# Lab: Spring Introduction MVC

# West Compass Dealer Shop - Part 2

West Compass Dealer Shop is an application in which you register cars, with several properties.

You will have to create a simple application which has several pages and some object entities.

## Data

This is the data layer of the application. There are some data object for you to implement.

### Brand

Create a Brand class, which holds the following properties:

* id – a **uuid or number**.
* name – a **name of brand**.
* **created** – a **date and time**.
* modified – a **date and time**.

### Model

Create a Model class, which holds the following properties:

* id – **uuid or number**.
* name – a **model name**.
* imageUrl – the **url of image**.
* startYear – a **number**.
* endYear – a **number**.
* **created** – a **date and time**.
* modified – a **date and time**.
* brand – a **model** **brand**.

### Offer

Create a Model class, which holds the following properties:

* id – **uuid or number**.
* description – some **text**.
* engine – **enumerated** value.
* imageUrl – the **url of image**.
* mileage – a **number**.
* price – the **price of the offer**.
* transmission – **enumerated** value.
* year – the **year** of offered car.
* **created** – a **date and time**.
* modified – a **date and time**.
* model – the **model of a car**.
* seller – a **user that sells the car.**

### User

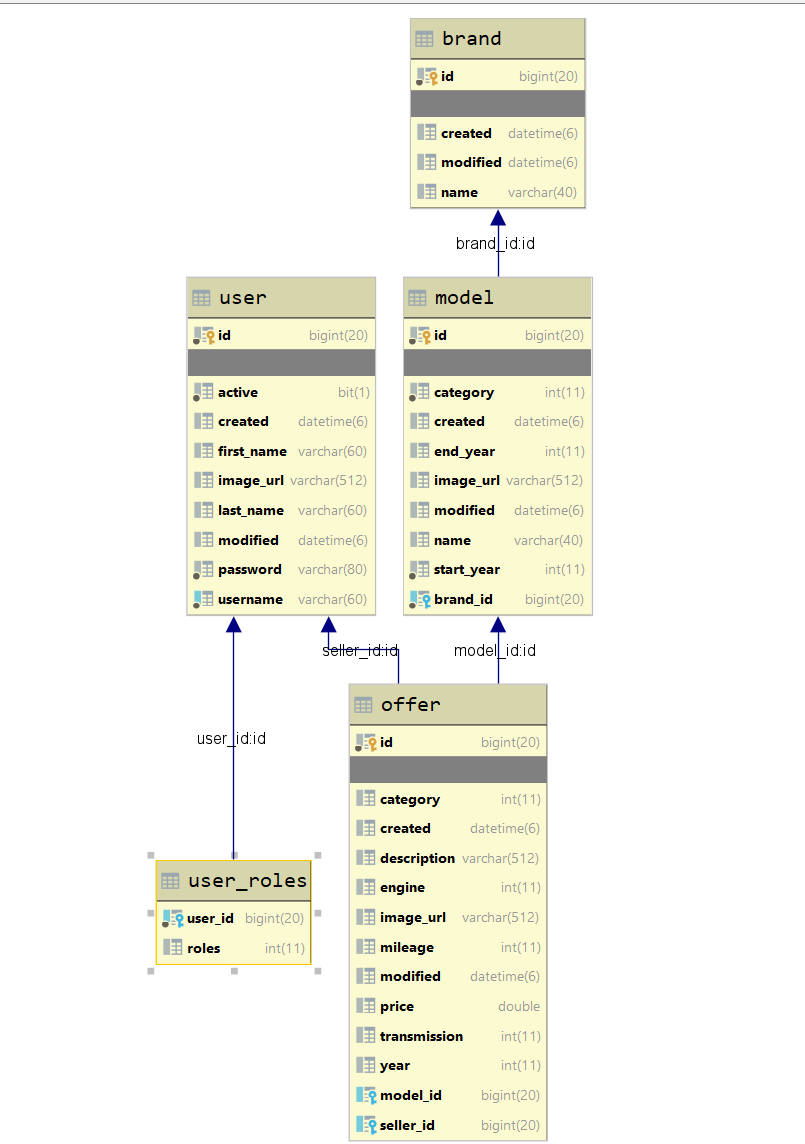
Create a User class, which holds the following properties:

* id – **uuid or number**.
* **username** – username of the **user**.
* **firstName** – first name of the **user**.
* **lastName** – last name of the **user**.
* active – **true OR false**.
* role – **user's role**.
* imageUrl – a url of user's picture.
* **created** – a **date and time**.
* modified – a **date and time**.

