Coffe Shop

SE 352 – Object Oriented Enterprise Application Development

SE 352 Project

**Contents**

[Overview](#_Overview)

[Requirements](#_Requirements)

[Use Case](#_Use_Case)

[Description of problem](#_Description_of_problem)

[Design](#_Design)

[Sequence of major functionality](#_Sequence_of_major)

[Web UI (Common Case)](#_Web_UI_(Common_1)

[Table layout](#_Table_layout)

[Deployment](#_Deployment)

[Discussion of how your design met the requirements](#_Discussion_of_how)

[Discussion of lessons learned](#_Discussion_of_lessons)

[Decision log](#_Decision_Log)

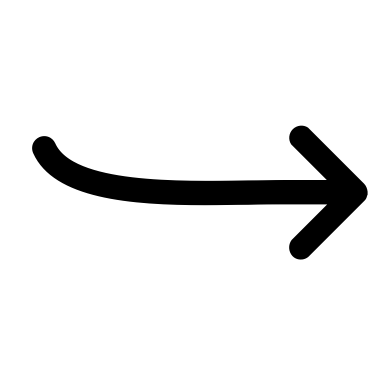
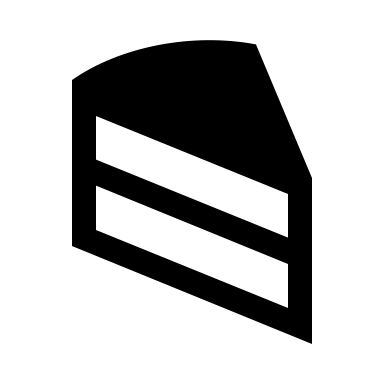
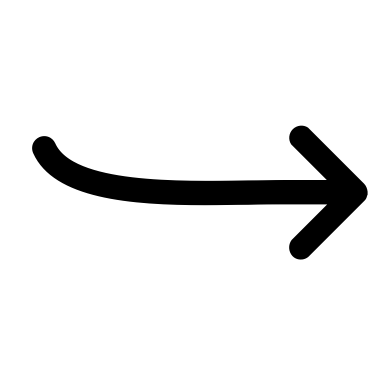
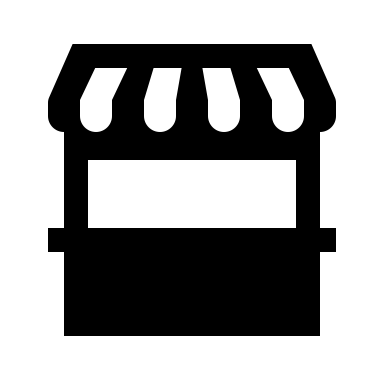
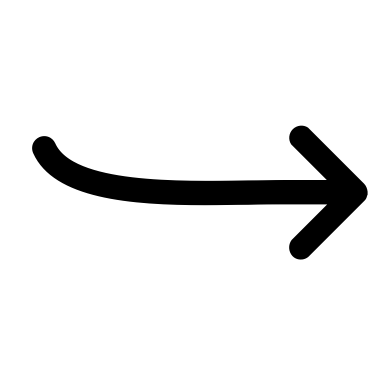
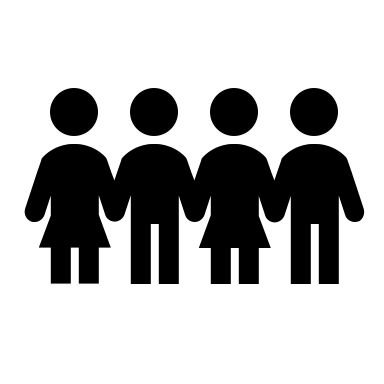
[Milestone Screenshot](#_Milestone_Screenshot_1)

# Overview

This is a web application for an imaginary coffee shop. It will allow users to select, view, buy and edit selected products. It would also allow registration for new customers and a log in for existing customers.

# Requirements

## Use Case



Users want coffee or sweets and order them through the application. The order is received and purchased, and the users get satisfied.

## Description of problem

Coffee shop application that allows clients to purchase the shop’s products.

# Design

## Sequence of major functionality

***For MYSQL:***

* Download MYSQL.
* Start the server.
* While the server is running you can operate with your database.

***Application.properties:***

Here is the configuration of the “application.properties” to connect to the database. Using

*# Configuration for the datasource***spring.datasource.driver-class-name**=**com.mysql.jdbc.Driver  
spring.datasource.url**=**jdbc:mysql://localhost:3306/coffeeShop\_schema?serverTimezone=UTC&useSSL=false&allowPublicKeyRetrieval=true  
spring.datasource.username**=**root  
spring.datasource.password**=**julio123***# Configuration for Java Persistence (JPA)  
#Will show the SQL statements being executed.***spring.jpa.show-sql**=**true  
spring.jpa.properties.hibernate.dialect**=**org.hibernate.dialect.MySQL5Dialect  
spring.jpa.hibernate.ddl-auto**=**update**

### Web UI (Common case)

**Front Page (home) 🡪 localhost:8080/home or localhost:8080/home**



* Products and Shopping Cart at the moment are empty pages that can be requested like this:

**Products:** *localhost:8080/products*

**Shopping Cart:** *localhost:8080/shoppingcart*

**Home:** *localhost:8080/home & localhost:8080*

## Table layout

* I also have a Schema in MYSQL called “**coffeeShop\_schema**”.
* JPA creates the tables automatically.

## Deployment

* Download the code.
* Run the application.

# Discussion of how your design met the requirements

* Right now, my design meets the requirement of a Java Enterprise application thanks to the Spring framework. Uses the Model Viewer Controller (MVC) design pattern.
* Framework uses Controllers / Servlets
* Framework was configured to use Java Server Pages (JSP) as the view.

