# Exercises: Spring Essentials

Problems for exercises and homework for the [“Java MVC Frameworks - Spring” course @ SoftUni](https://softuni.bg/trainings/1538/java-mvc-frameworks-spring-march-2017).

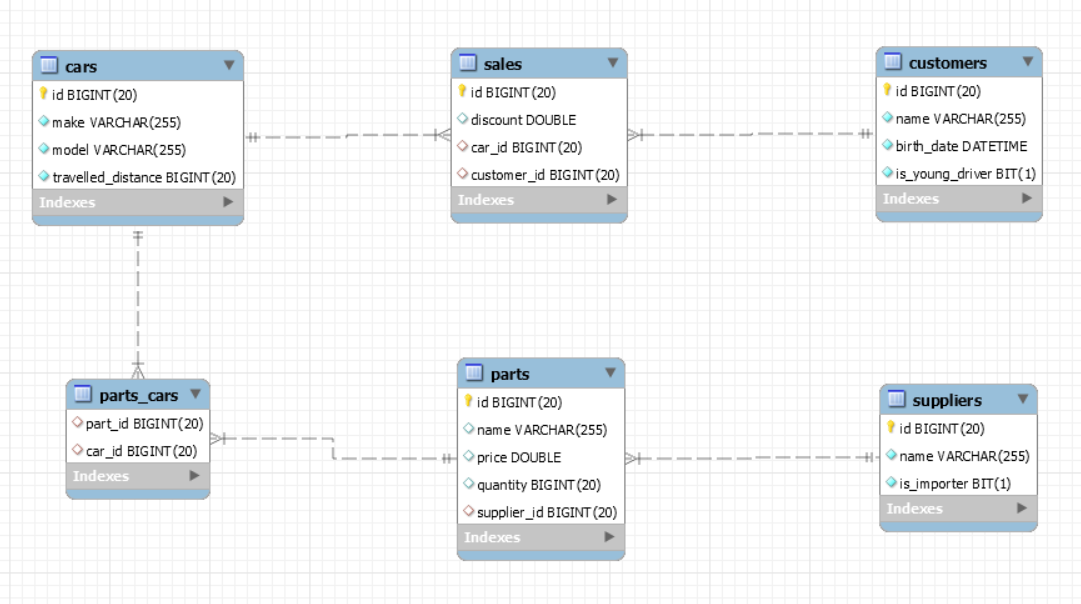
## Car Dealer

You need to create a Spring Boot project with database model as follows.

A car dealer needs information about cars, their parts, parts suppliers, customers and sales.

* **Cars** have **make, model**, travelled distance in kilometers
* **Parts** have **name**, **price** and **quantity**
* Part **supplier** have **name** and info whether he **uses imported parts**
* **Customer** has **name**, **date of birth** and info whether he **is young driver** (Young driver is a driver that has **less than 2 years of experience**. Those customers get **additional 5% off** for the sale.)
* **Sale** has **car**, **customer** and **discount percentage**

A **price of a car** is formed by **total price of its parts**.



Relations between models:

* A **car** has **many parts** and **one part** can be placed **in many cars**
* **One supplier** can supply **many parts** and each **part** can be delivered by **only one supplier**
* In **one sale**, only **one car** can be sold
* **Each sale** has **one customer** and **a customer** can buy **many cars**

## Car Dealer Import Data

Use the provided SQL script to **create and populate the database** with sample data.

## Queries

Using the provided project skeleton create an application that can show to the user results from different queries described below. The results from the queries should be visualizes **in a table.**

#### Query 1 – Ordered Customers

Get all **customers** ordered by their **birth date ascending or descending**. If two customers are born on the same date **first print those who are not young drivers** (e.g. print experienced drivers first). **Show** their name, birthdate and whether he is young driver or not.

