take this course and was eagerly looking forward to it. As soon as the project was released, I enthusiastically began working on it, drawing upon my web development knowledge from a previous course. However, to my surprise, I discovered that most of my previous work was largely irrelevant, presenting my first major challenge. I had to figure out how to integrate my code with Bootstrap since I hadn't implemented it before. Despite feeling a bit frustrated, I started from scratch once again. However, I encountered another obstacle when my laptop started generating numerous errors, such as database and connection issues. To overcome this, I opted to work on the university's computers, which proved to be more reliable than my own laptop. However, this decision presented a significant challenge as it required me to be physically present at the university every day. I even worked on public holidays and weekends, from 9:00 AM to 11:00 PM daily. Additionally, a considerable amount of time was devoted to research. Many of the errors we encountered were new to us and hadn't been covered in class. One common issue we faced was the "no reverse match" error, which we had to understand the concept of MVT (Model View Template) in order to resolve. Initially, the main focus was on the functionality of the project. However, during the coding phase, implementing Bootstrap with the CSS code I had created or found online became a time-consuming task. Working together as a group proved to be beneficial, as we were able to research and resolve errors more efficiently. Nevertheless, the frontend development phase required a significant amount of time and effort, as it was challenging to find the specific elements I needed. I had to utilize advanced coding techniques to achieve the desired results. As with any other project, the challenges did not end there. When it came time to record the video, we realized that we did not have a microphone. We managed to acquire a headset, but unfortunately, when we connected it, the computer failed to recognize the headset's microphone. To overcome this obstacle, we purchased an adapter that allowed us to connect the headset to the PC. In conclusion, despite the challenges faced, this was an exceptional learning experience, and I truly enjoyed working on this project.

## The GitHub link

<https://github.com/AliAnnan9/Web-Project.git>

## The Video link