# 

# Discussion of lessons learned

**Lessons Learned Milestone 2:**

* Learned how to use a database client such as MongoClient to connect to a database.
* Learned how to map Java classes to Collections in the database as *Documents*.
* Learned how to *query* Documents from the Mongo database.
* Downloaded a plug-in on IntelliJ IDEA that gave me a GUI for my Mongo database. This GUI is called “MongoExplorer”.
* Learned how to connect my application to MYSQL.
* Learned how to map Entities to Java classes.

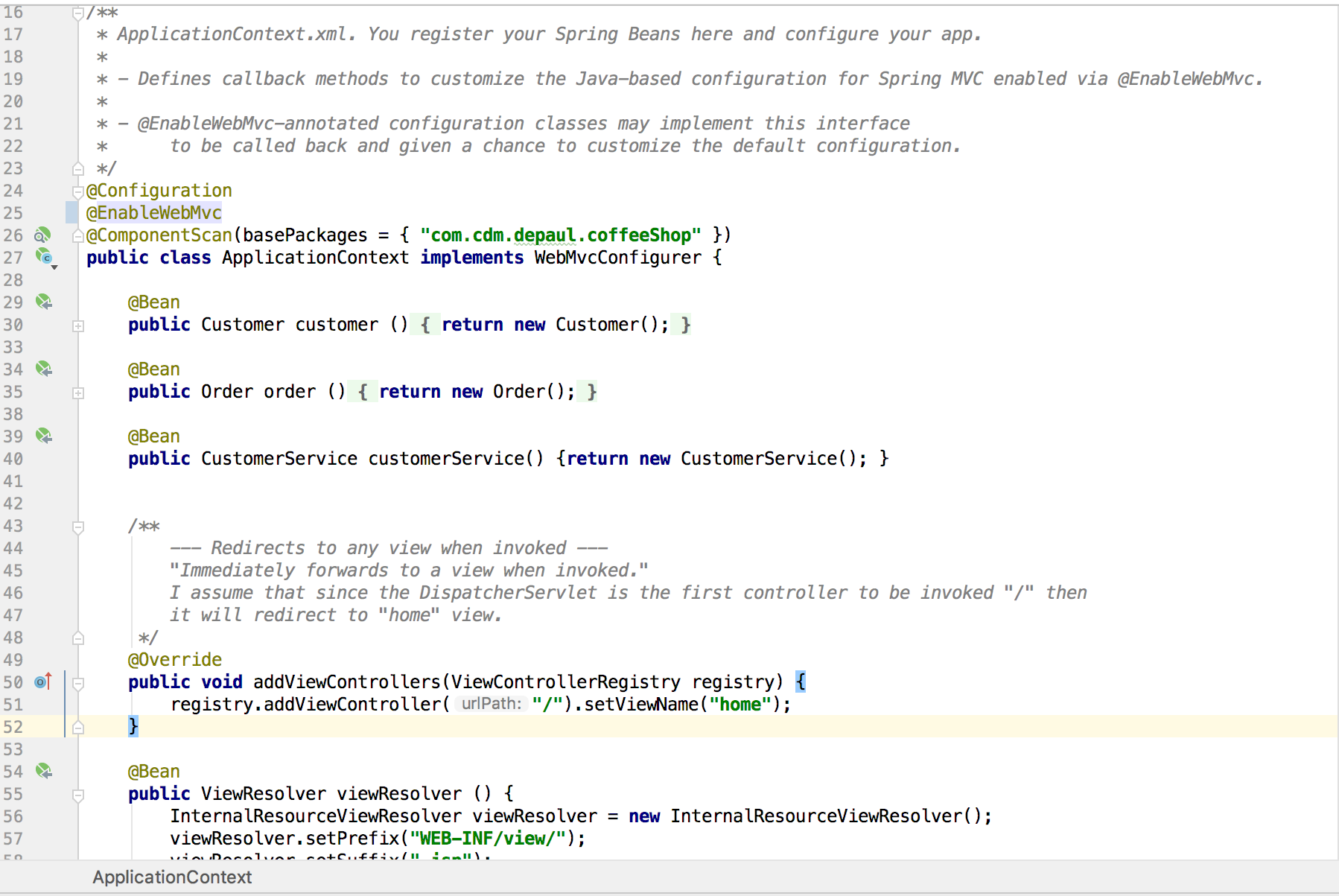
**Lessons Learned Milestone 3:**

* Learned how Servlets get called by a web server to fulfill a request made by the client.
* Learned about the Model View Controller (MVC) design pattern used in web applications.
* Learned how Java Enterprise web applications implemented the Model Viewer Controller initially and how “coupled” it was initially.
* Learned about Spring MVC framework that leverages the implementation of the Model Viewer Controller design pattern while following the same architecture (Servlets, JSP, Application Server).
* Learned about the folder structure of a web application when it is packaged as a WAR project.

# Milestone 3 Screenshot

The following pictures represent the state of my application that is following the standards of the Spring framework.

