Project Name: CarXplore

Technologies Used: Angular 7, TypeScript, Material Design, Spring MVC, Maven, Machine Learning, Python, SQL, Spring ORM, Hibernate.

Database: MySQL 8.0

Description:

CarXplore Application has 7 modules.

1. Customer
2. Admin
3. Registration
4. Login
5. Cart Module
6. Payment Module
7. Store Locator Module
8. Add Cars Module

The Application is a SPA (Single Page application) built using Angular 7 in the front End and Spring and hibernate at the backend.

The app is used to view all the latest cars and upcoming cars of all Car brands.

The users can View all the cars, Locate the stores around them. This is implemented using K-Nearest Neighbor Machine Learning Algorithm.

The users can also add the cars to the cart and checkout and pay online. In the payment module we have Captcha, to avoid the bots from booking the cars.

The new upcoming cars can be pre-booked by the user.

The Registration and Authentication is done via AJAX calls from the angular application to the backend via HTTPClient.

The Ajax Calls to the server ensures quick communication between frontend and backend without

Refreshing the Entire page when the form is submitted.

The Response from the server is processed by using Observables.

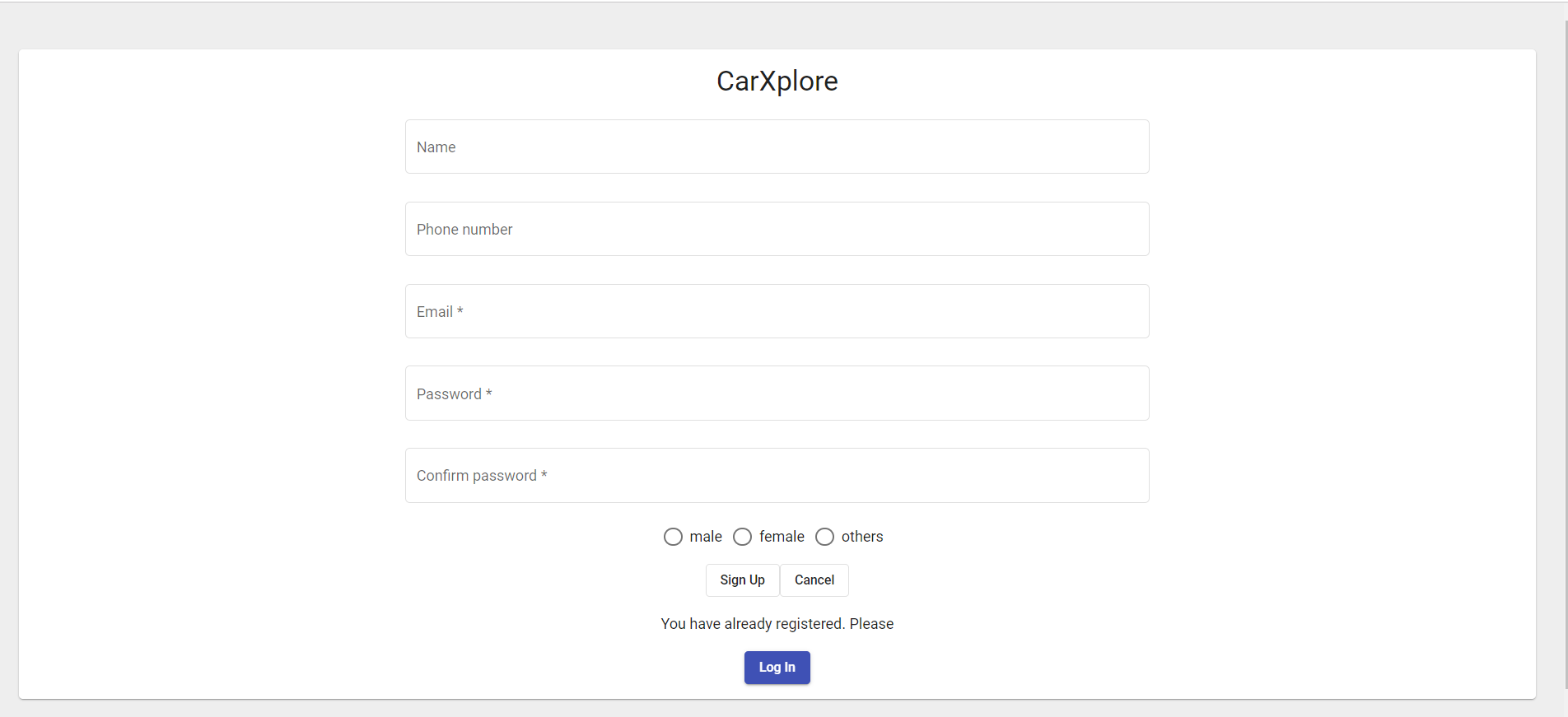
The Registration module has two Roles.

1. User
2. Admin

The user can view all cars, book and locate store for a specific car brand.

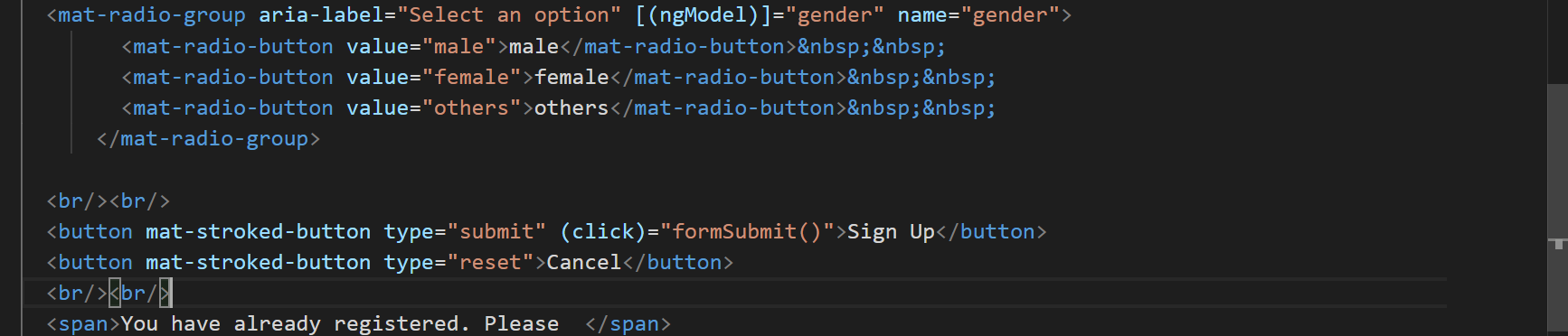
The Admin can view all cars, Book and locate the stores for a specific car and can also add new cars

The Below Screenshot shows the Registration Page of CarXplore



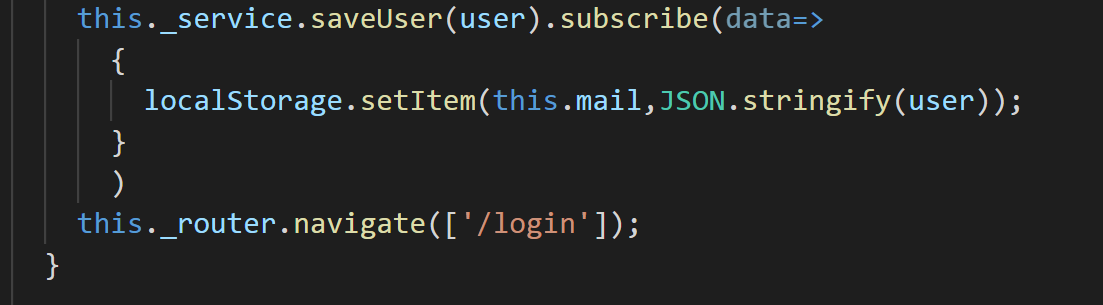
The HTML code for the Registration page:



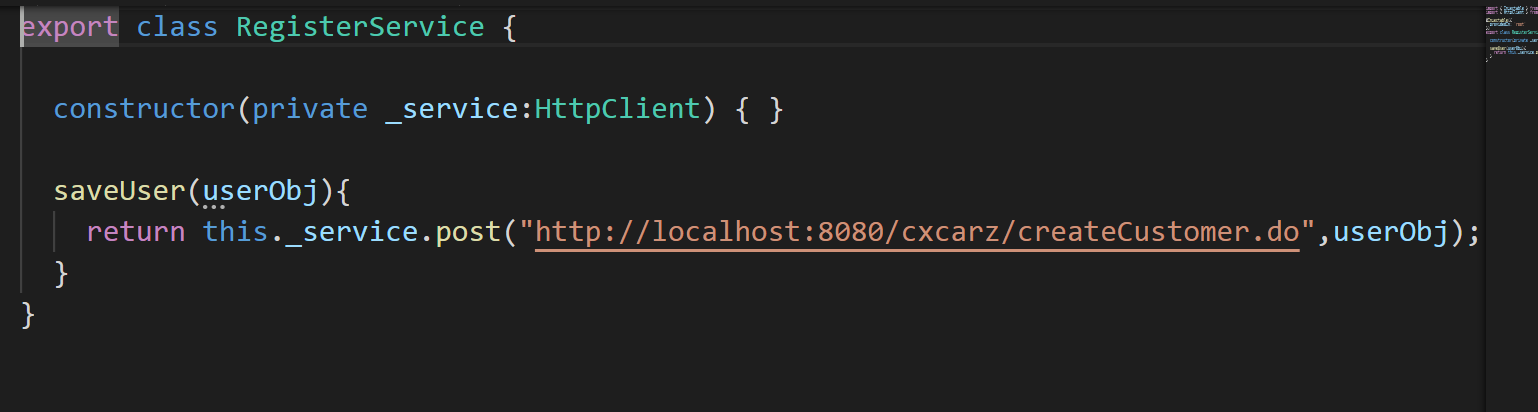


The following Code snippet shows Login Implementation using TypeScript:

1. In Component

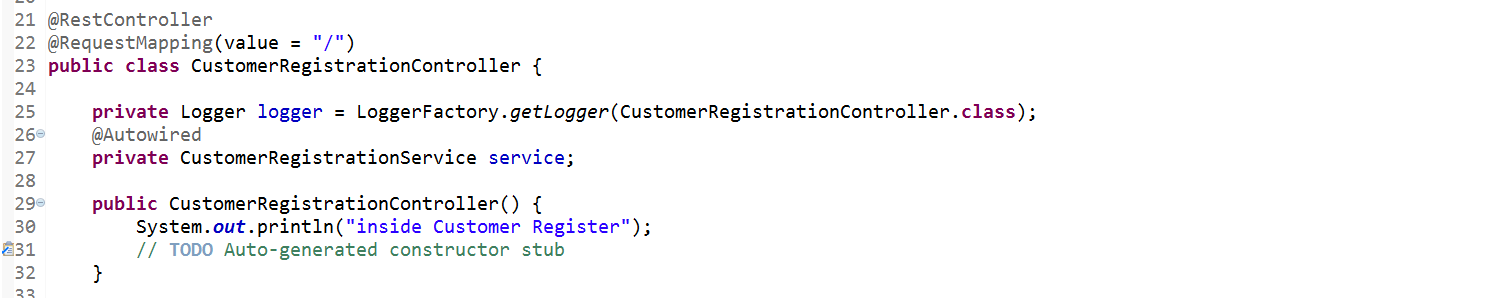


1. In Service



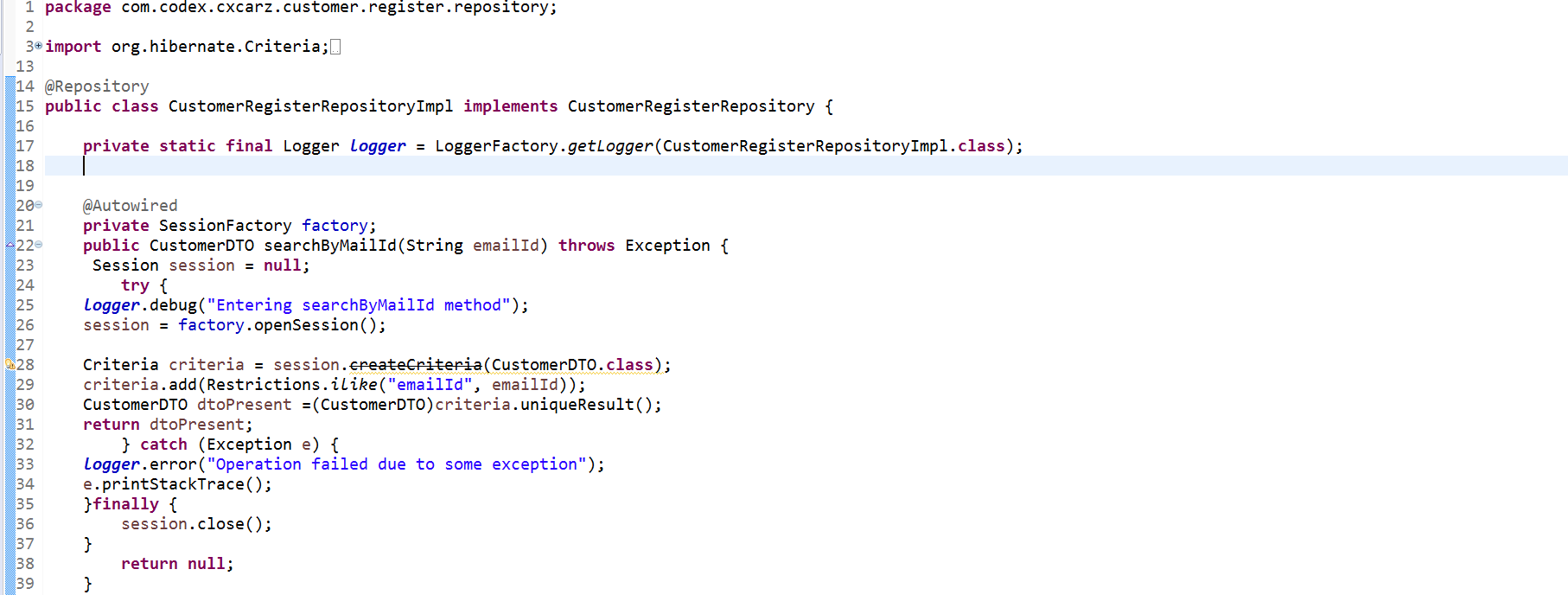
The below snippet shows the implementation of registration in backend to save the data using Spring ORM.