##### Example URLs

{host}/customers/all/ascending

{host}/customers/all/descending

#### Query 2 – Cars from make

Get all **cars** from make **provided in the URL** and **order them by model alphabetically** and by **travelled distance descending**. **Show** them in a table with their **make**, **model** and **travelled distance**.

##### Example URL

{host}/cars/{make}

#### Query 3 – Filter Suppliers

Get a list of all suppliers. There are two types of suppliers

* **Local** – do not import parts from abroad
* **Importers** – import parts from abroad

Get their **id**, **name** and the **number of parts they can offer to supply**.

##### Example URLs

{host}/suppliers/local

{host}/suppliers/importers

#### Query 4 – Cars with Their List of Parts {by id}

Get all **cars along with their list of parts**. For the **car** get only **make, model** and **travelled distance** and for the **parts** get only **name** and **price**.

##### Example URL

{host}/cars/{id}/parts

#### Query 5 – Total Sales by Customer

Get a **customer by ID** and show his **name**, **bought cars** **count** and **total spent money** on cars.

##### Example URL

{host}/customers/{id}

#### Query 6 – Sales with Applied Discount

Get all **sales** with information about the **car (make, model, travelled distance)**, **customer** and **price** of the sale **with and without discount** and the **discount percent** itself.

##### Example URLs

{host}/Sales – list of all sales

{host}/Sales/{id} – more details about sale by provided Id (car make, car model, and customer name)

{host}/Sales/discounted – all sales that are discounted by any percentage

{host}/Sales/discounted/{percent} – all sales that are discounted with given percent

## Hyperlinks

Add menu to the website with the following structure

* **Customers** - customers/all/ascending
* **Sales**
  + **All** – /sales
  + **Discounted** - sales/discounted
* **Cars** - cars/all list of all cars with their make, model and travelled distance

## Add Customers

Add functionality to **add customers** to the database. A customer should be added by providing a **name** and **birthdate**. Use appropriate **controls** for the form for each field and display well formatted labels for the input fields. Note that the field whether he is young driver or not should be filled automatically depending on the provided birthdate.

## Edit Customers

Add option to **edit a customer** from the database. The **name** and the **birthdate** to each customer can be changed.

## Add Parts

Add functionality to **add car parts**. A car park should be added successfully by only providing **name, price** and **supplier**. The **default quantity** if not specified explicitly is **1**. Use appropriate controls for the form and well formatted labels for the input fields.

## Delete Parts

Add functionality to see a **list of all parts** with option to **delete** any of them. When trying to delete some part the user should be **prompted** for confirmation and if the user confirms then the part should be deleted from the database.

## Edit Parts

Once part is added to the database its **price** and **quantity** can be modified. All other fields are not allowed to be changed after that.

## Add Cars

Add functionality to **add cars** to the database. A car should be added if has valid **make, model** and **travelled distance**. Choose appropriate controls for the form and well formatted labels for the input fields.

## Add Cars with Parts

Modify the form from the previous exercise and add another control that allows adding parts for the car. This might be:

* **Text field** where the IDs of the parts can be placed separated with a space
* Group of **checkboxes**
* \*\*Multiple **dropdown** fields

## Add Users

Add feature that allows users to be **registered** in the system as well as **logged in** and **logged out.** The logged in users should have **several privileges** that not logged in users should be restricted. These privileges are described in the tasks below.

To register users should provide **email**, **username** and **password**. To log in the user must provide **username** and **password**. Once logged in he can log out.

## Change Permissions

Right now, every visitor of our web application can **add new cars**. Restrict that feature only to **users that are logged**.