### UserRole

Create a UserRole class, which holds the following properties:

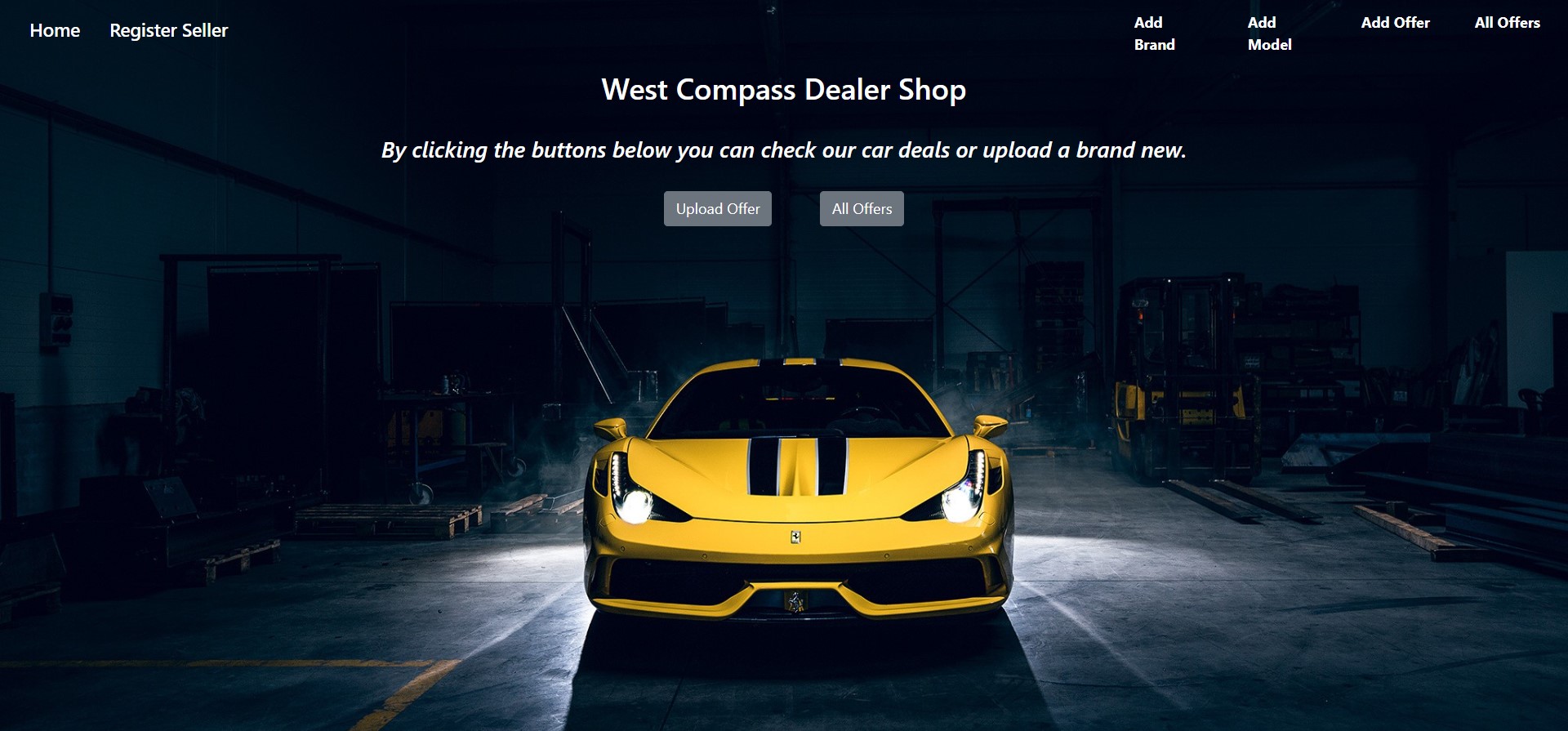
* id – **uuid or number**.
* **role** – **enumerated** value.

This is an example of ER Diagram

## Home/index - route ("/")

It should support only a **GET** request.

It should return the following HTML page, upon a **GET** request.