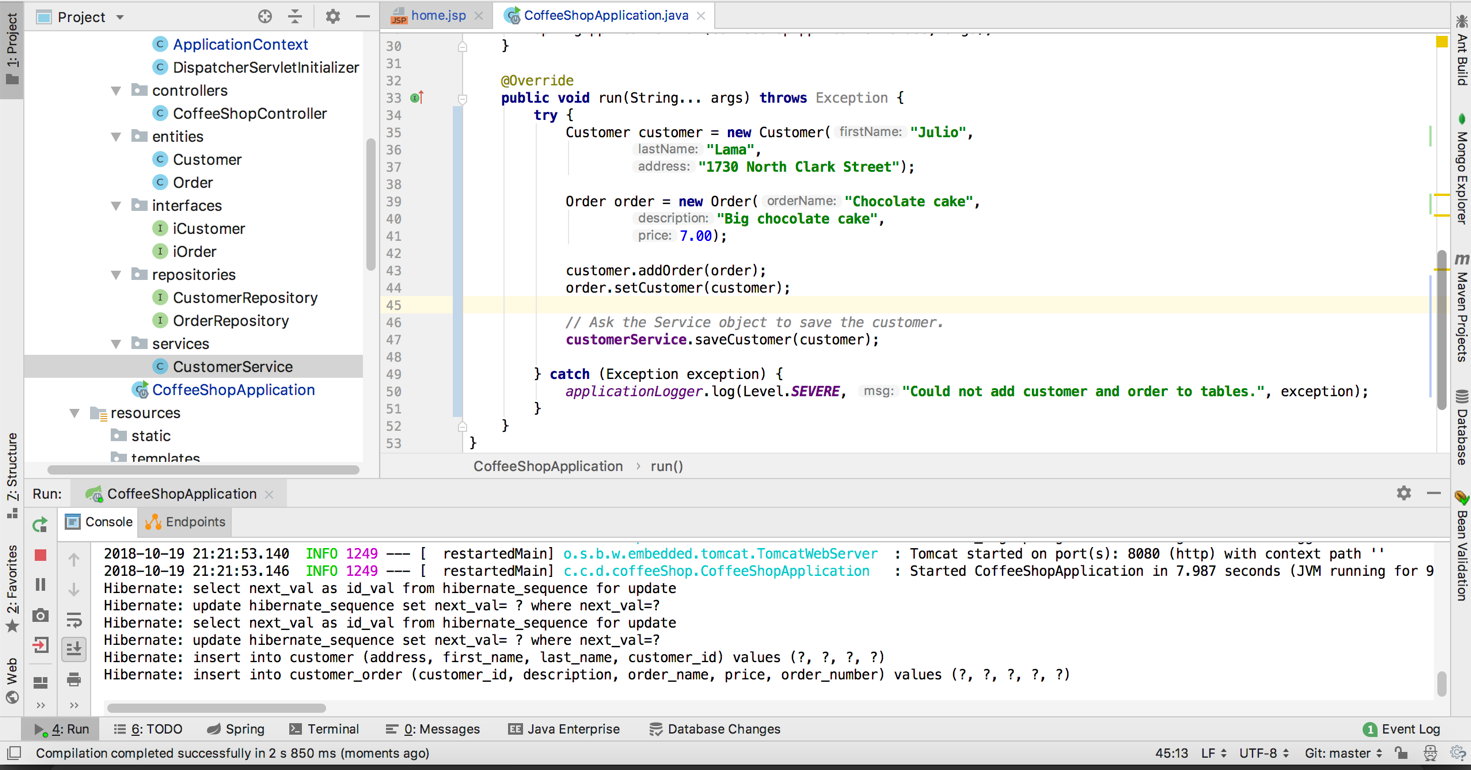
**ApplicationContext / Configuration file (Java configuration way):**



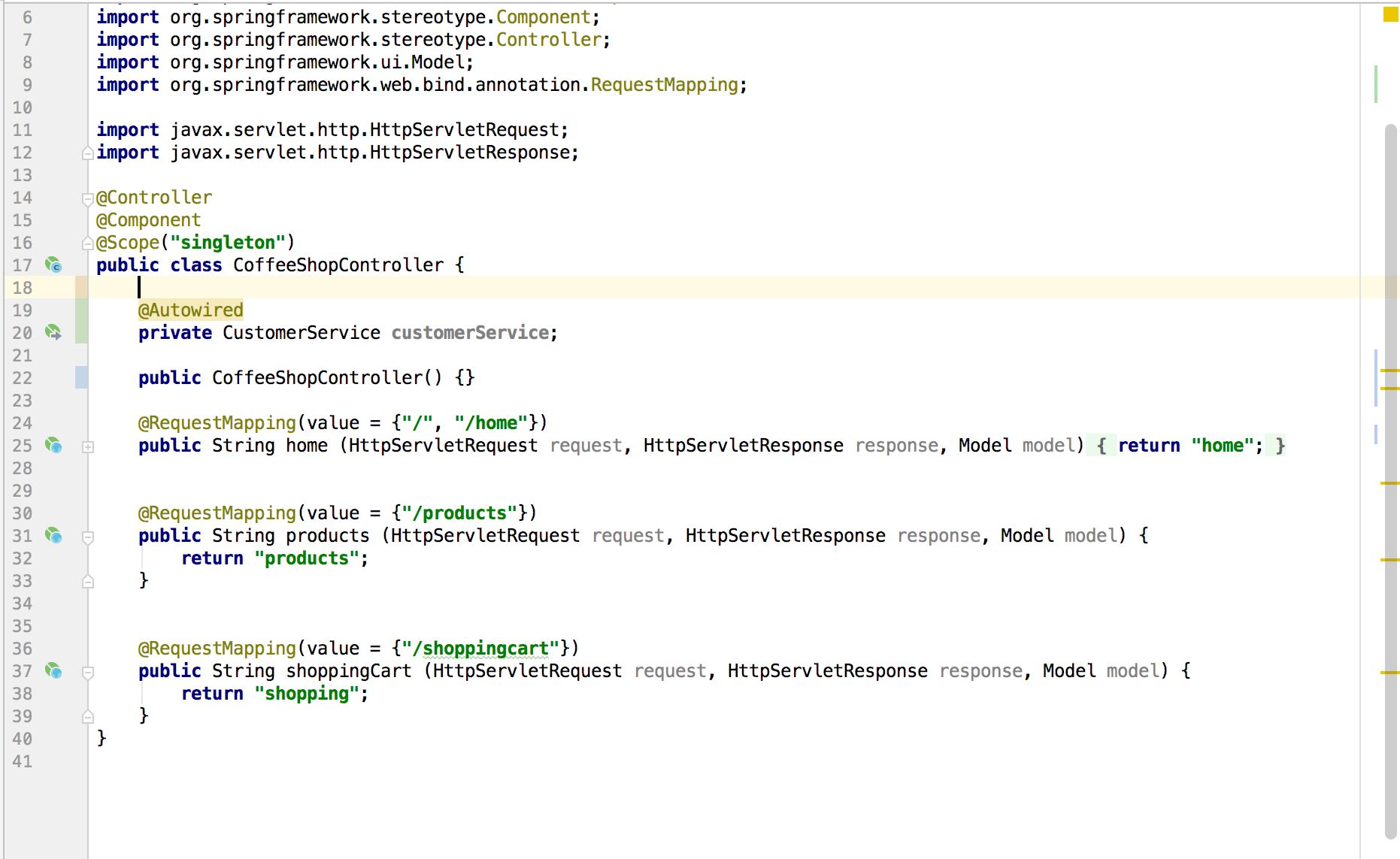
**Front Controller / Dispatcher Servlet Configuration (Java Configuration way):**