## Add Sales

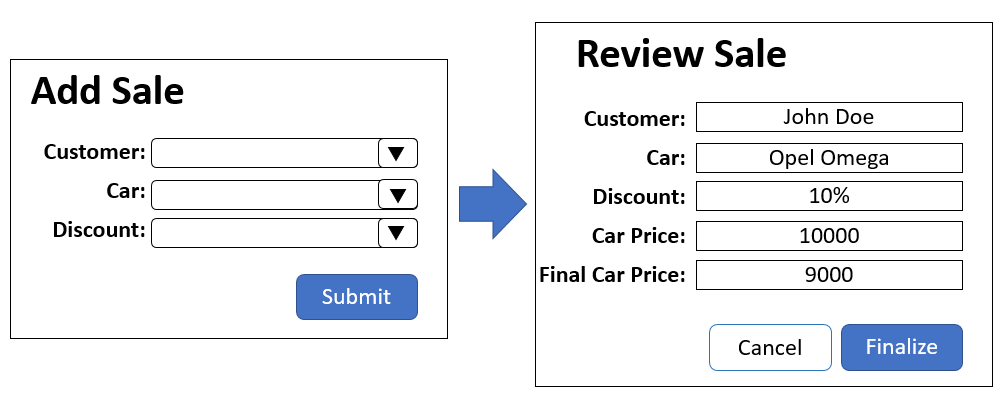
The logged in users should have option to **add sales**. A sale requires the user to choose **car** and **customer**. Also, the user should specify the **discount percentage** of the sale. After the user fill out the form when he submits it, he should be **redirected** to a page that shows **summarized information** for the sale:

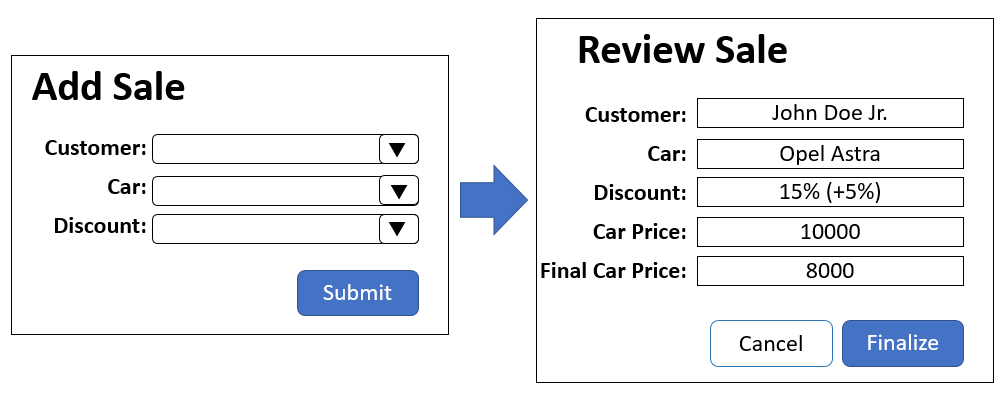
* customer **name**
* car **make** and **model**
* total **discount** percentage
* **price** of the car **before/after applied discount**

On that page, there should be a button that **finalizes** the sale and **add it to the database**.

Note that if the selected customer is young driver **additional 5%** should be added to the **discount** percentage.

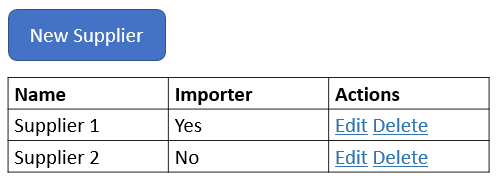
Once a sale is added to the database it should **not** be able to be **edited** or **deleted**.





## Add, Edit and Delete Suppliers

Add functionality to the logged users to **add**, **edit** and **delete** suppliers to/from the database. When supplier is deleted all of his parts should be deleted.



## Logger

Every time a user adds a car, sale or **add/edit/delete supplier** that action must be logged in the database in table

* **Logs** 
  + User
  + Operation - add, edit or delete
  + Modified table name,
  + Date and time of modifying).

All logs should be visible by logged in users on {host}/logs/all. The logs should be sorted by date ascending. The users can **delete** these logs

The page showing all logs should have also the following functionalities:

* Show **all logs for given user** by providing username
* **Clear** all logs from the database