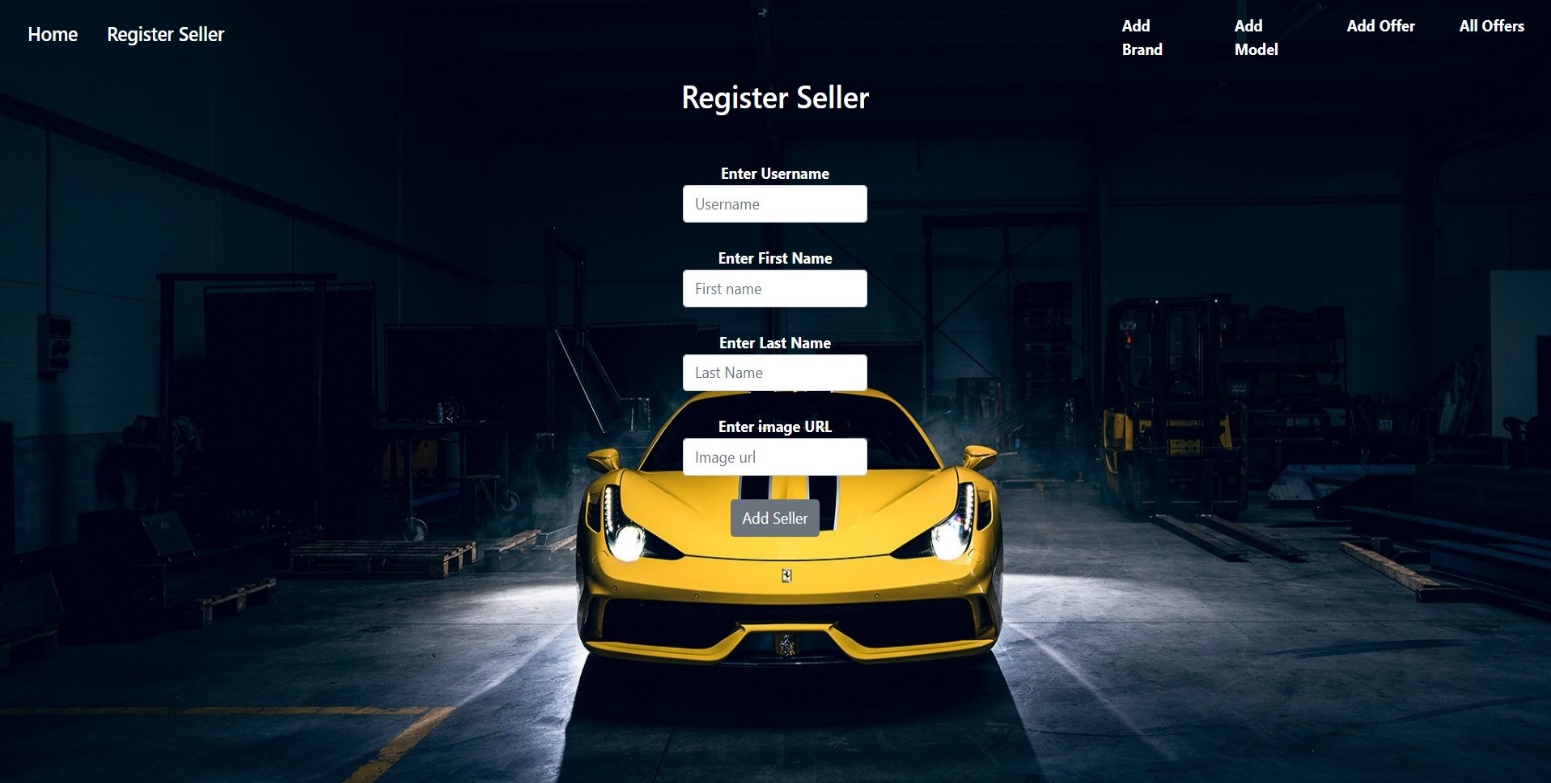


## Register Seller - route ("/users/register").

It should support only a **GET & POST** request.

It should return the following HTML page, upon a **GET** request.

First we need to add some sellers in our DB.