1. In Controller

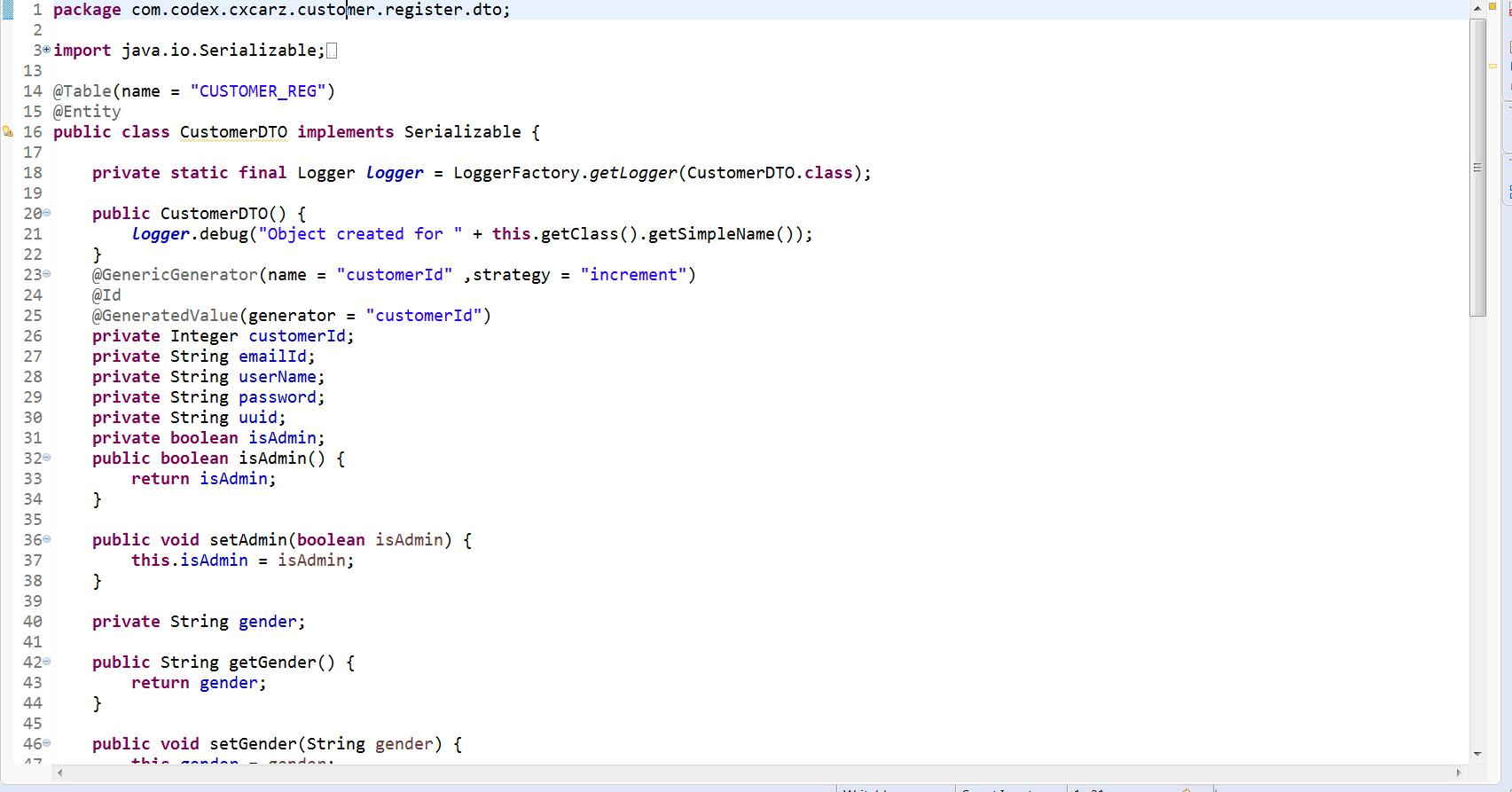


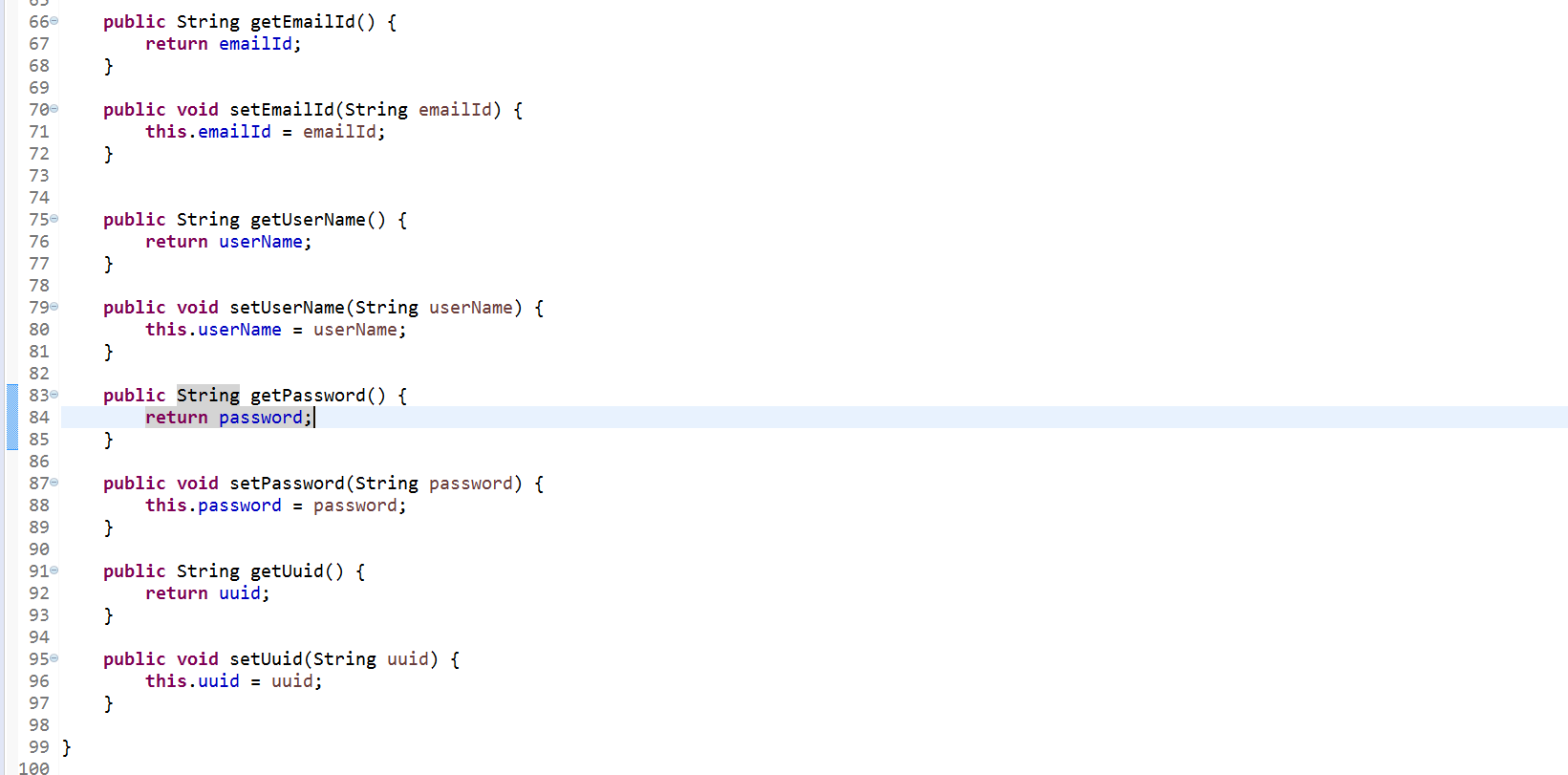


1. In Repository

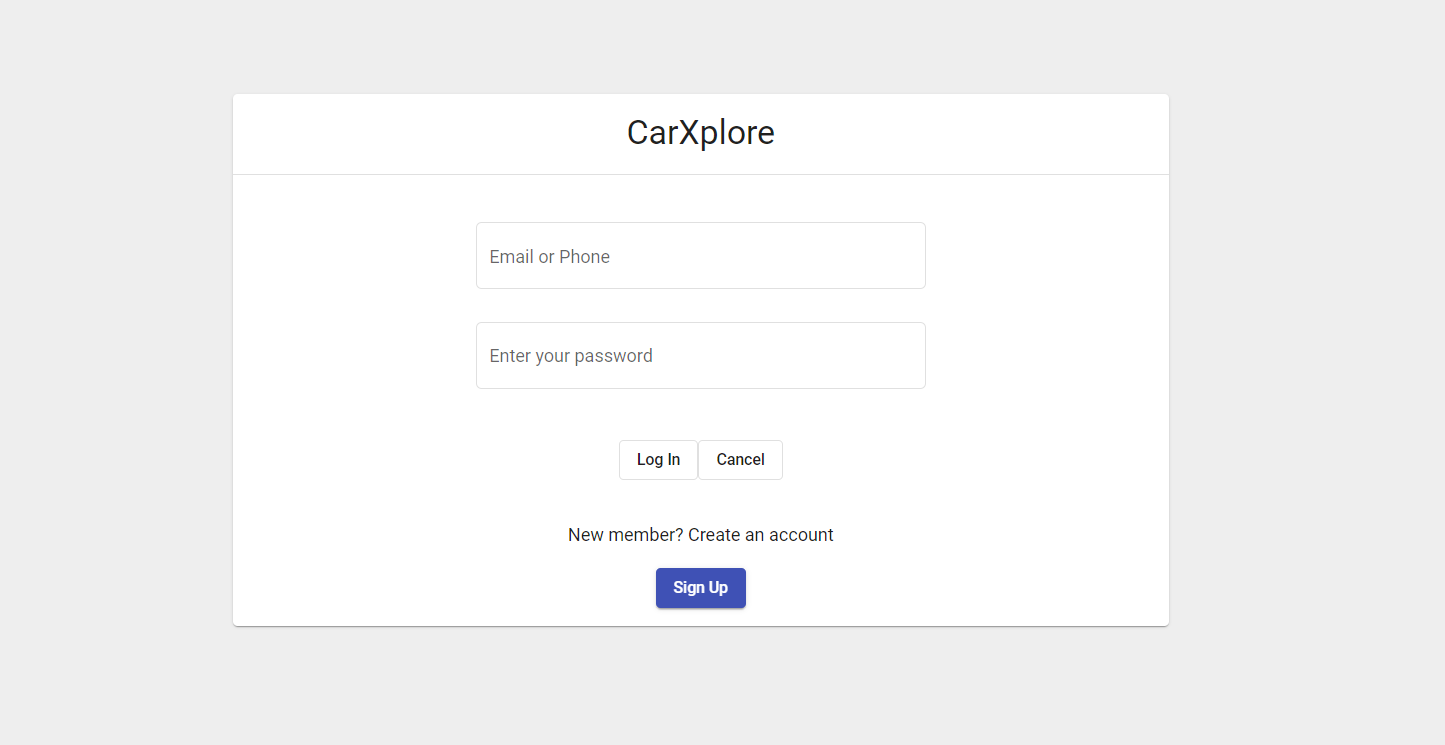








The Following Screenshot shows the Login Page

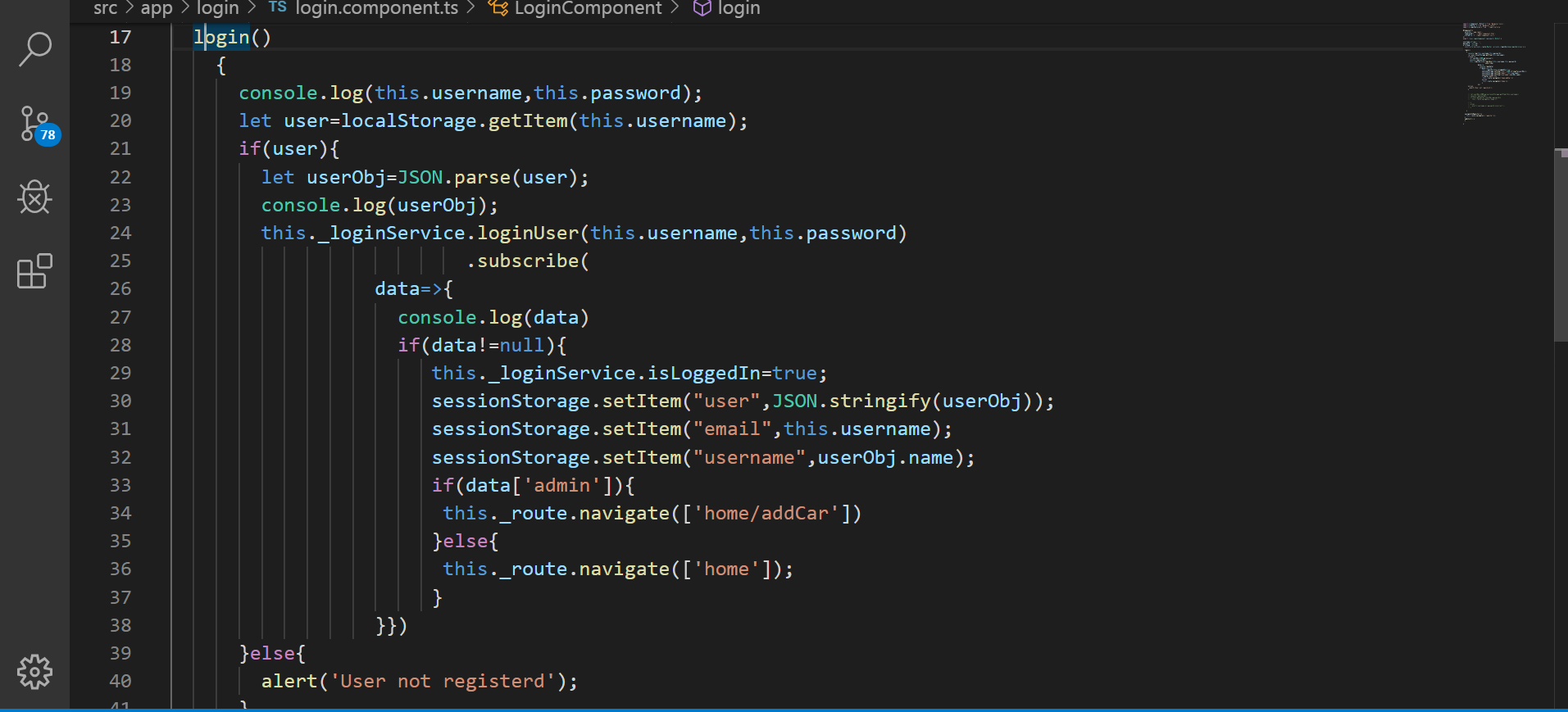


The HTML for the Login page:



The following code snippet shows the implementation of Login in TypeScript

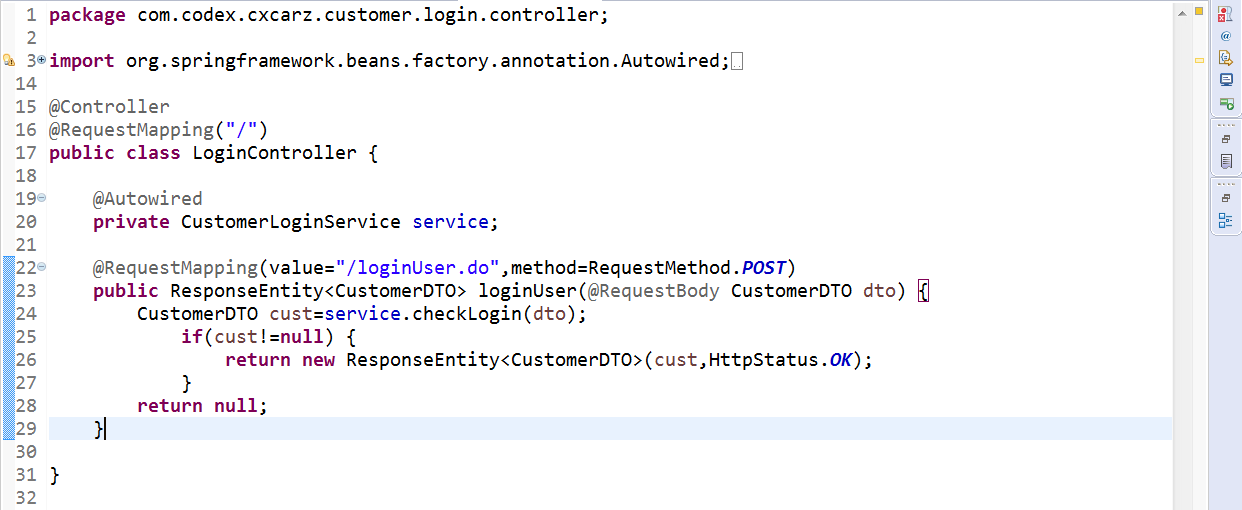
1. In Component



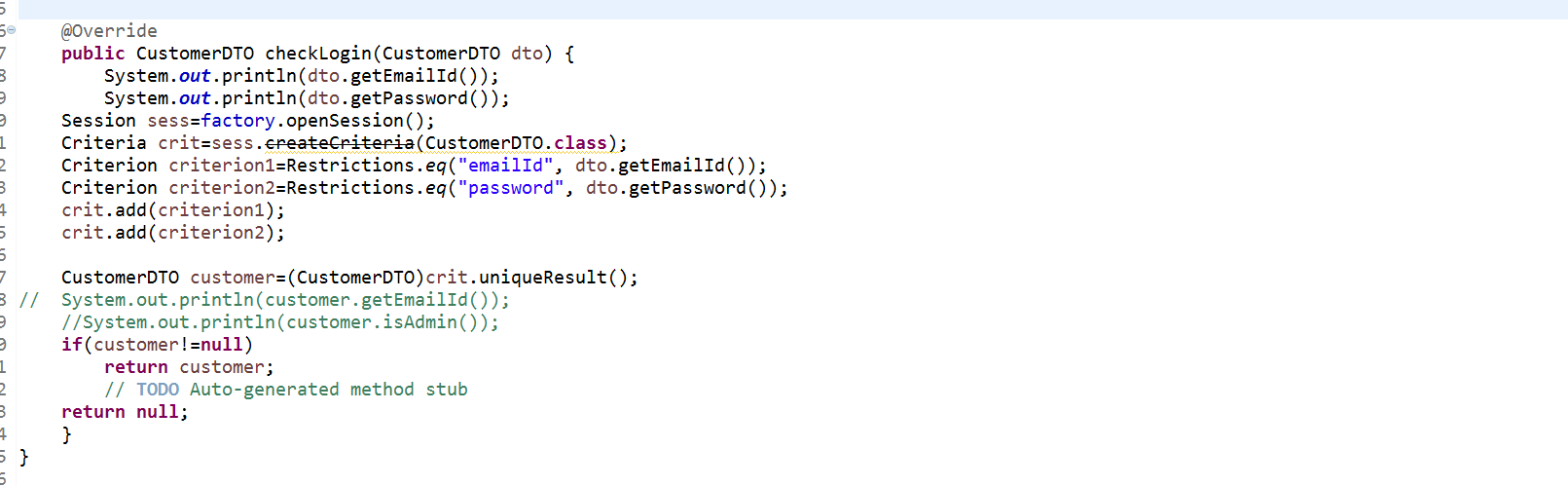
1. In service



The following Snippet shows the implementation of Login in the backend using Spring ORM.



In Repository

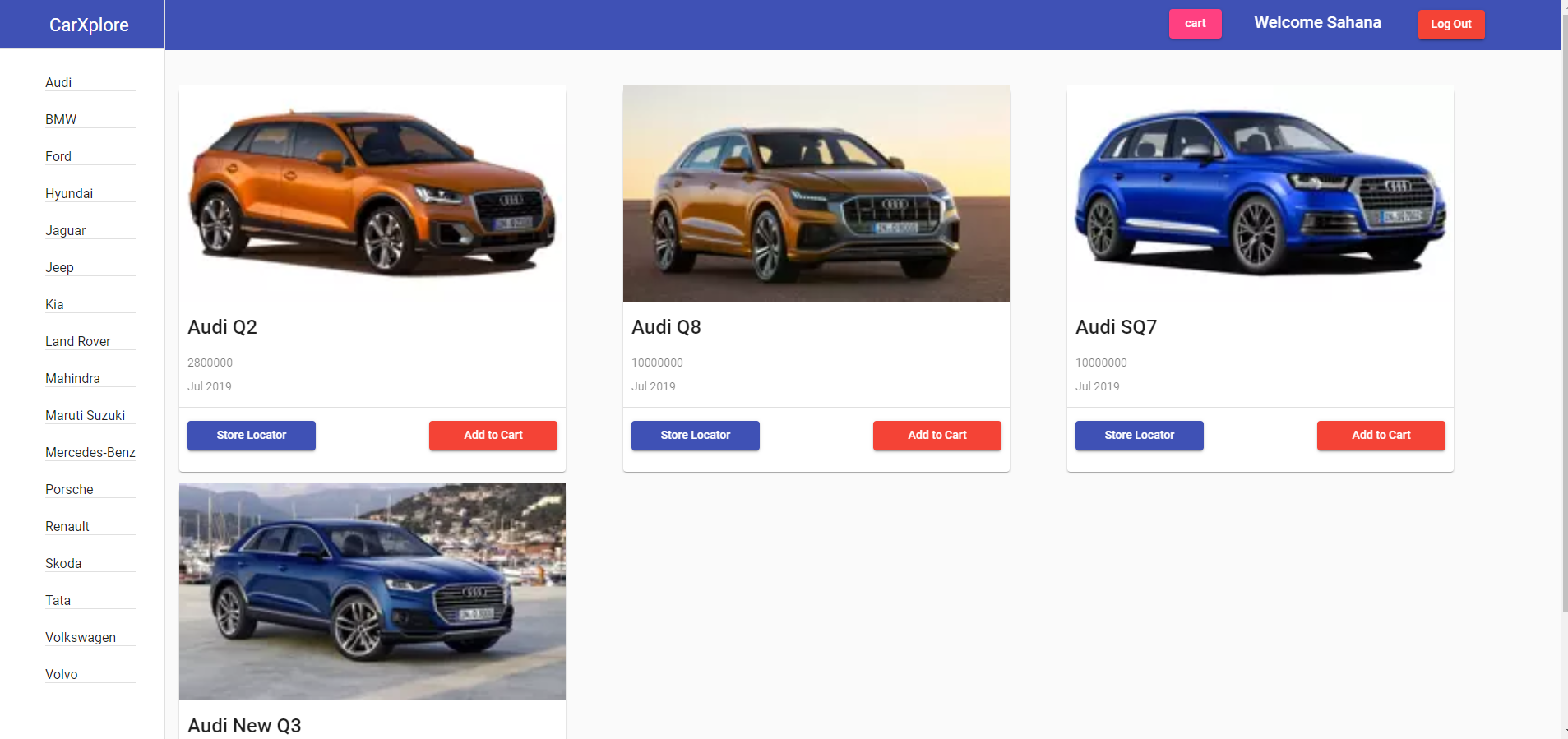


**Dashboard**

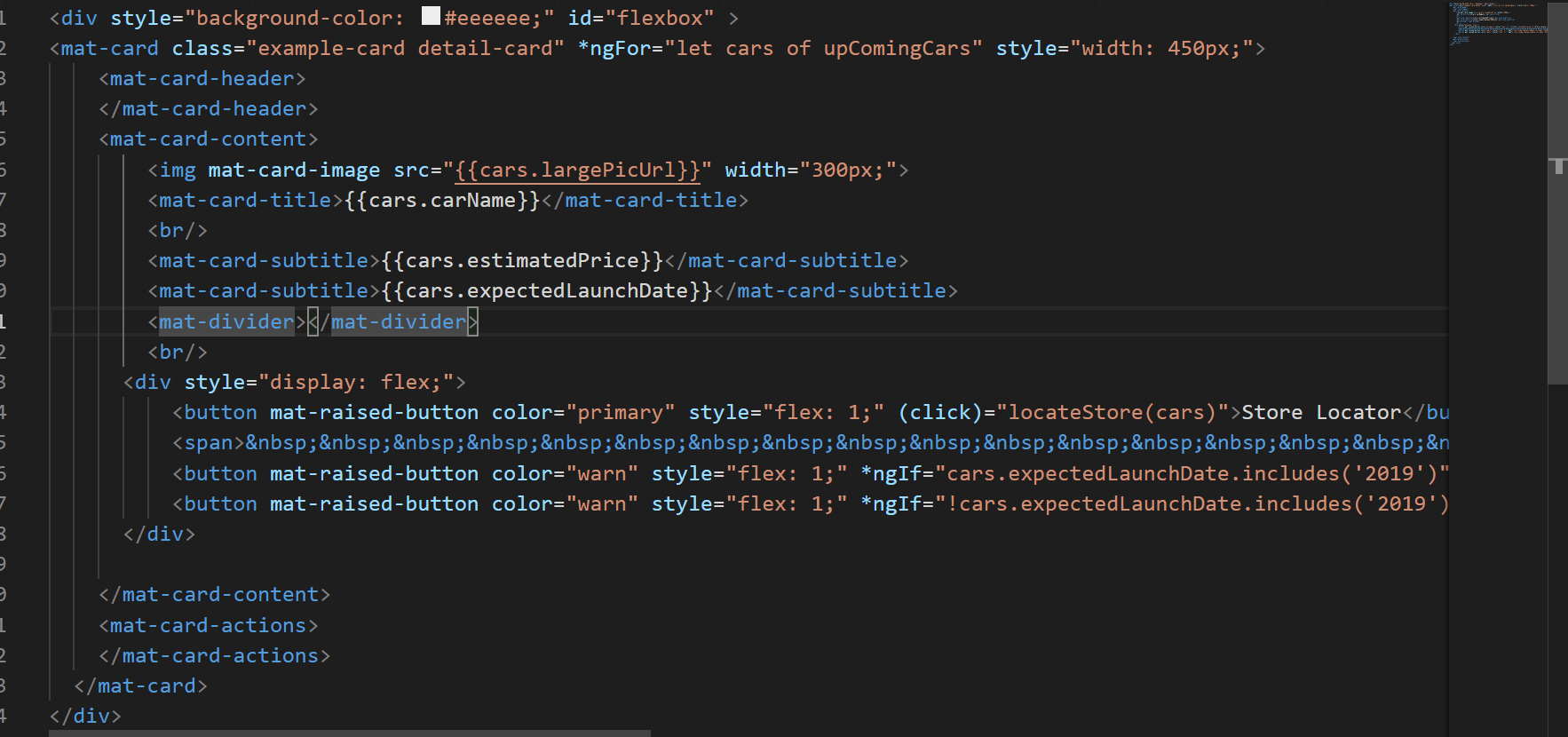
The user, after successful Authentication, is redirected to the Dashboard Page.

The Dashboard page consist of a Side Bar which shows all the car brands available, a Navigation bar for viewing the cart, checkout and for payment.

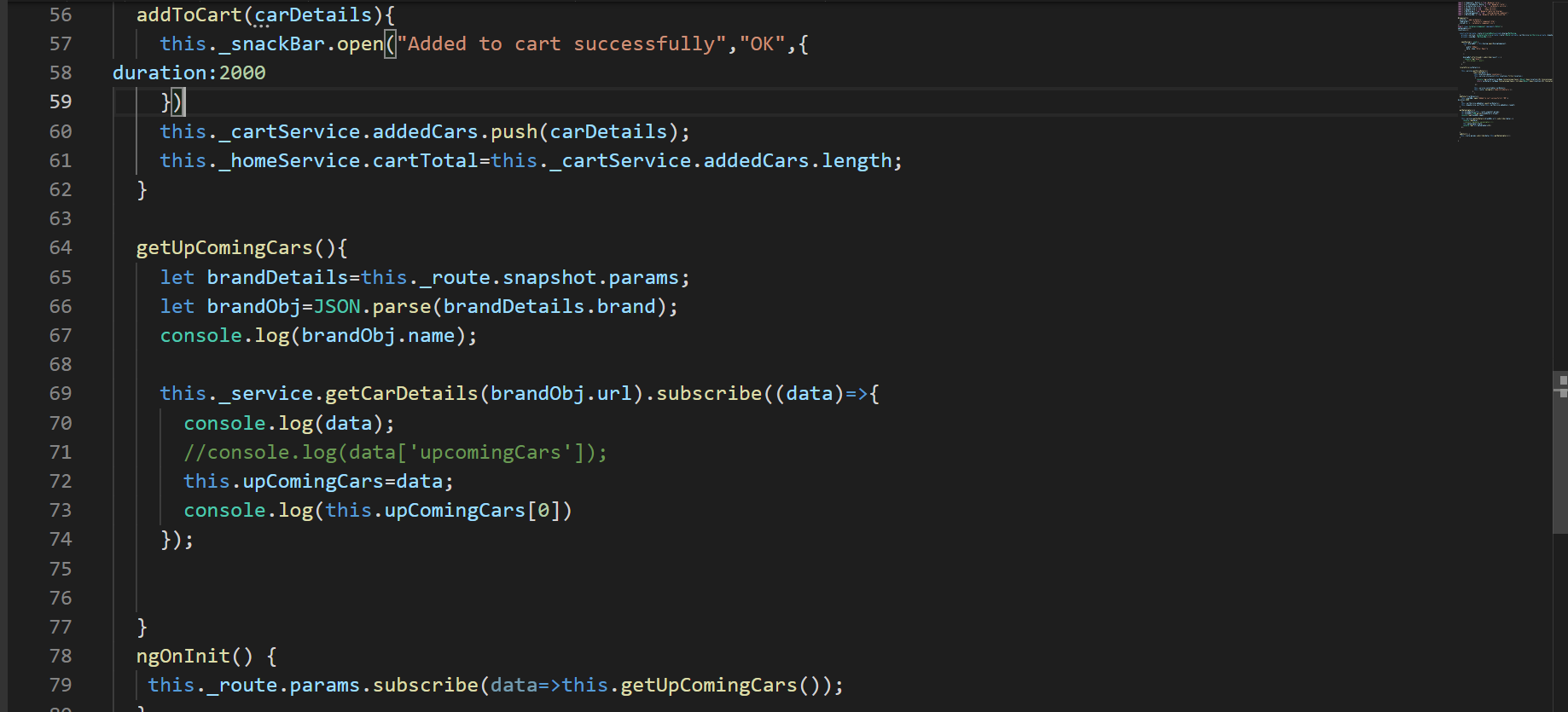
The user can view all the available cars and upcoming cars in the Dashboard page.



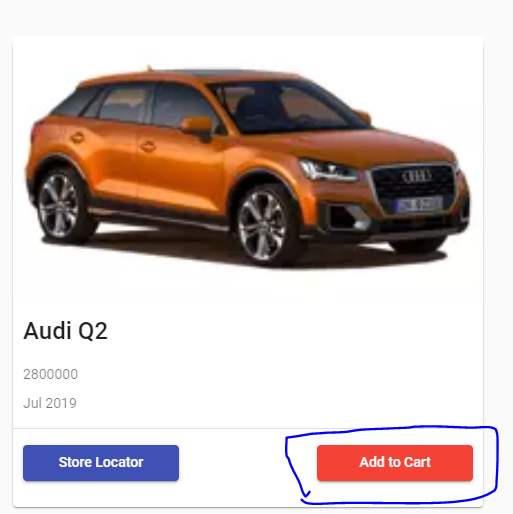
The following snippet shows the HTML code for Dash Board Page.



The TypeScript Code for the Dash Board:

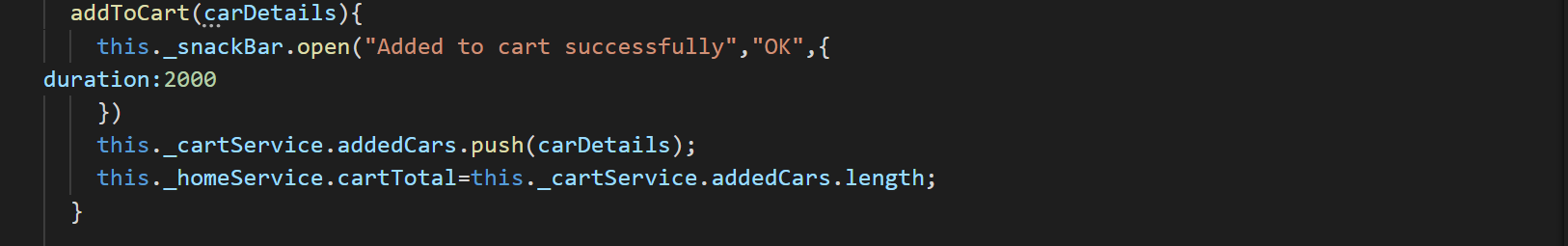


The Available Cards can be added to cart by using Add to cart button.