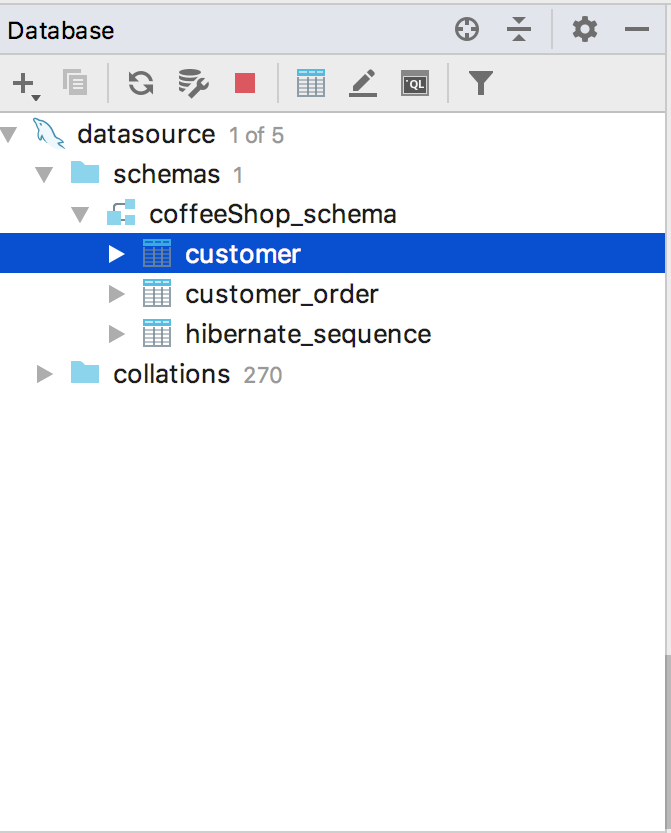
****

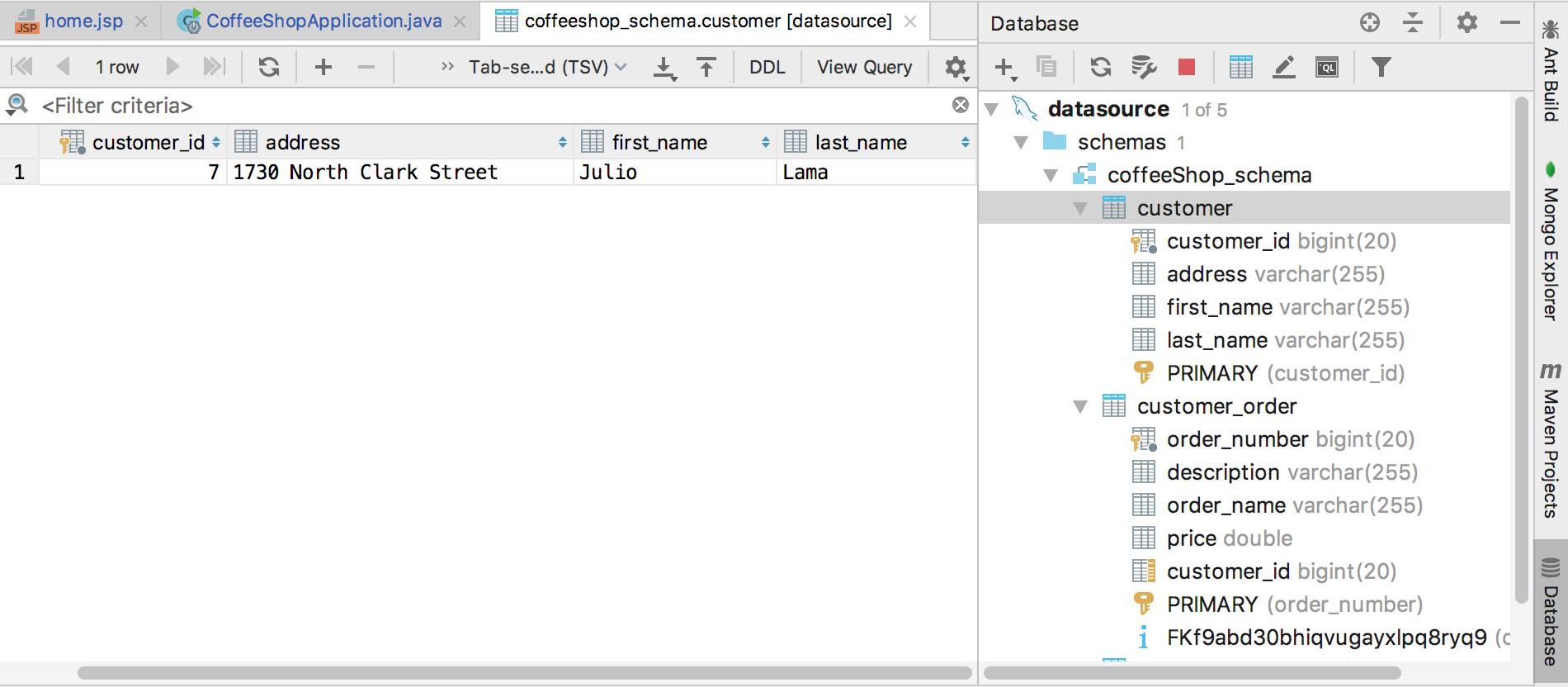
**Inserting Customer and Order to tables using Spring Data JPA:**