**Hint section:**

* **Because you will learn Thymeleaf in details on the next next lecture, we'll give you hints on how to implement some things**
* **Do not forget to add Thymeleaf in you pom.xml file**
* Do not forget to add Thymeleaf name spaces**:**



* Also you need to add in are the html form action and method (remember last lecture)**:**

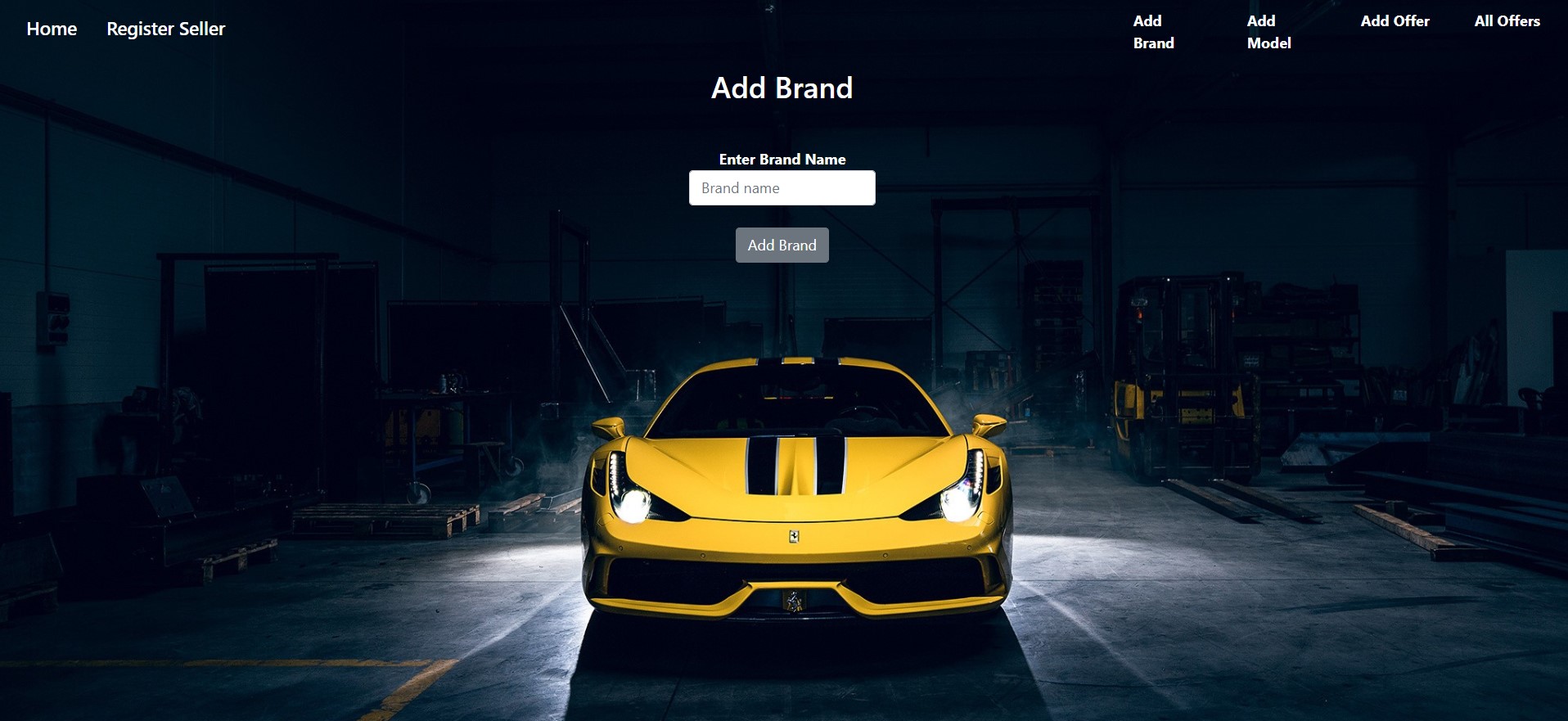


## Add Brand - route ("/brands/add")

It should support only a **GET & POST** request.

It should return the following HTML page, upon a **GET** request.

Now we add Brand of the most famous car manufacturers.



**Hint section:**

* **When adding Models, you need to load, from the DB, all the names of the brands in a select dropdown:**
* **In the controller, you need to pass brands (collections of brands names)**