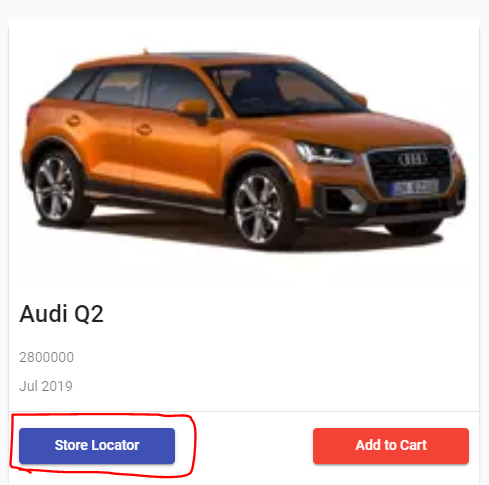


The Add to Cart button adds the selected car to the cart, where the user can check out.

The snippet for adding the items to the cart:



The user can locate the nearest store for a car



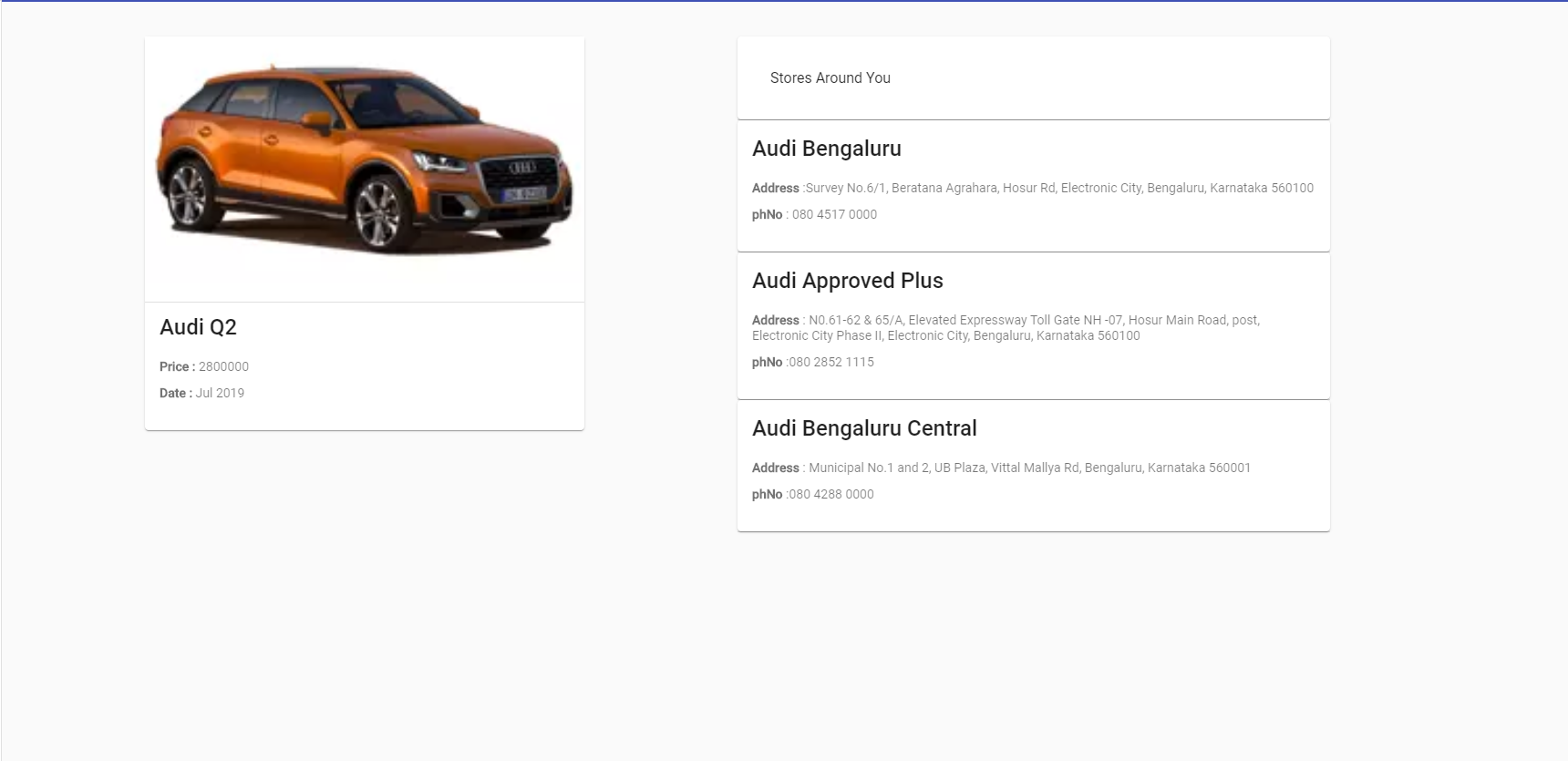
**The Store Locator:**

Store locator uses the popular Machine Learning algorithm called as K-Nearest Neighbor to find the Nearest car showroom for the selected Car.

This algorithm finds all the nearest showrooms and provides details about the showroom.

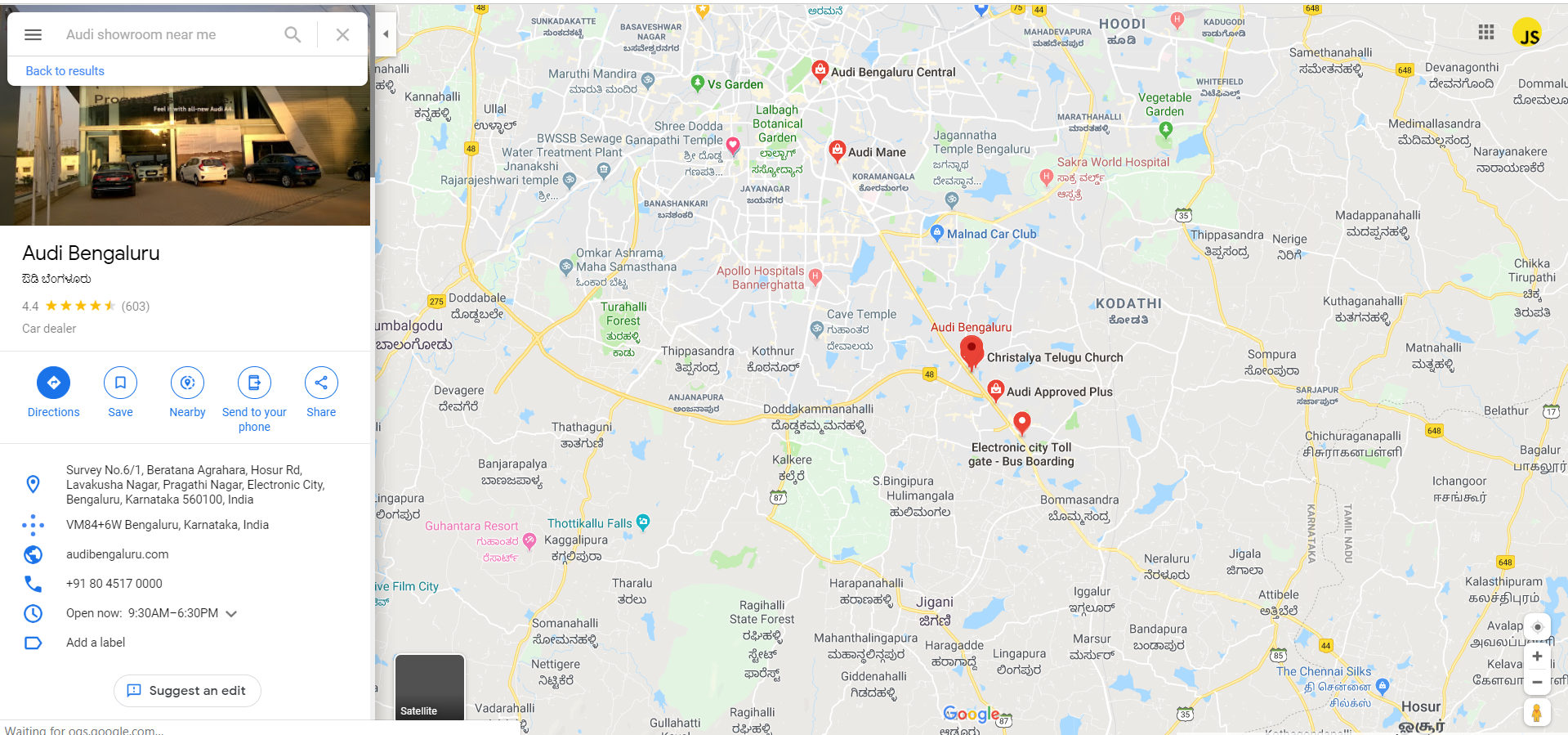
The user can select a store location, and the Google maps will be opened to show the location of the showroom for the user.

The screenshot shows the Screenshot of the store Locator for Audi Q2.

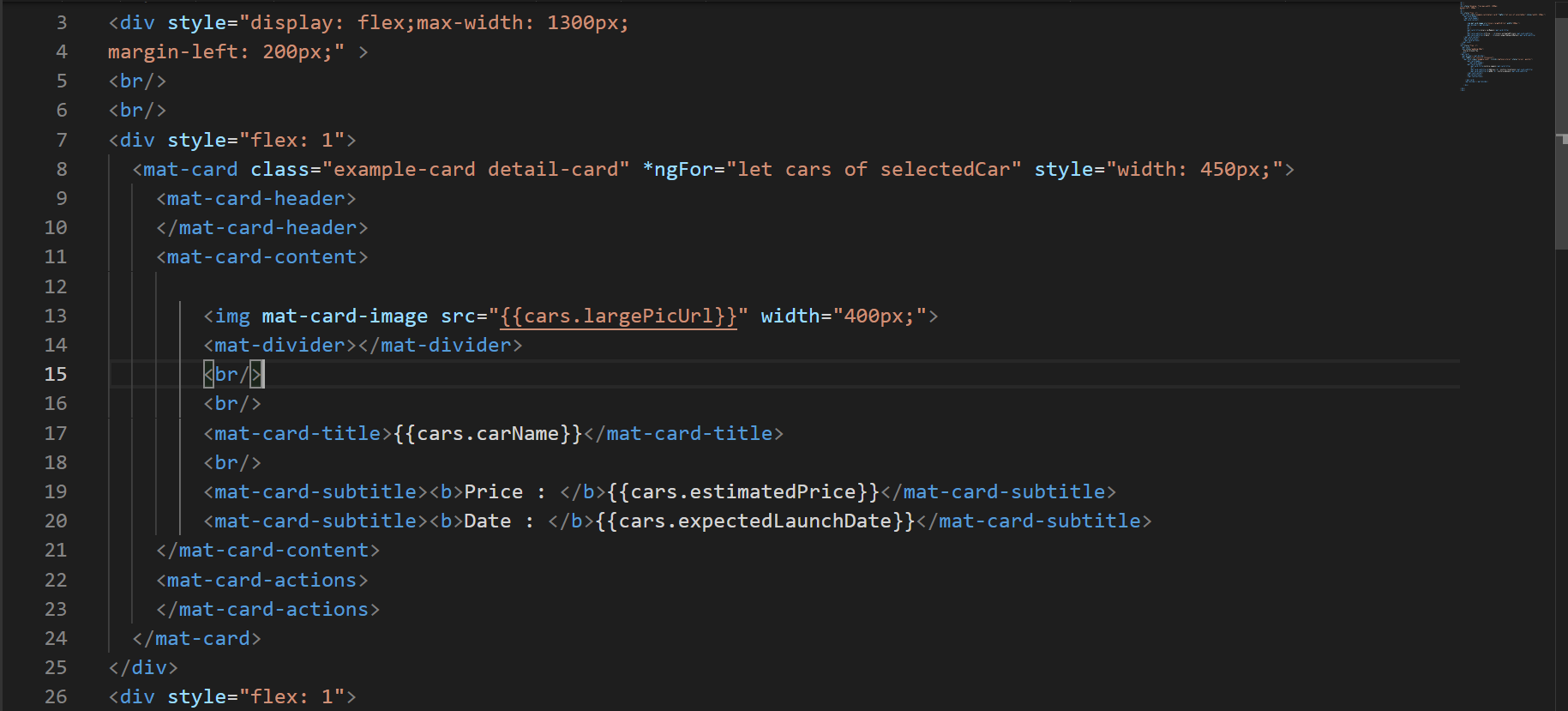


Upon clicking a store in the list, the user can view the Google map of the Store.

The below Screenshot shows the google map of a store selected by user (Ex: Audi Q2)



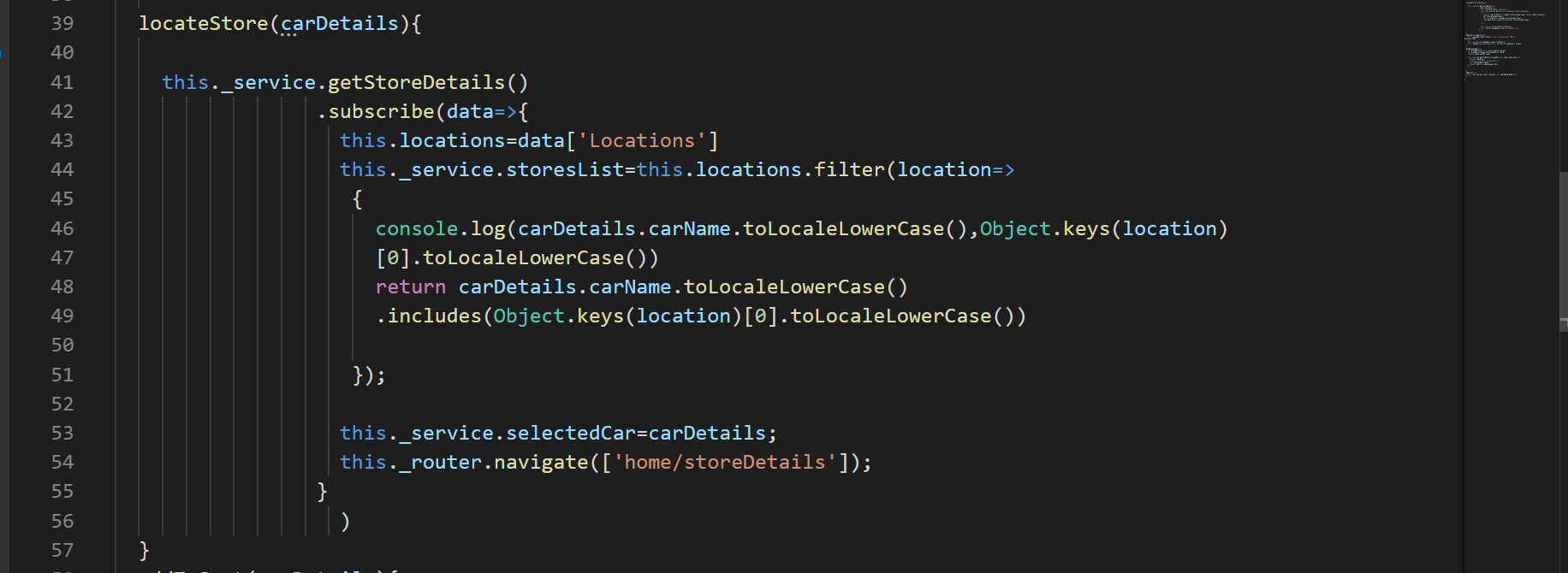
The HTML snippet for the Store Locator:

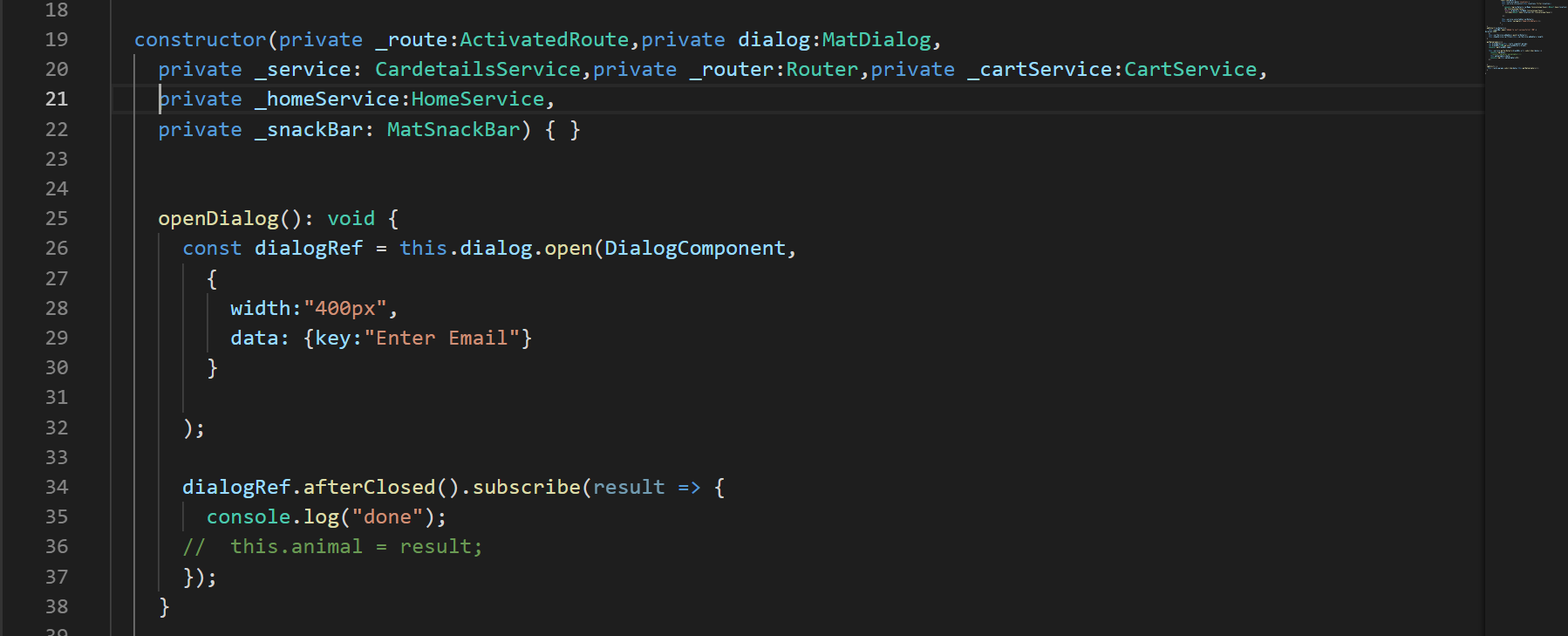




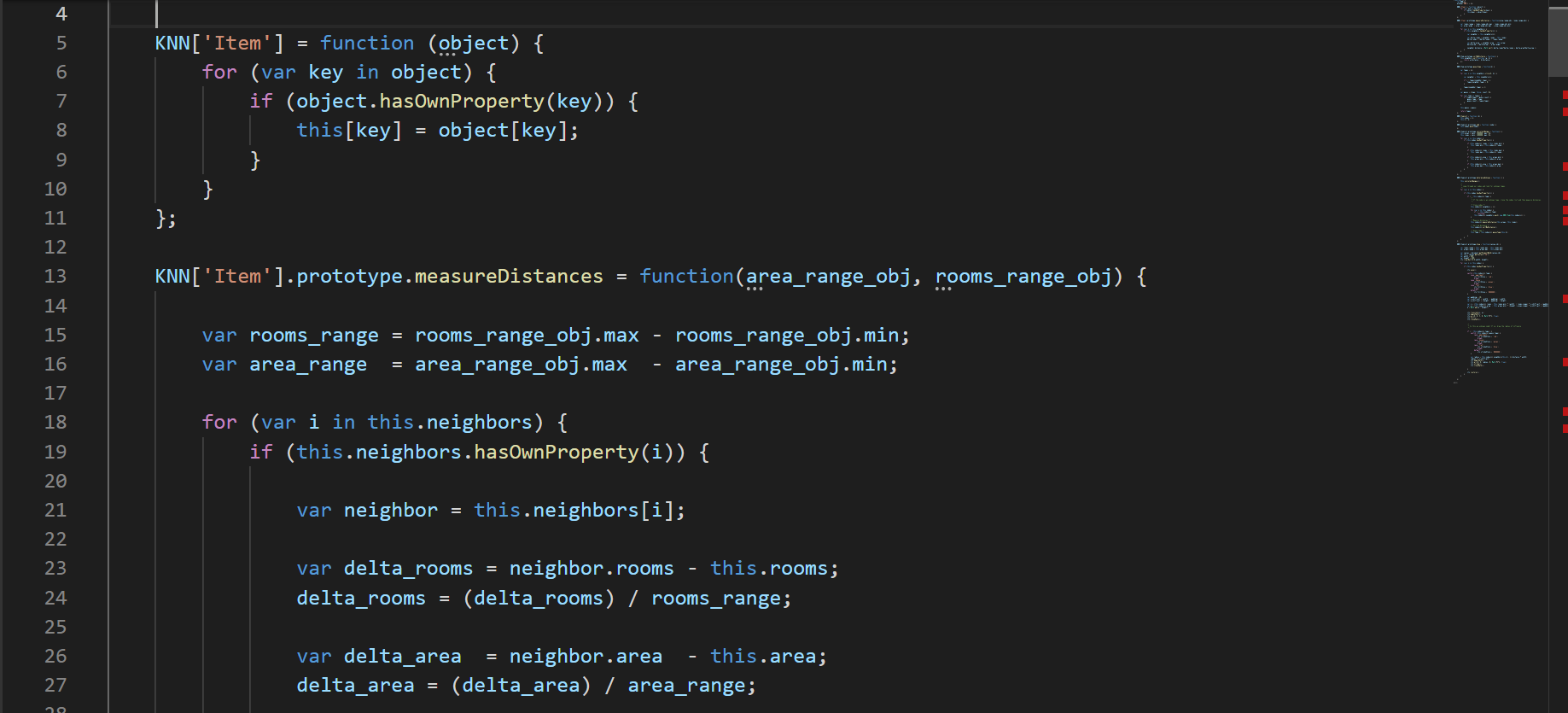
The TypeScript Implementation of the Store locator:

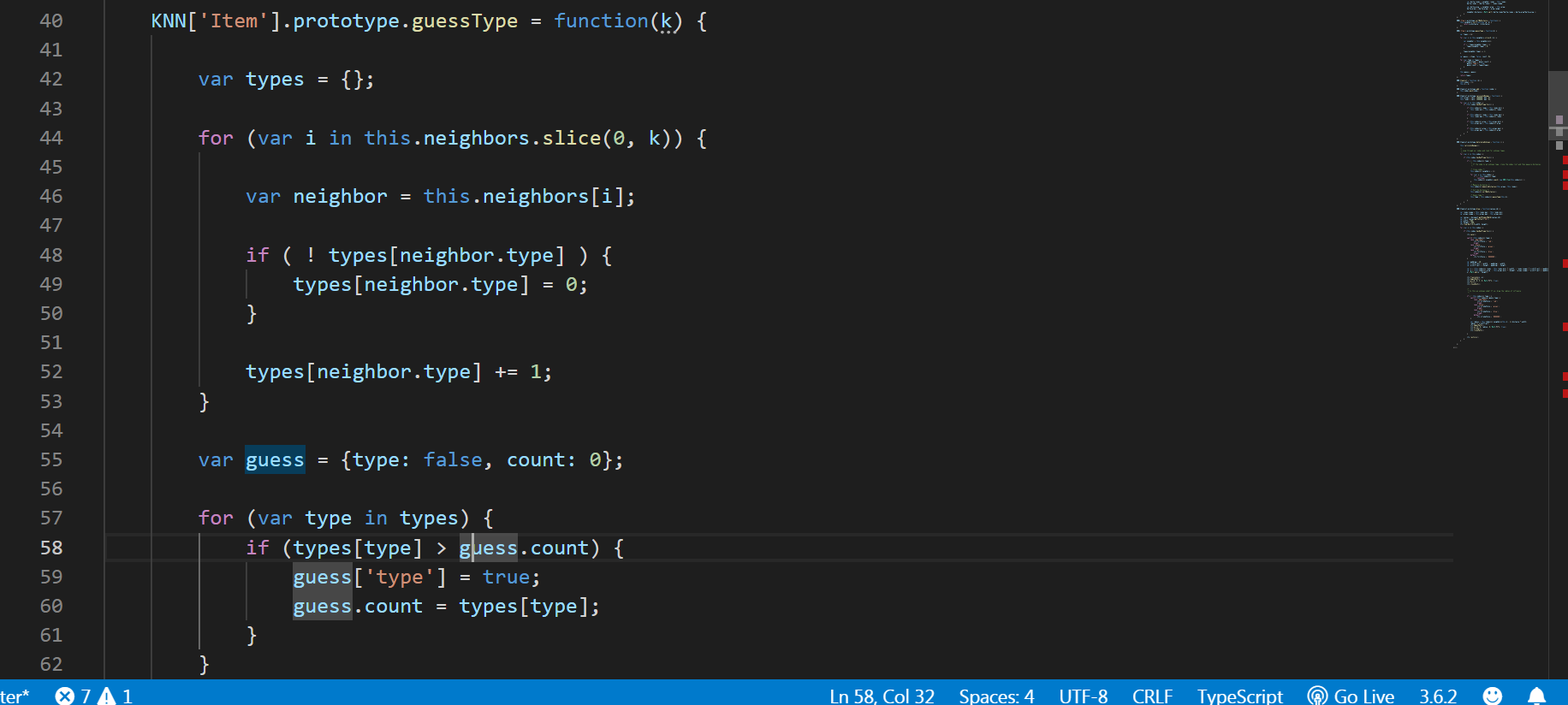


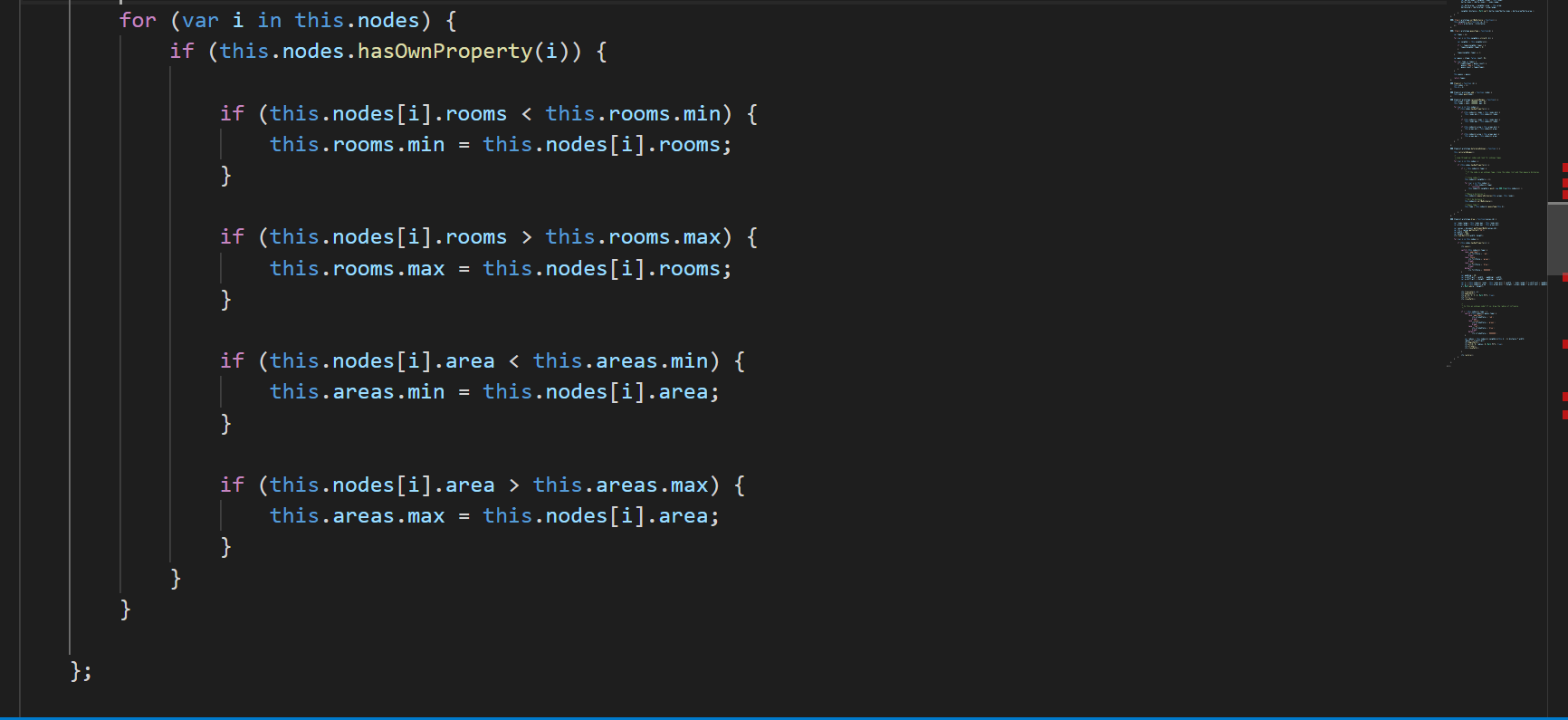


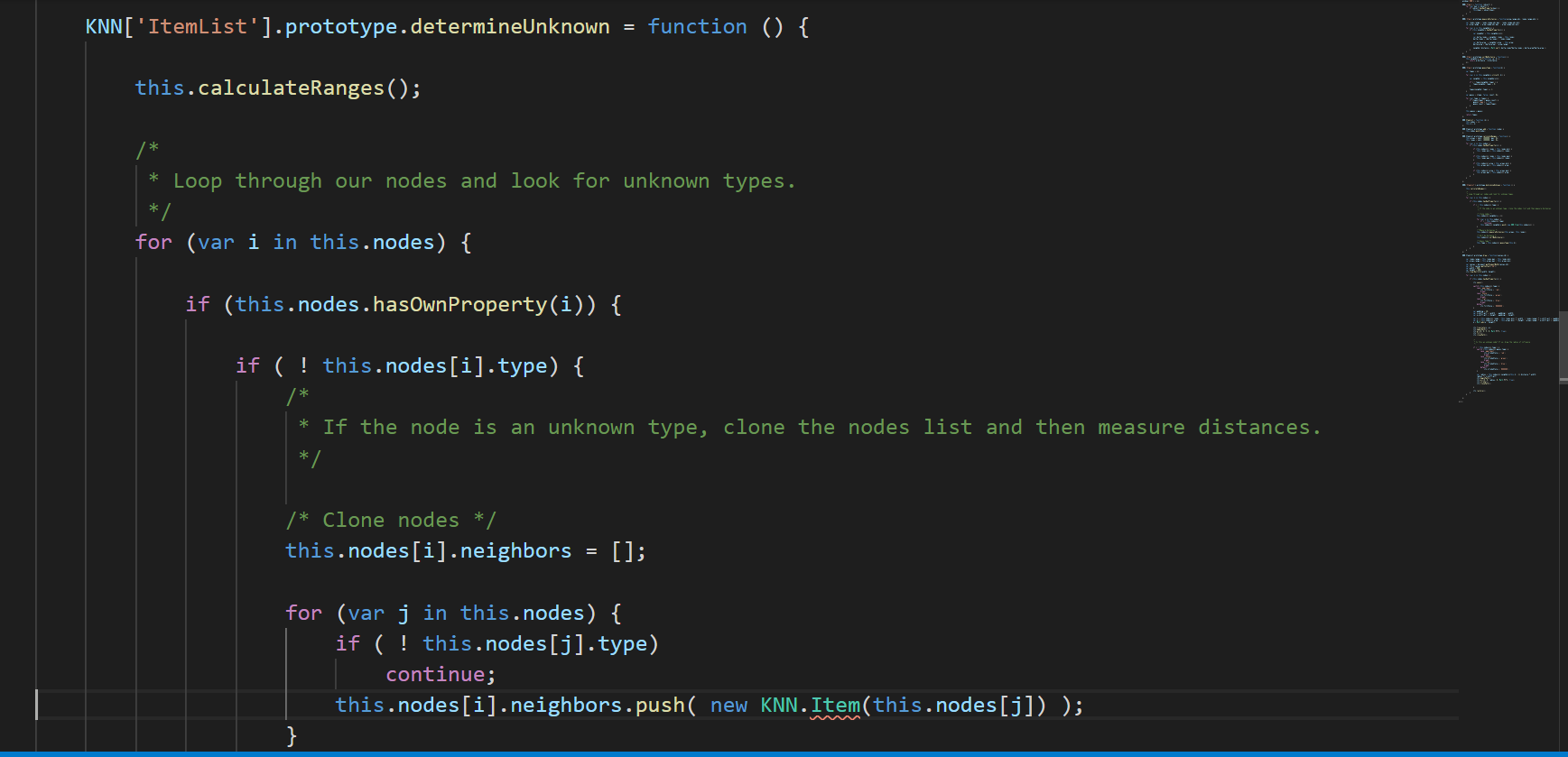


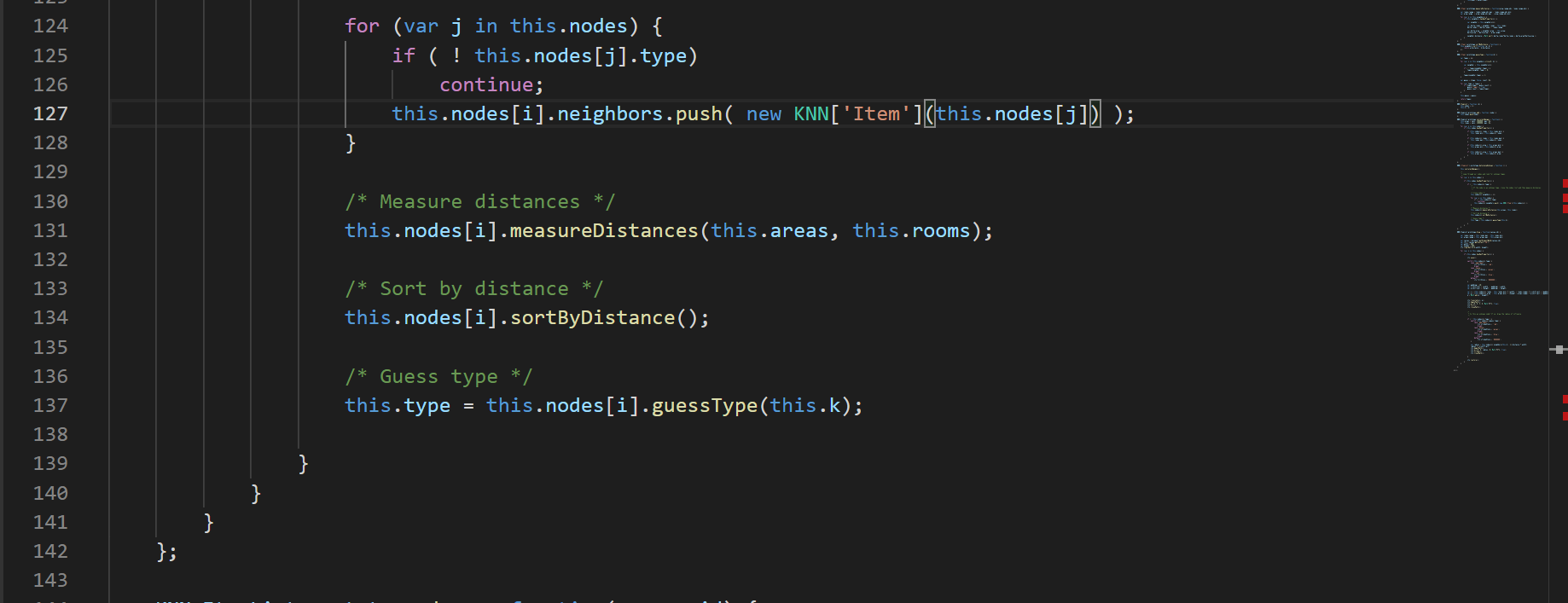
The Machine Learning Algorithm: (K-Nearest Neighbor):

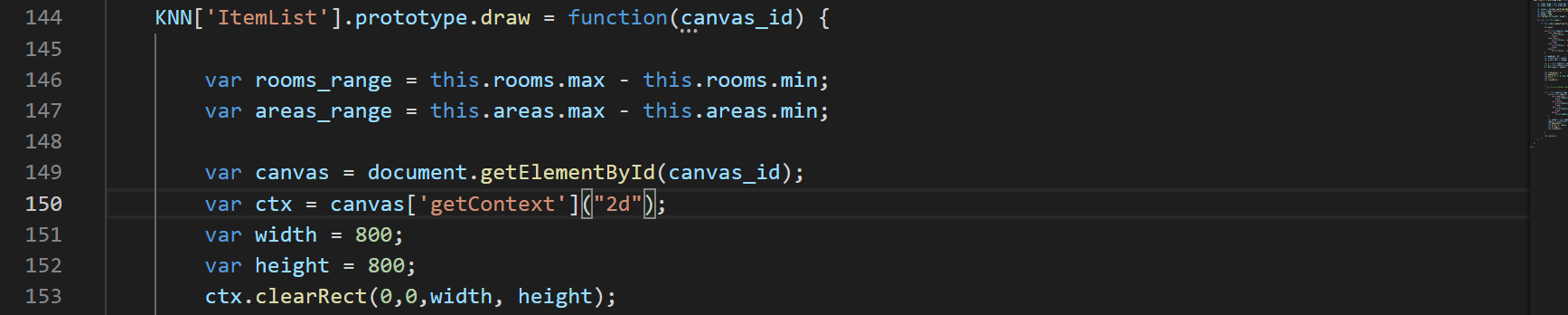












The Data Set Used for training Machine Learning Model is obtained from a Repository.

The URL is:

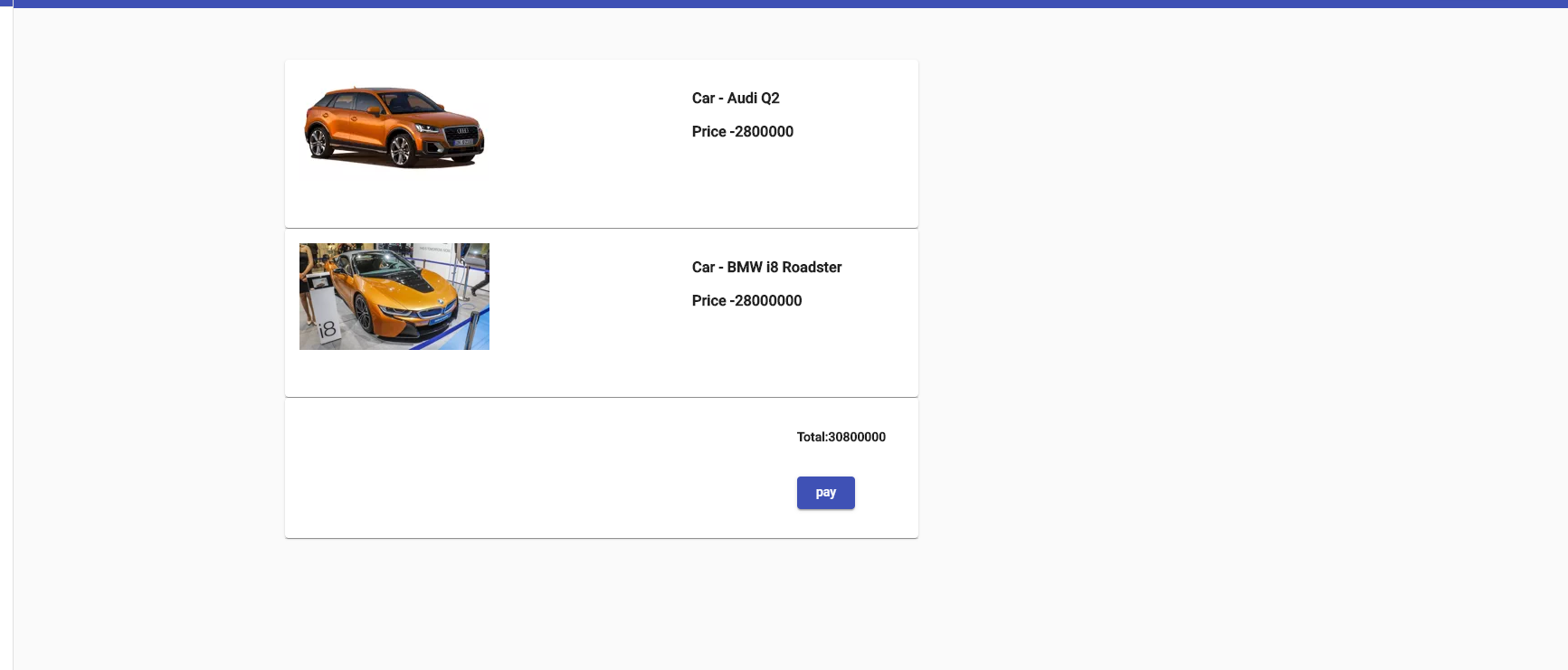
https://raw.githubusercontent.com/NearestLocation/master/Locations.json

**The Cart Module:**

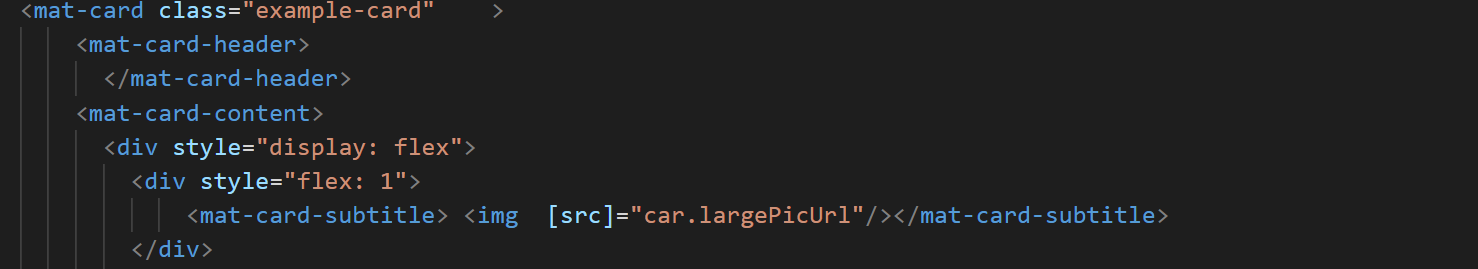
The cart Module is used to list out all the cars selected by the user.

The user can view all the cars and proceed to checkout page for the payment.

The screenshot of the cart page:



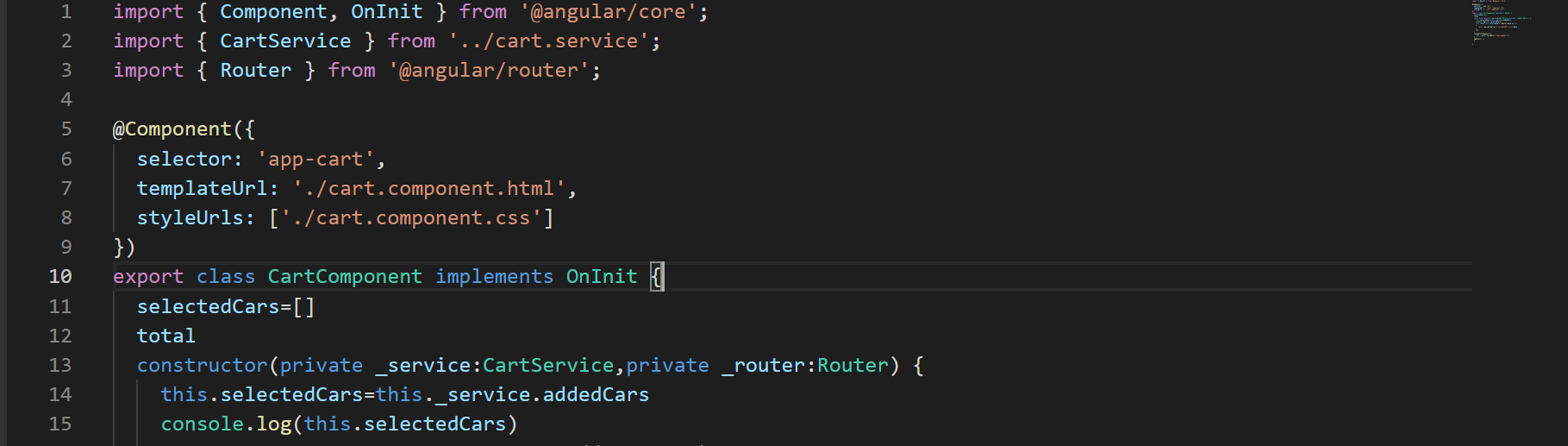
The HTML snippet for the cart:

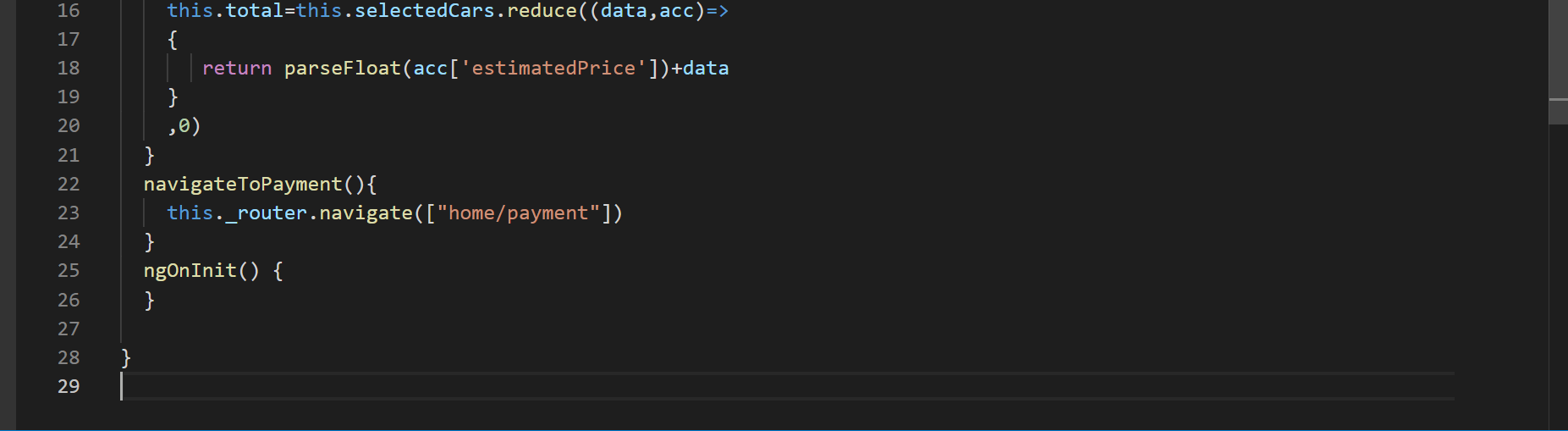






The Typescript Implementation for cart:





**The Payment Module:**

The user can navigate to Payment page to pay.