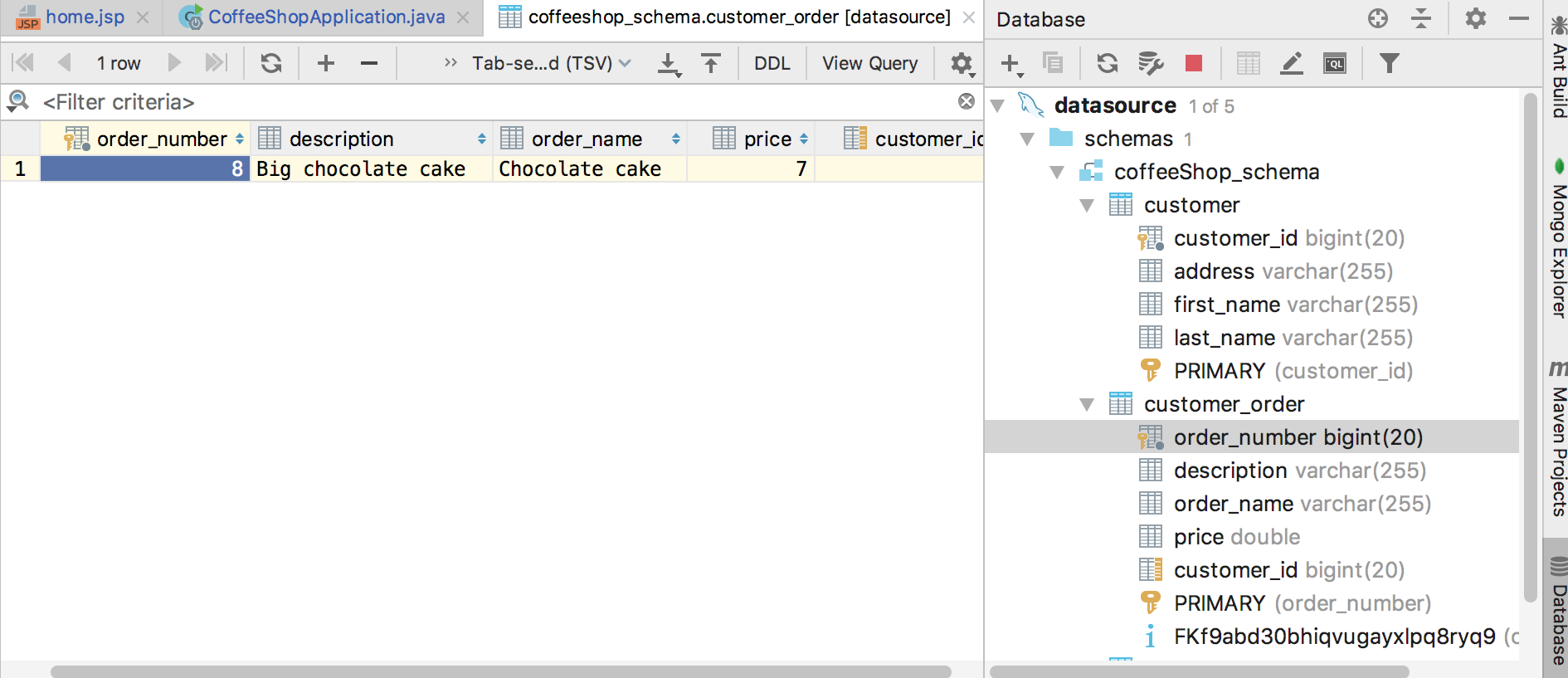


**Controller (one at the moment) state:**

****

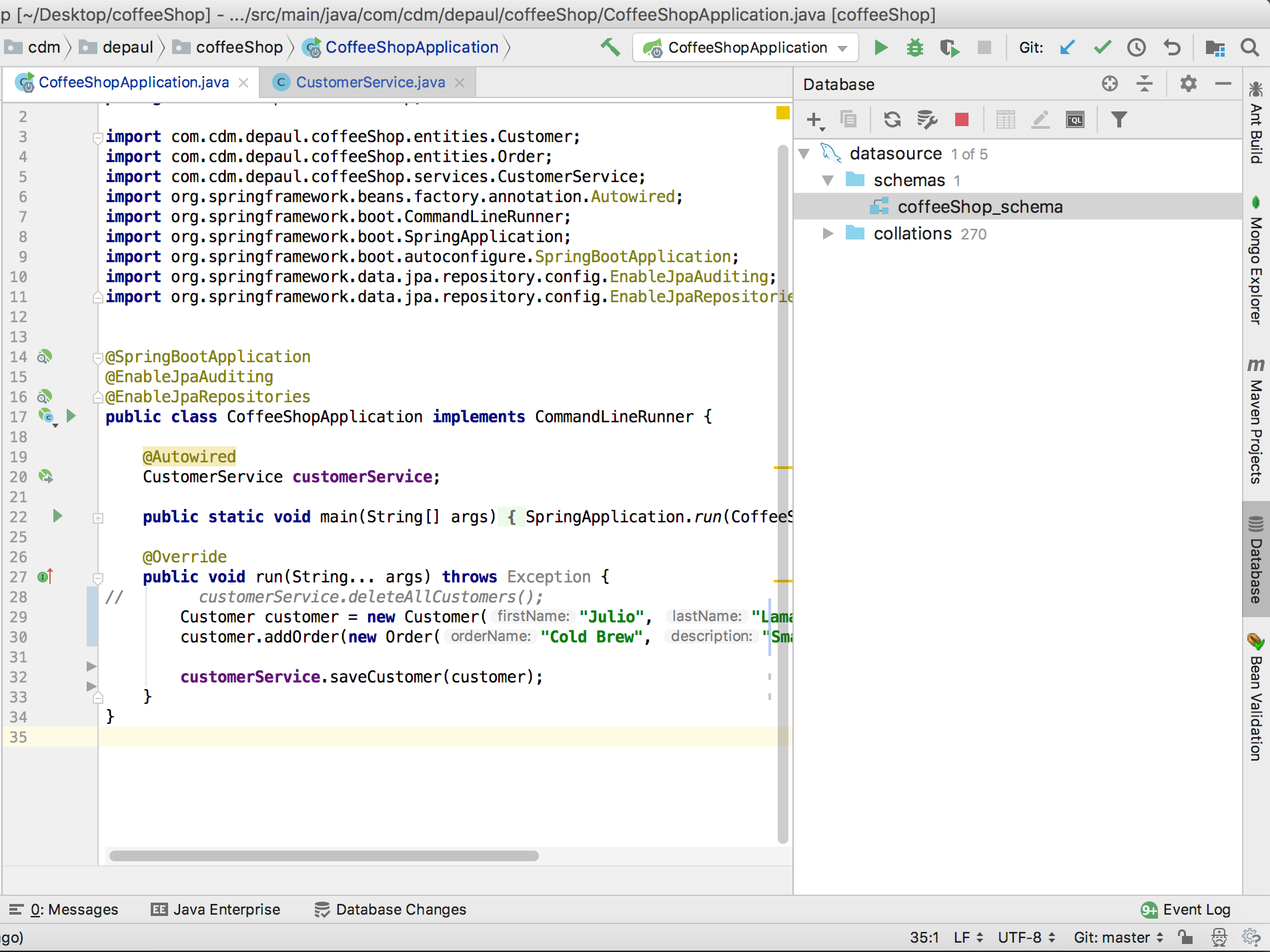




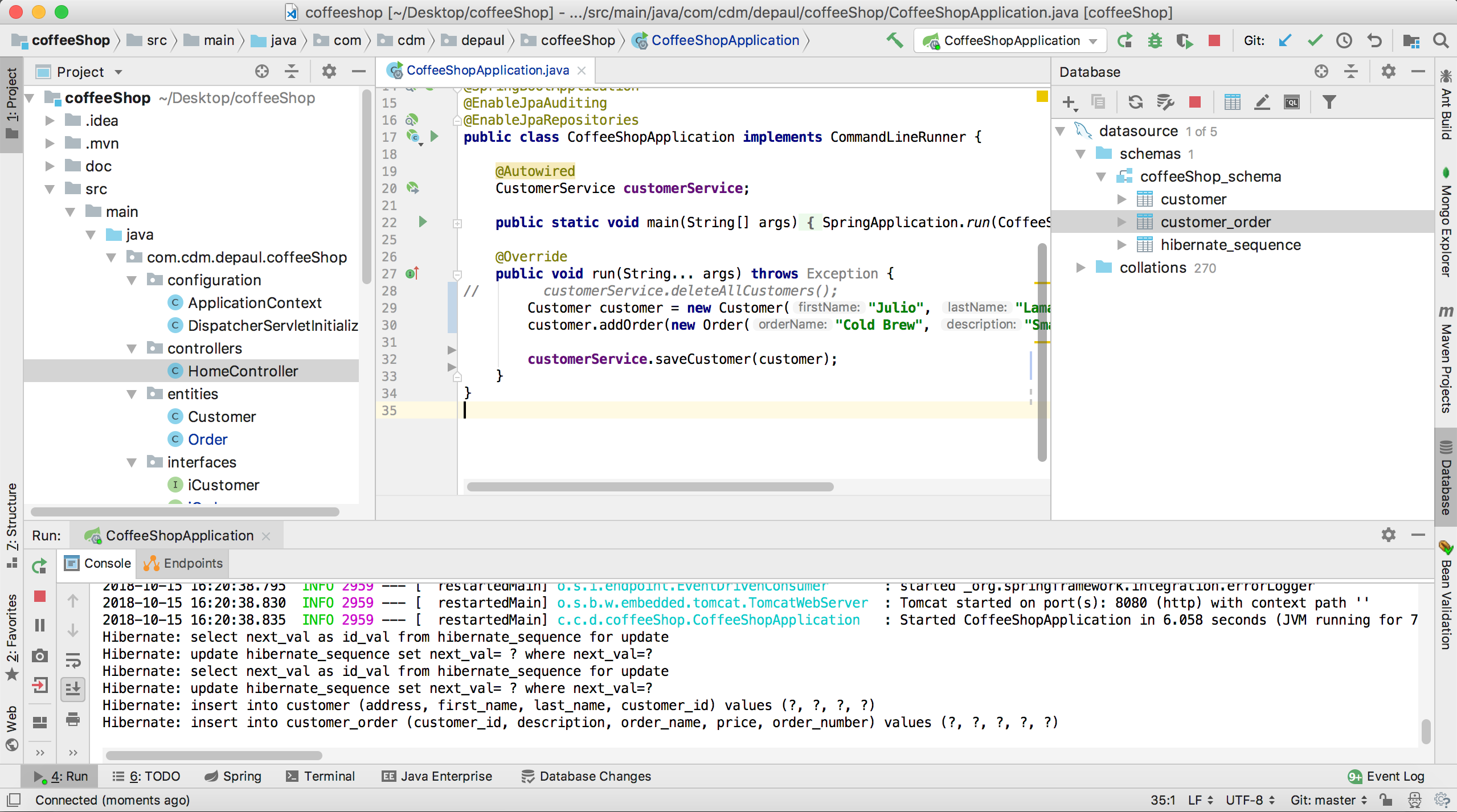


# Milestone 2 Screenshot

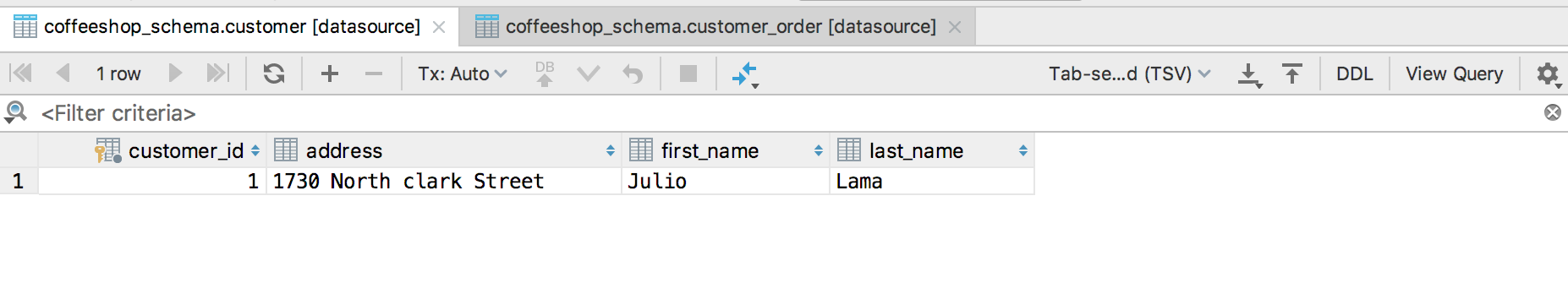
I changed my database to be just a relational database. Check out pictures from Milestone 2 to notice the changes.

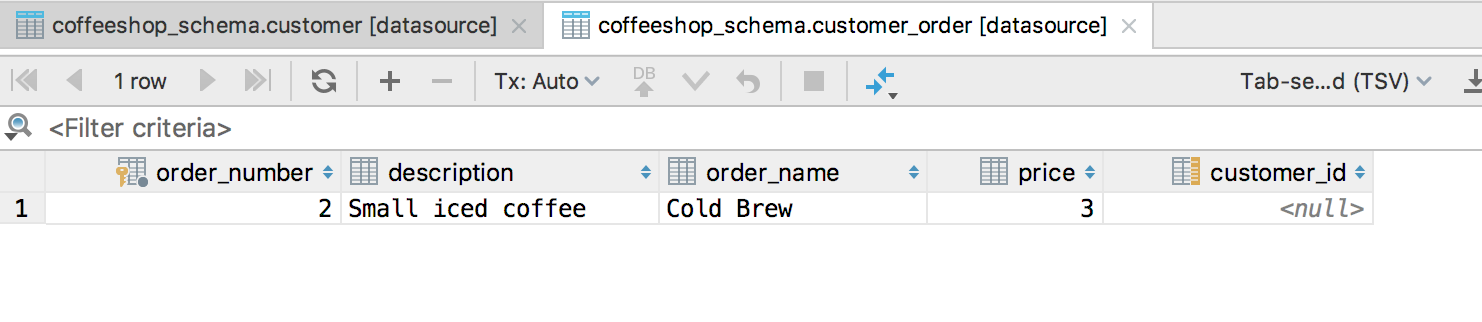
**Database is empty/ 0 collections:**

**Insertion code:**

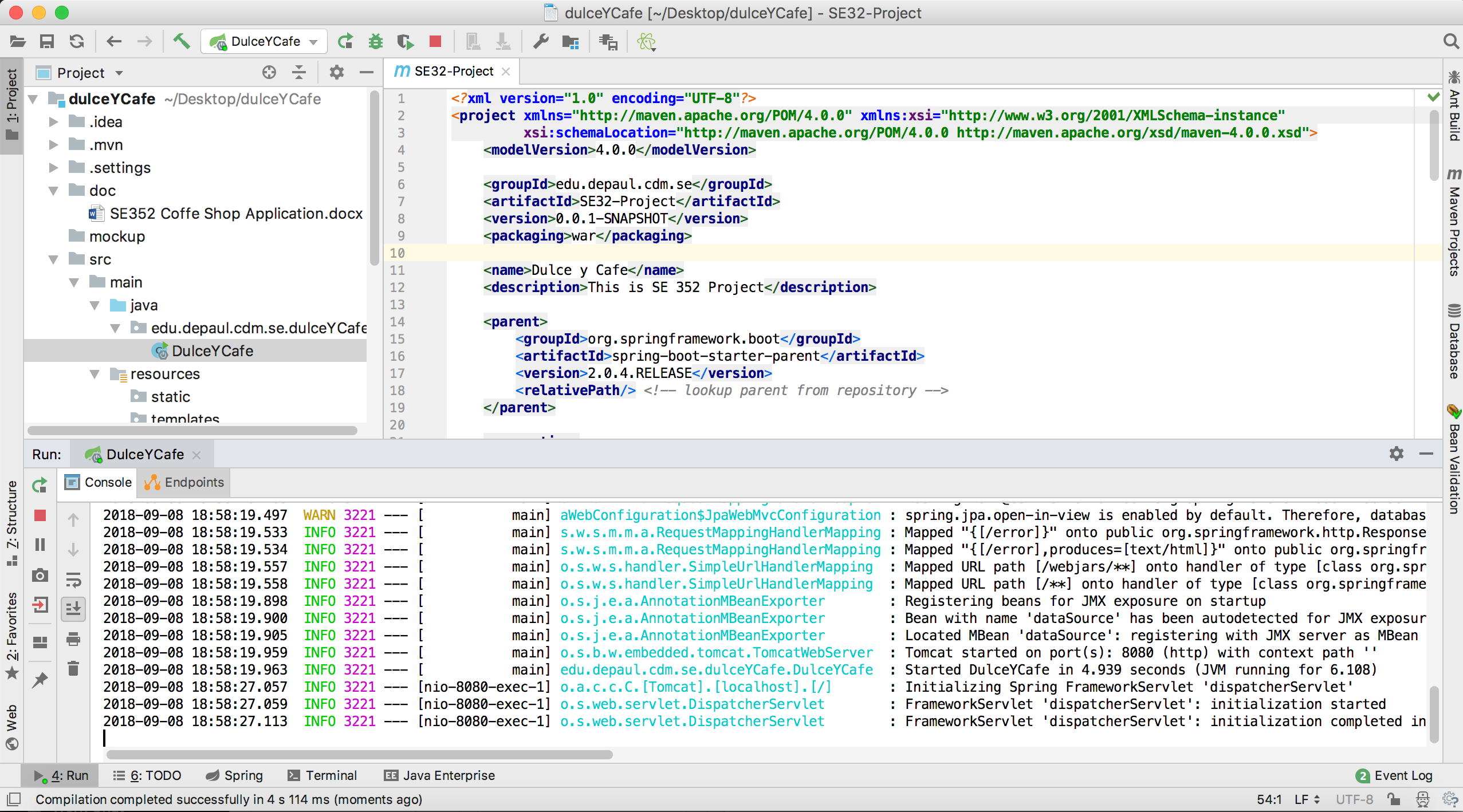
****

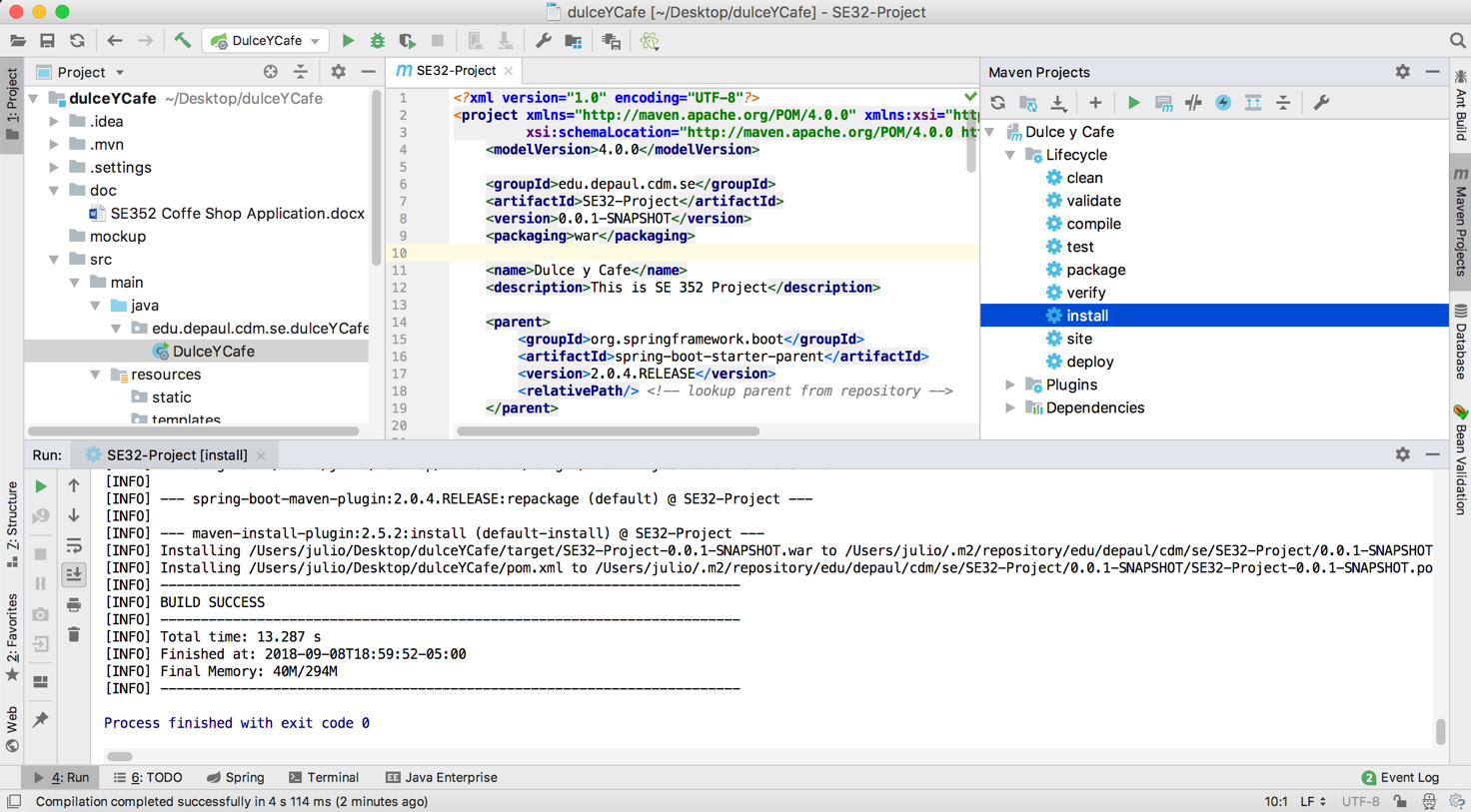
**Tables:**

****



# Milestone 1 Screenshot





# Decision Log

|  |  |  |  |
| --- | --- | --- | --- |
| **Problem** | **What was decided** | **Alternatives considered** | **Rationale** |
| Which IDE to use | IntelliJ IDEA | Eclipse | IntelliJ is a very good IDE |
| Collaboration | Alone | Course provided |  |
| Code repo | Github | Course provided | It is available on the web. |