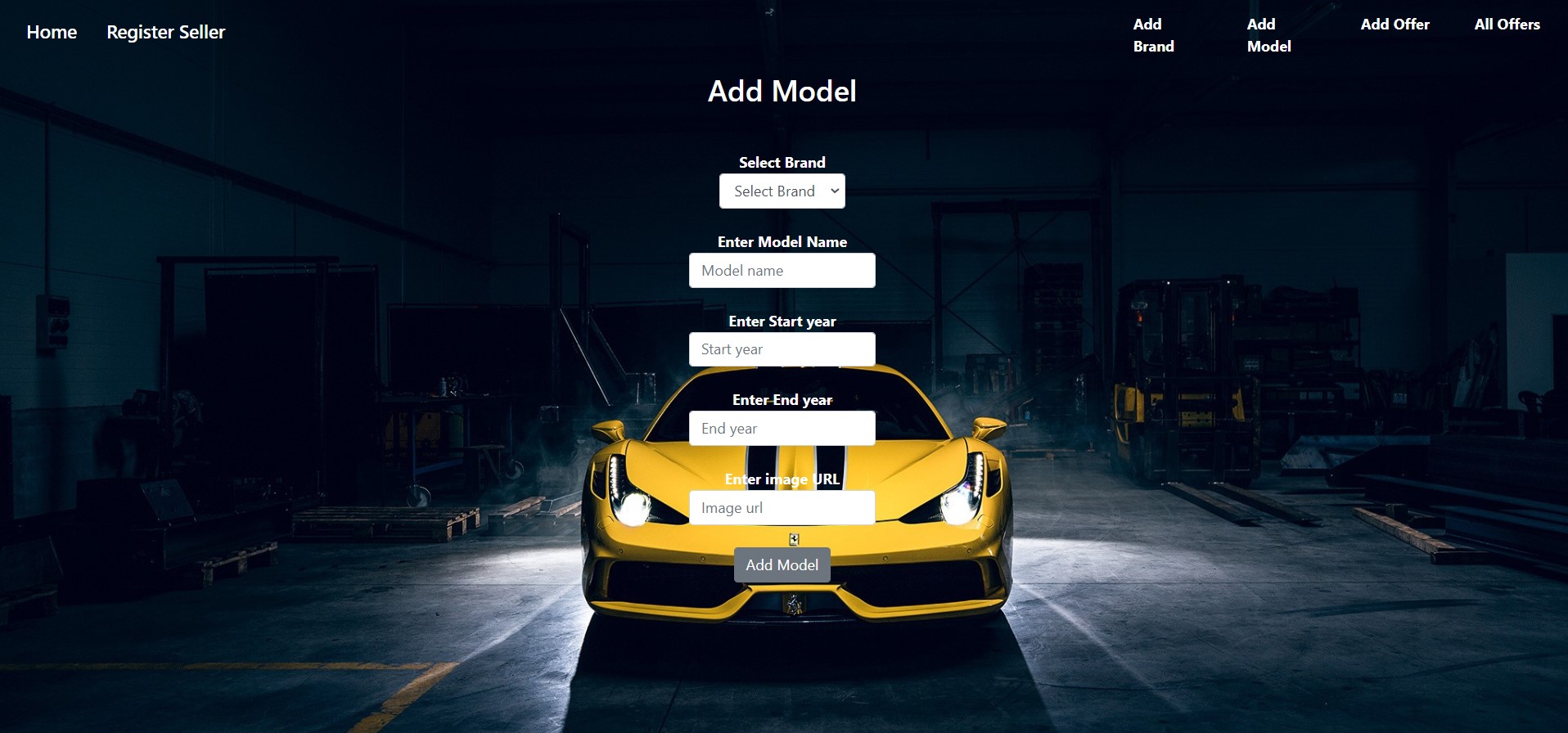


## Add Models - route ("/models/add").

It should support only a **GET & POST** request.

It should return the following HTML page, upon a **GET** request.

After this we add their top seller models. Select Brand dropdown show all registered users in DB.

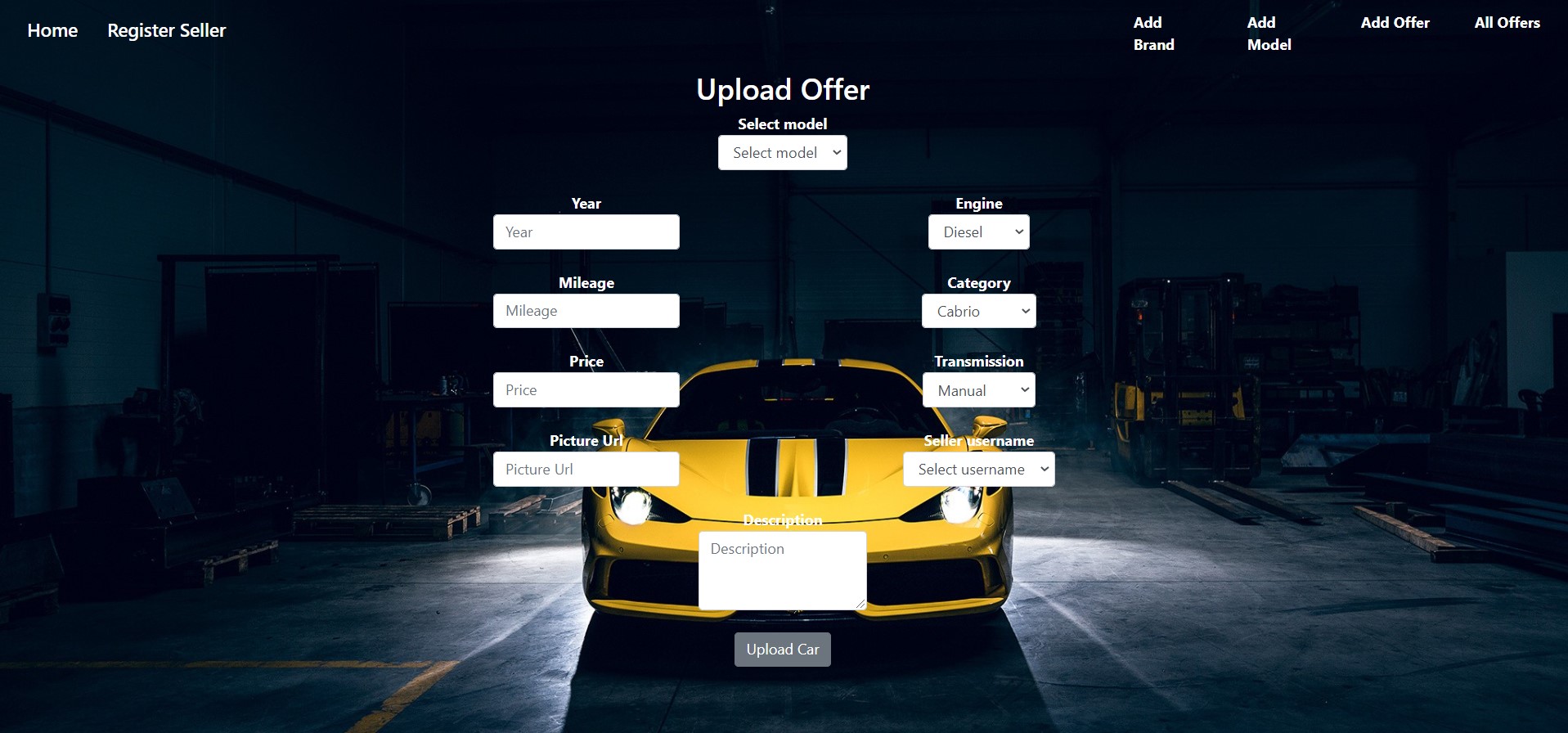


## Add Offers - route ("/offers/add")

It should support only a **GET & POST** request.

It should return the following HTML page, upon a **GET** request.