The user has all the available options for payment including,

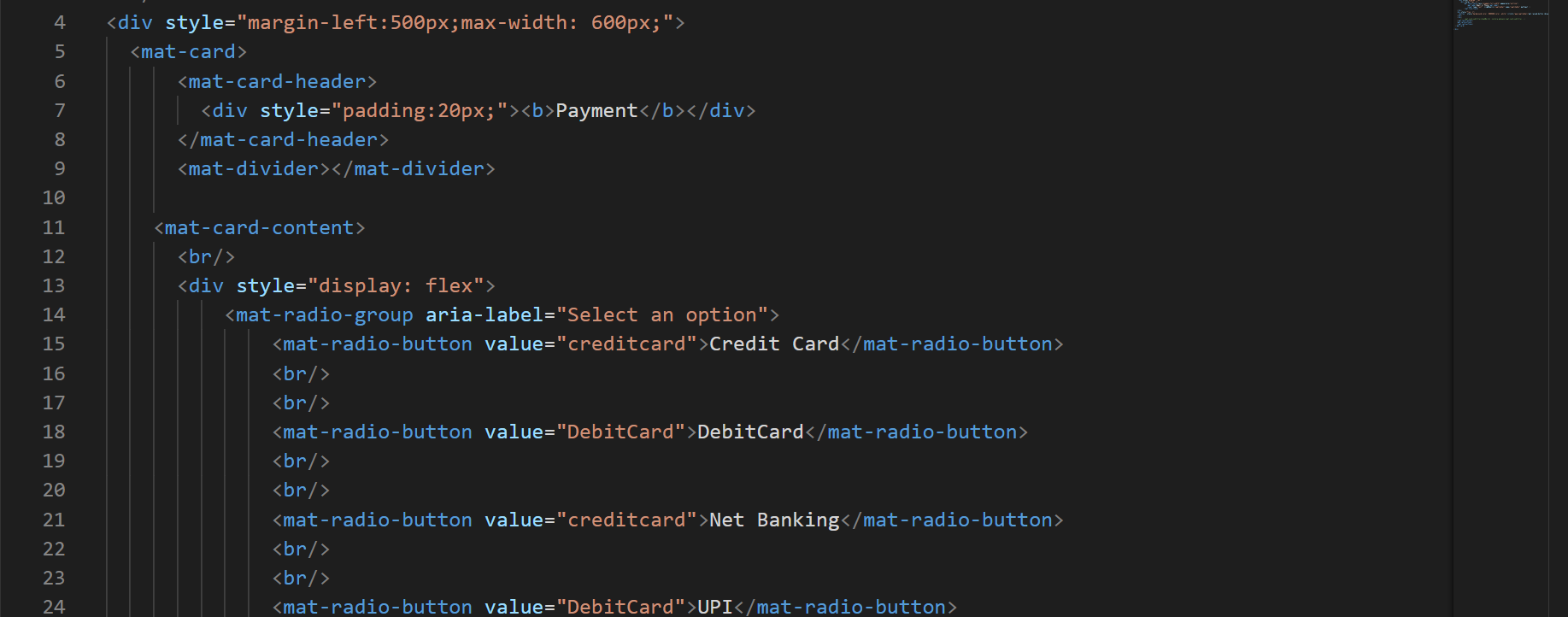
1)Net Banking.

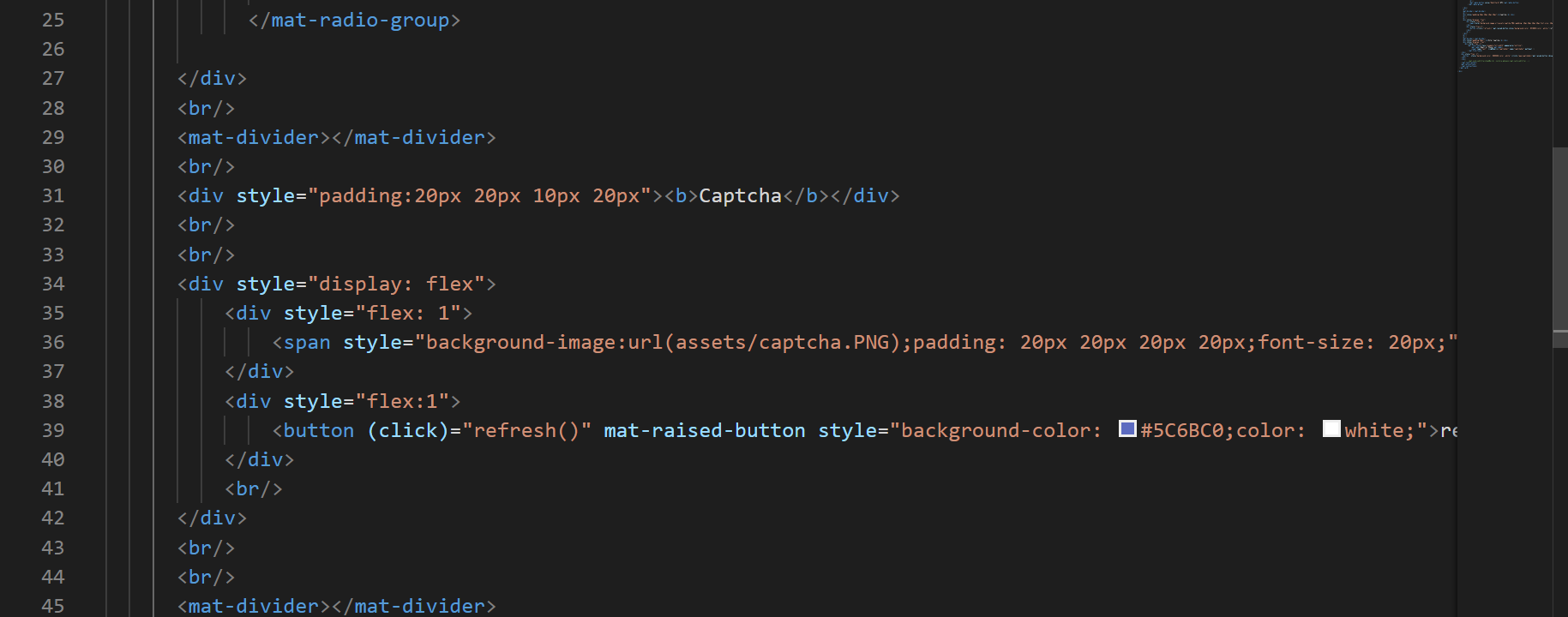
2)UPI.

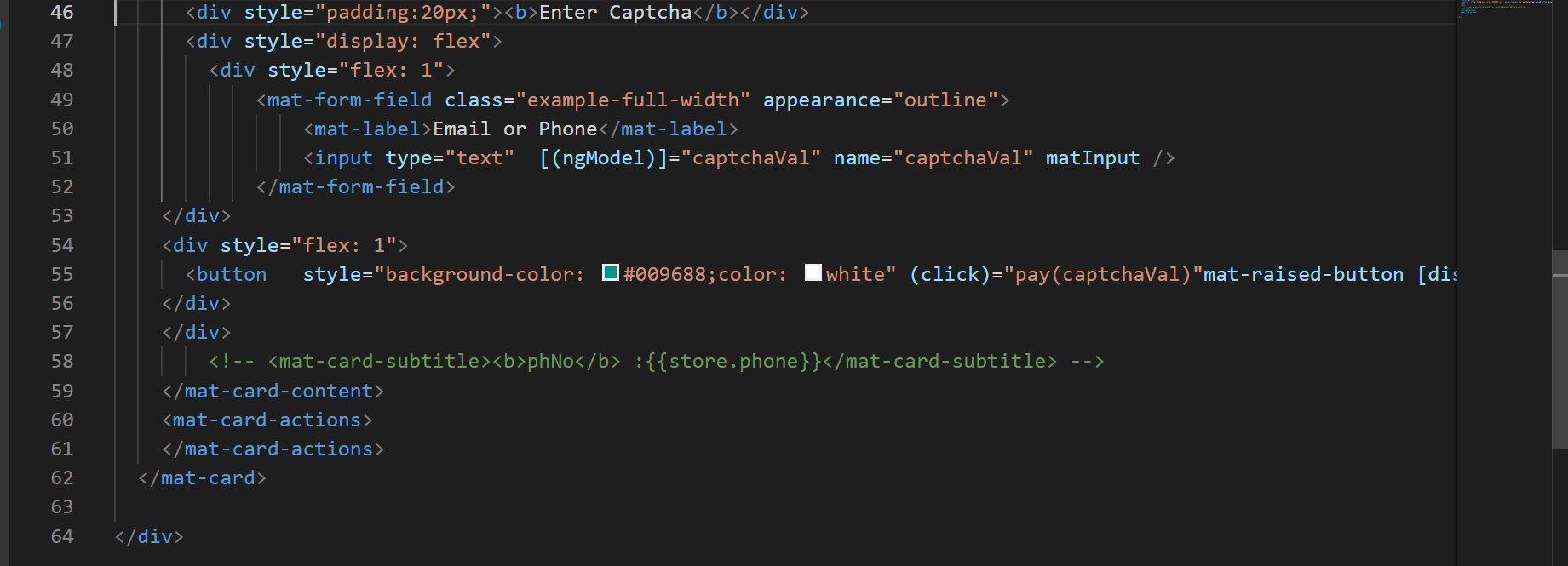
3)Credit Card.

4)Debit Card.

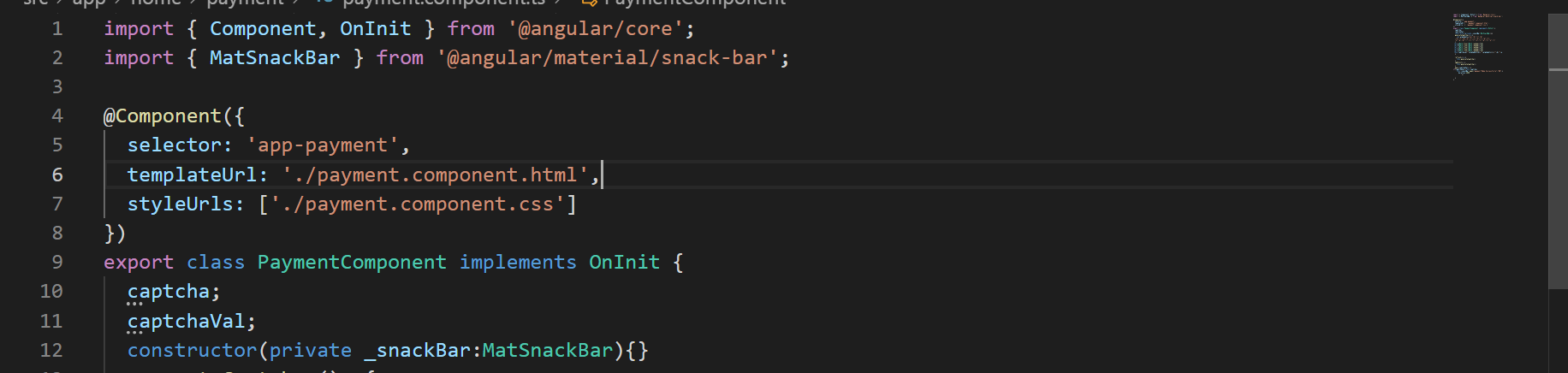
The HTML code Snippet:





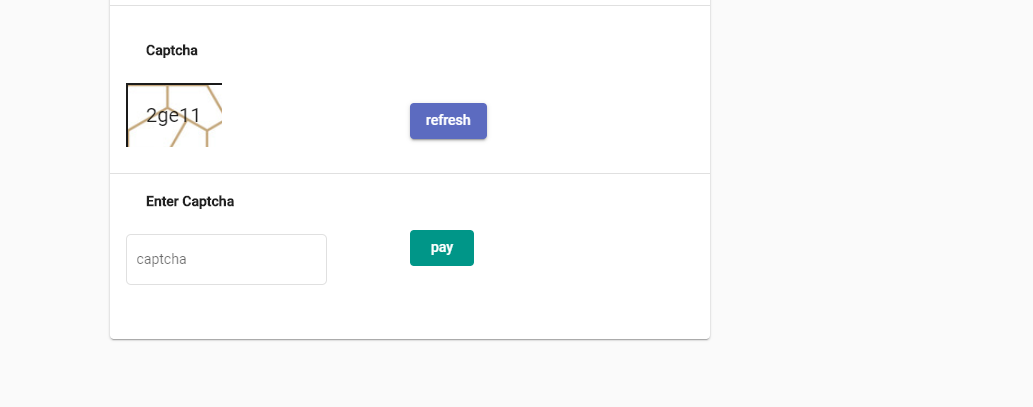


The Typescript Implementation :



The Payment Module has a captcha implementation for preventing the Bots from booking the cars online.