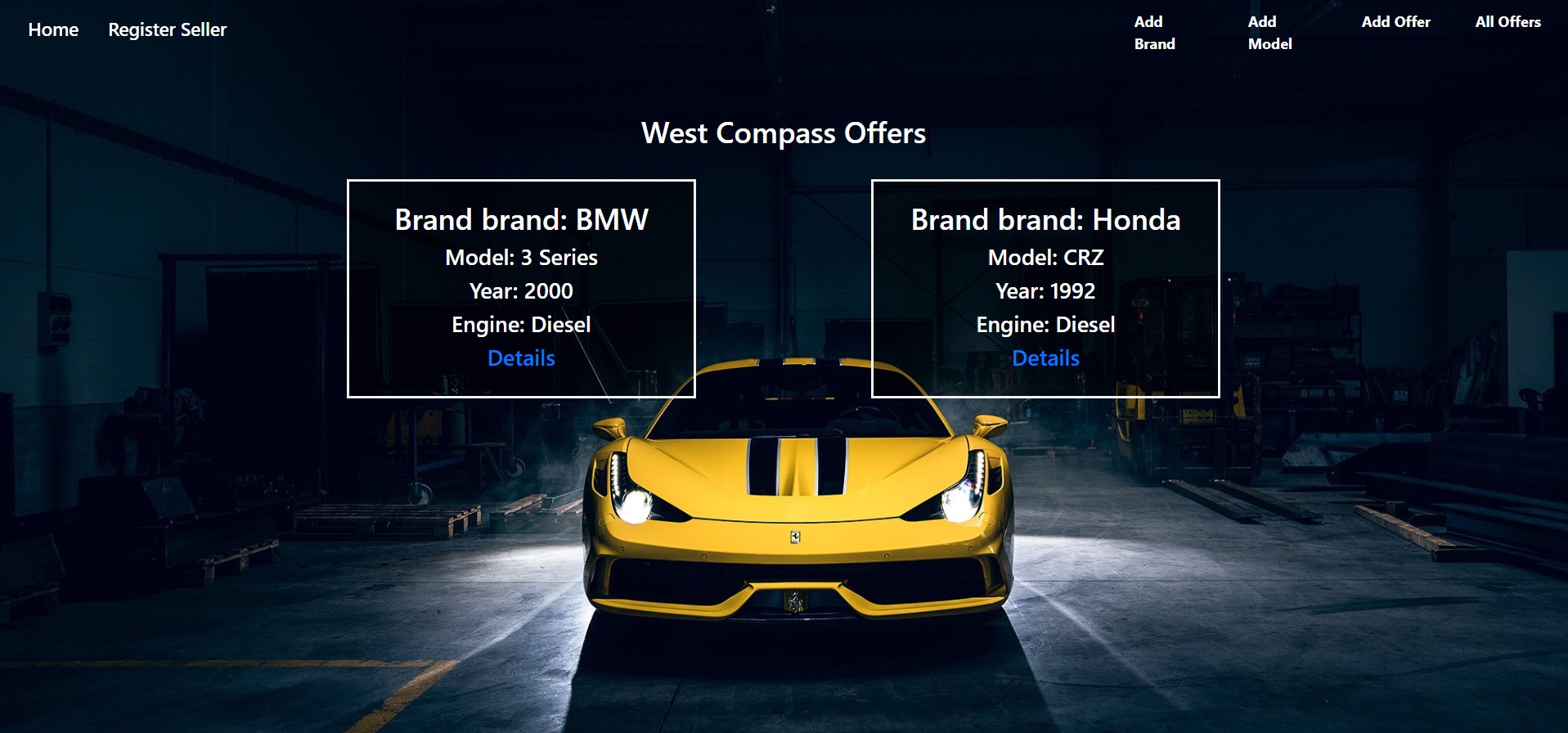
And finally we can add the real offer. Select model and Select username, shows info from our DB.



## All Offers - route ("/offers/all").

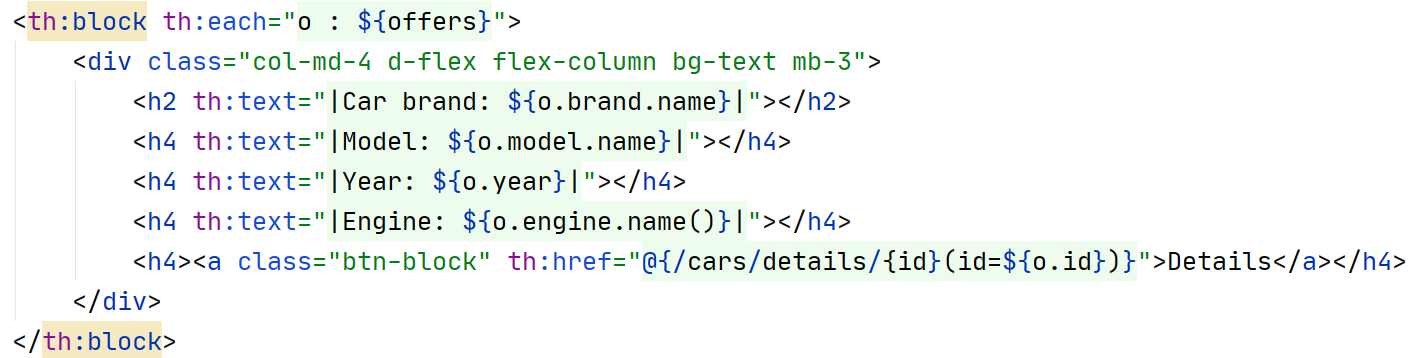
It should support only a **GET & POST** request.

It should return the following HTML page, upon a **GET** request.



**Hint section:**

* This is an example code of how to visualize all offers in the **all.html** page using **th:each**:
* **In the controller, you need to pass the offers (collections of offers view models)**

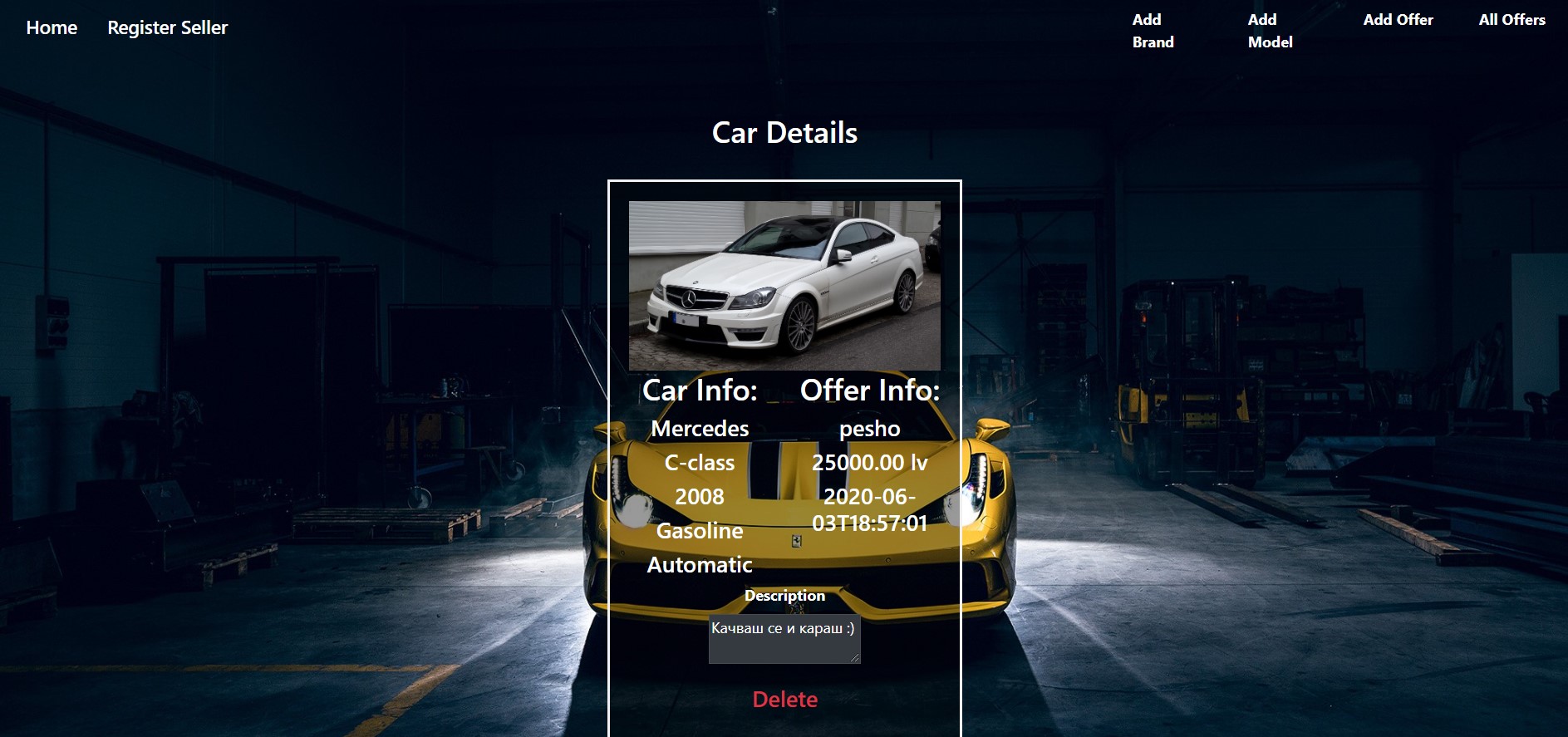


* This is a link to the documentation
  + <https://www.thymeleaf.org/doc/tutorials/2.1/usingthymeleaf.html#using-theach>

## Offer Details - route ("/offers/details/{id}").

It should support only a **GET** request.

It should return the following HTML page, upon a **GET** request.



**Hint section:**

* This is an example code of how to visualize offer details in the **details.html** page using **thymeleaf**:
* **In the controller, you need to pass offer (view model of Offer entity)**



We continue with more functionality on the next lab. ☺