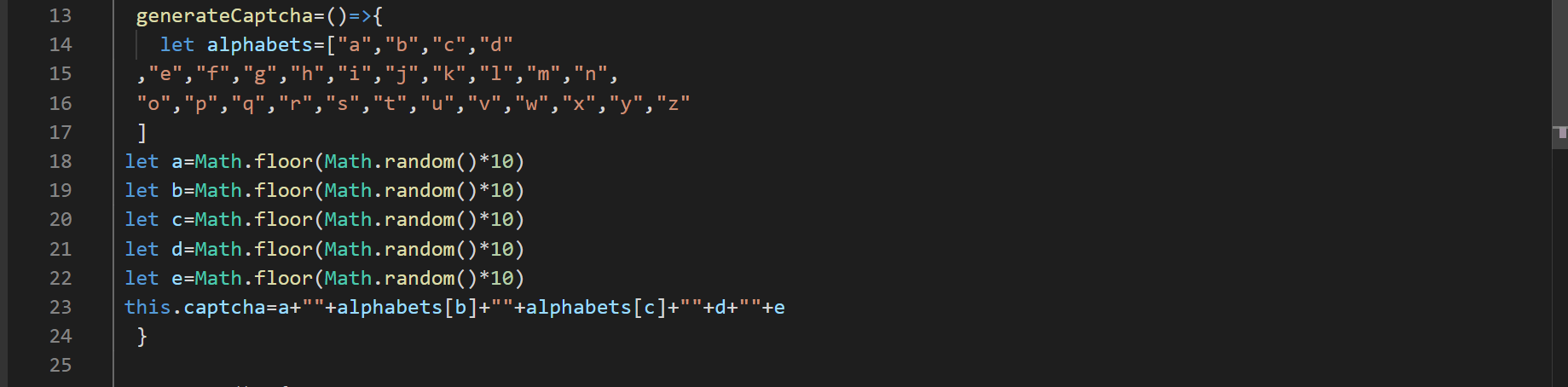
The Screenshot for captcha:



The HTML Code for the Captcha:



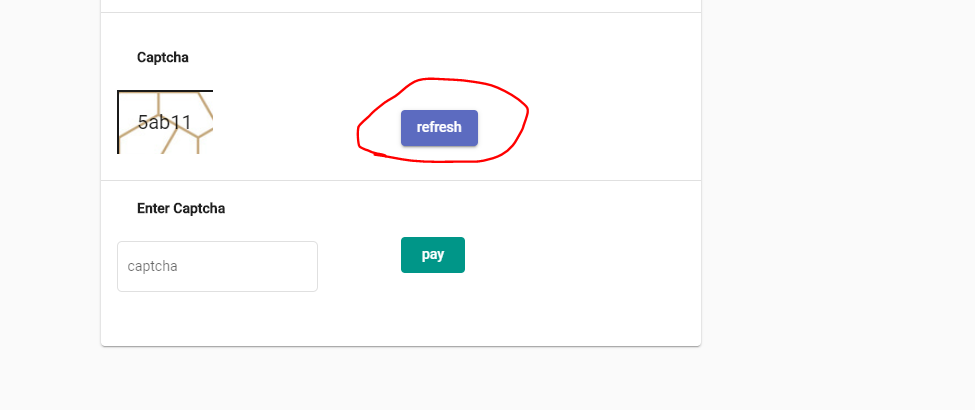
The TypeScript Implementation of Captcha:



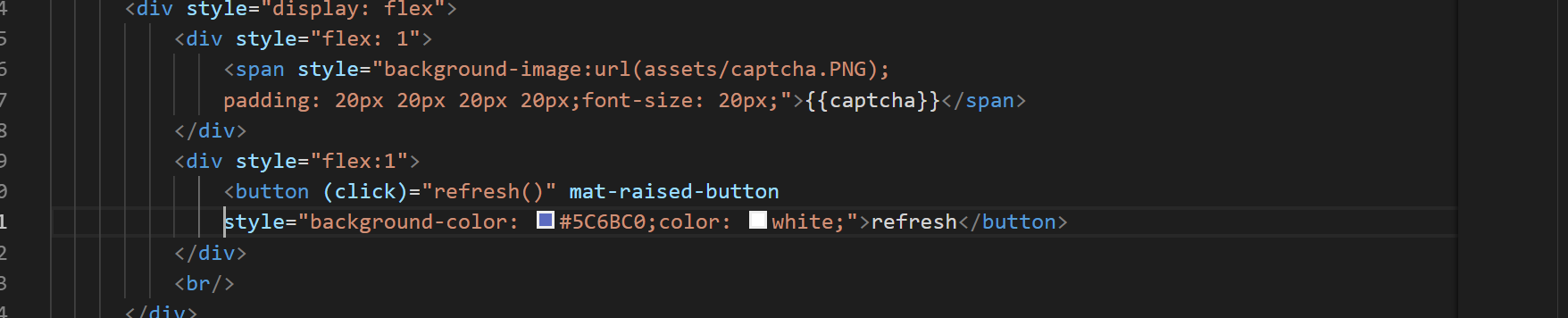
The captcha is generated by using the Random Class in JavaScript.

The user has to enter the captcha in order to proceed for payment, else the payment button is disabled.

The user can generate a new captcha by clicking on the refresh button.



The HTML snippet:



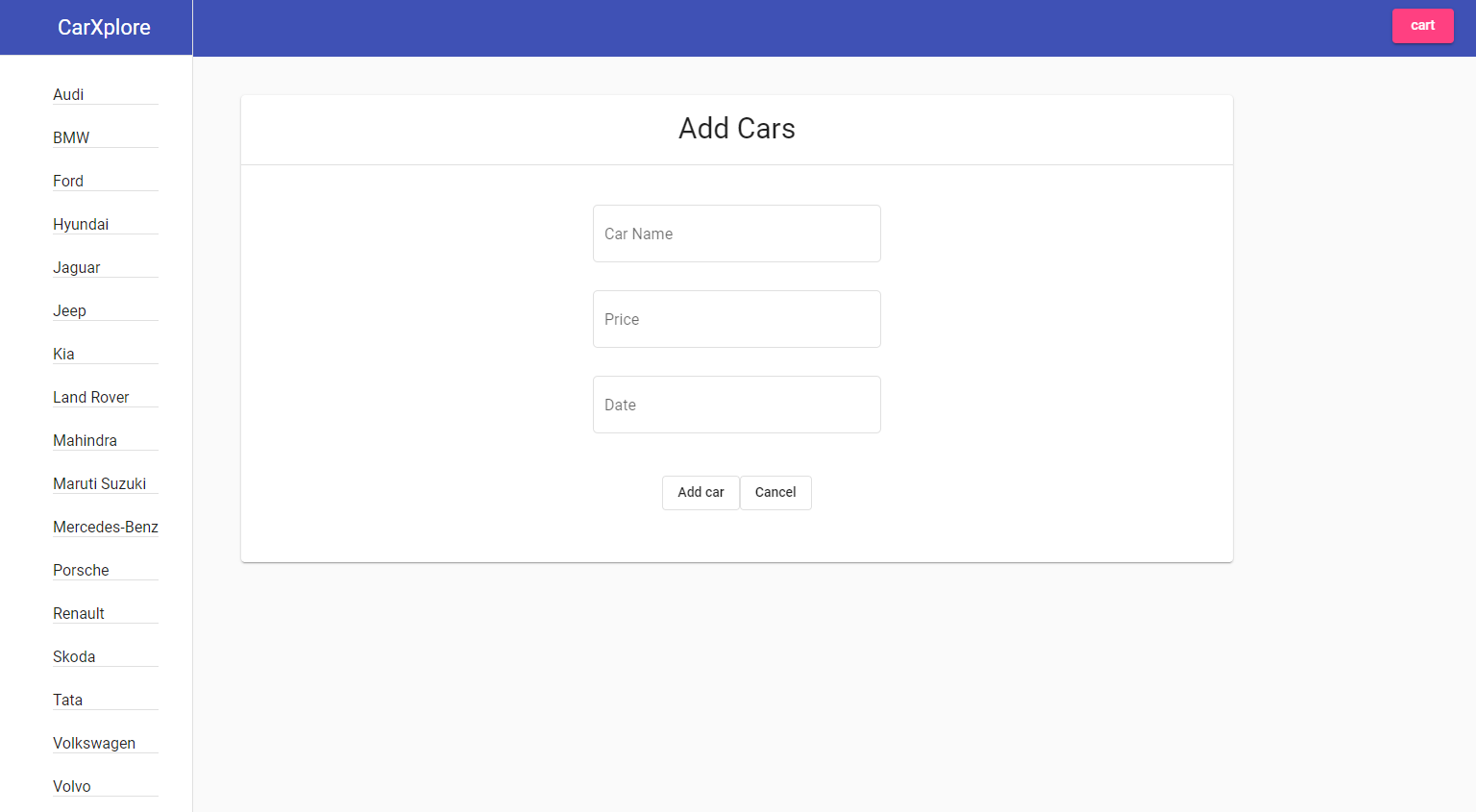
The TypeScript implementation:



**The Admin Module:**

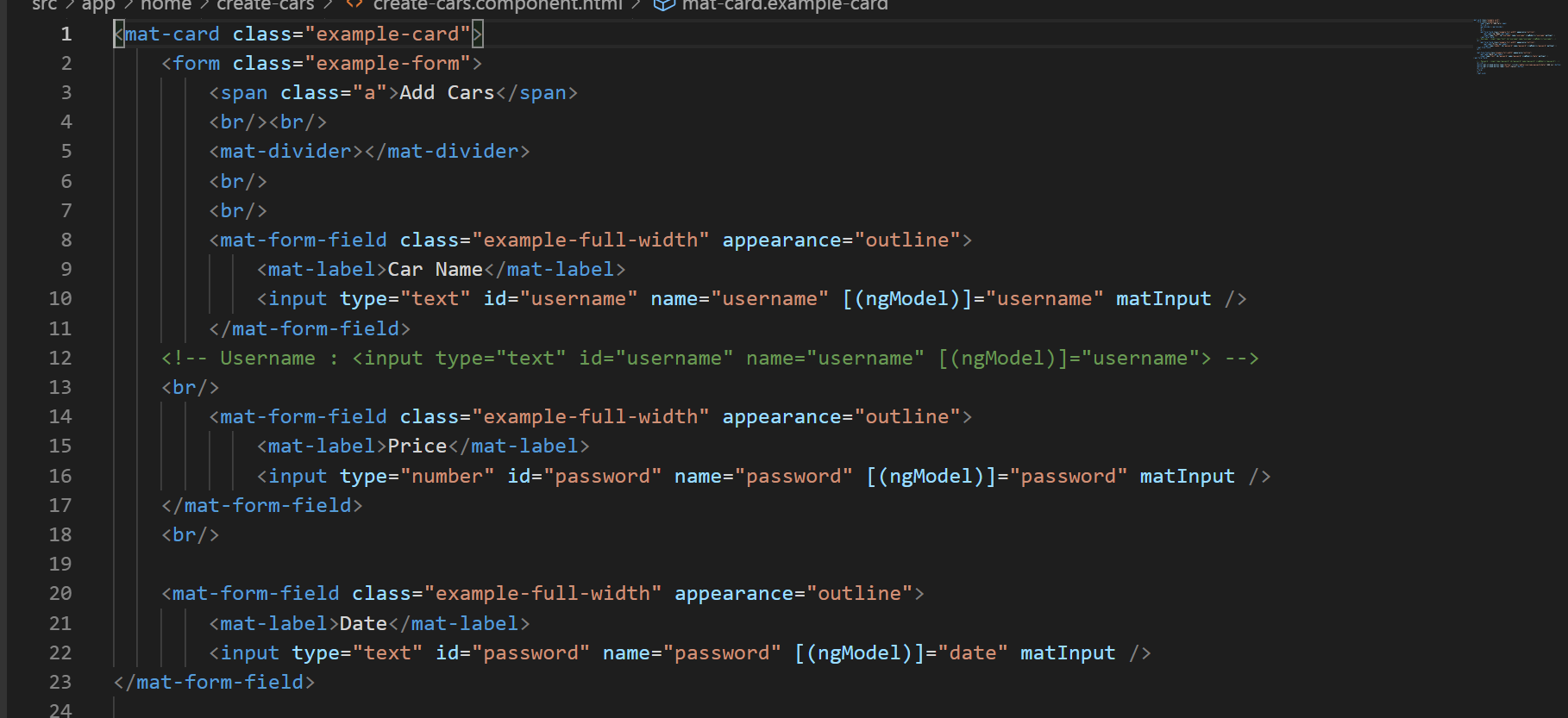
The admin has the option to add all the new cars.

The screenshot of the Add Car Page:

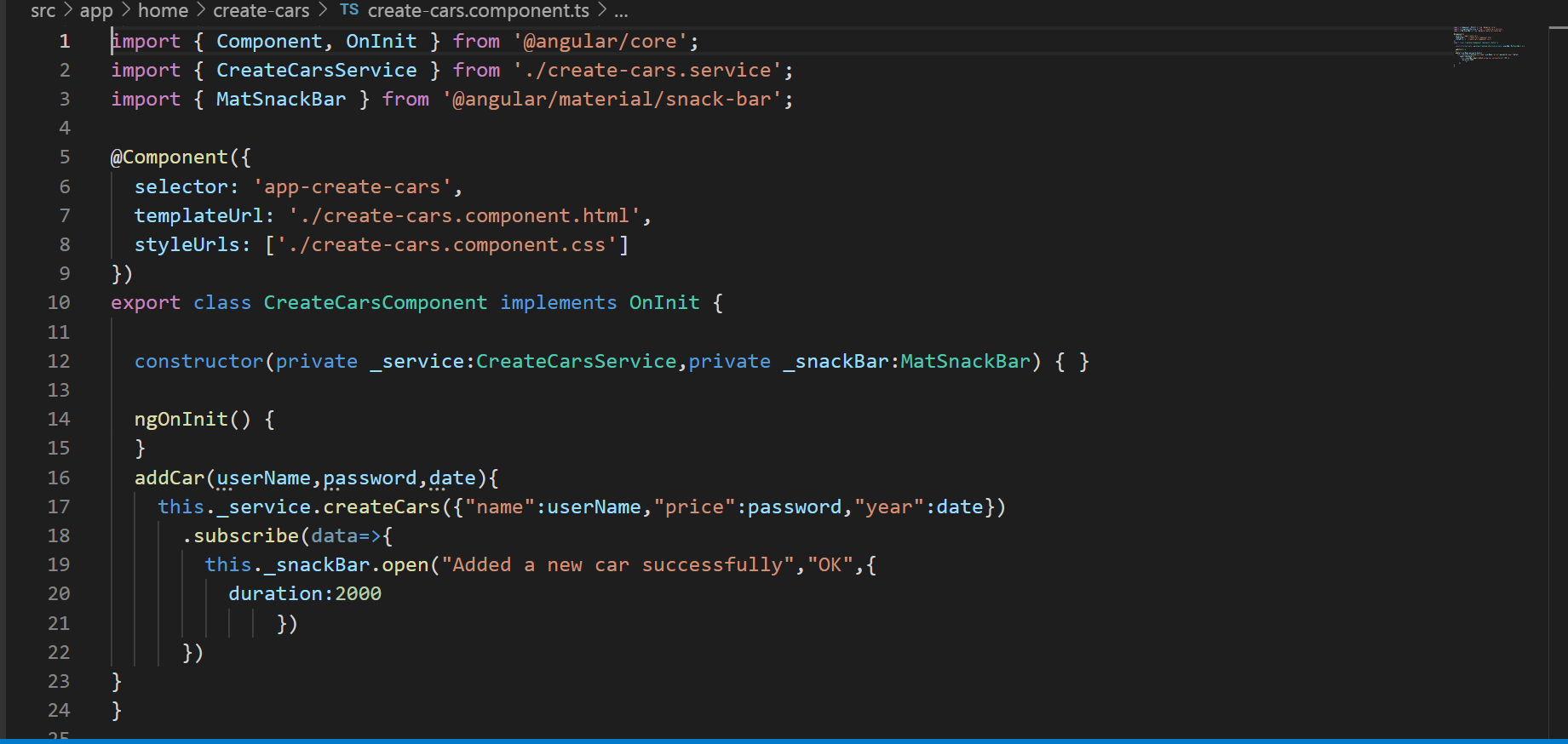


The Admin should fill all the car details and click on the Add car button to add the car to the Database.

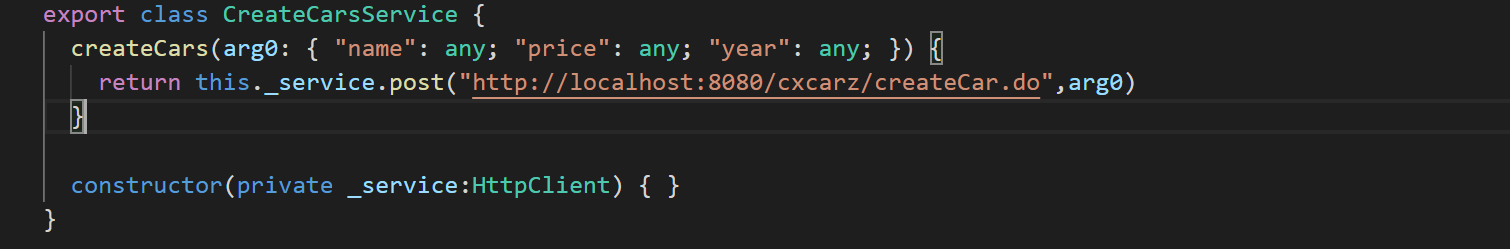
The HTML Snippet :



The TypeScript implementation :



The Service:



The Backend of the Add new car is implemented using Spring MVC and Spring ORM.

The snippet for Add New Car:

