Sneaker Store

SE352 – Object Oriented Enterprise Application Development

Greatest Chicago sneaker store

Contents

[Overview 2](#_Toc524127111)

[Requirements 2](#_Toc524127112)

[Use Case 2](#_Toc524127113)

[Description of problem 2](#_Toc524127114)

[Design 3](#_Toc524127115)

[Sequence of major functionality 3](#_Toc524127116)

[Web UI (Common case) 3](#_Toc524127117)

[Table layout 3](#_Toc524127118)

[Deployment 3](#_Toc524127119)

[Discussion of how your design met the requirements 3](#_Toc524127120)

[Discussion of lessons learned 4](#_Toc524127121)

[Decision Log 5](#_Toc524127122)

Milestone Screenshot 6

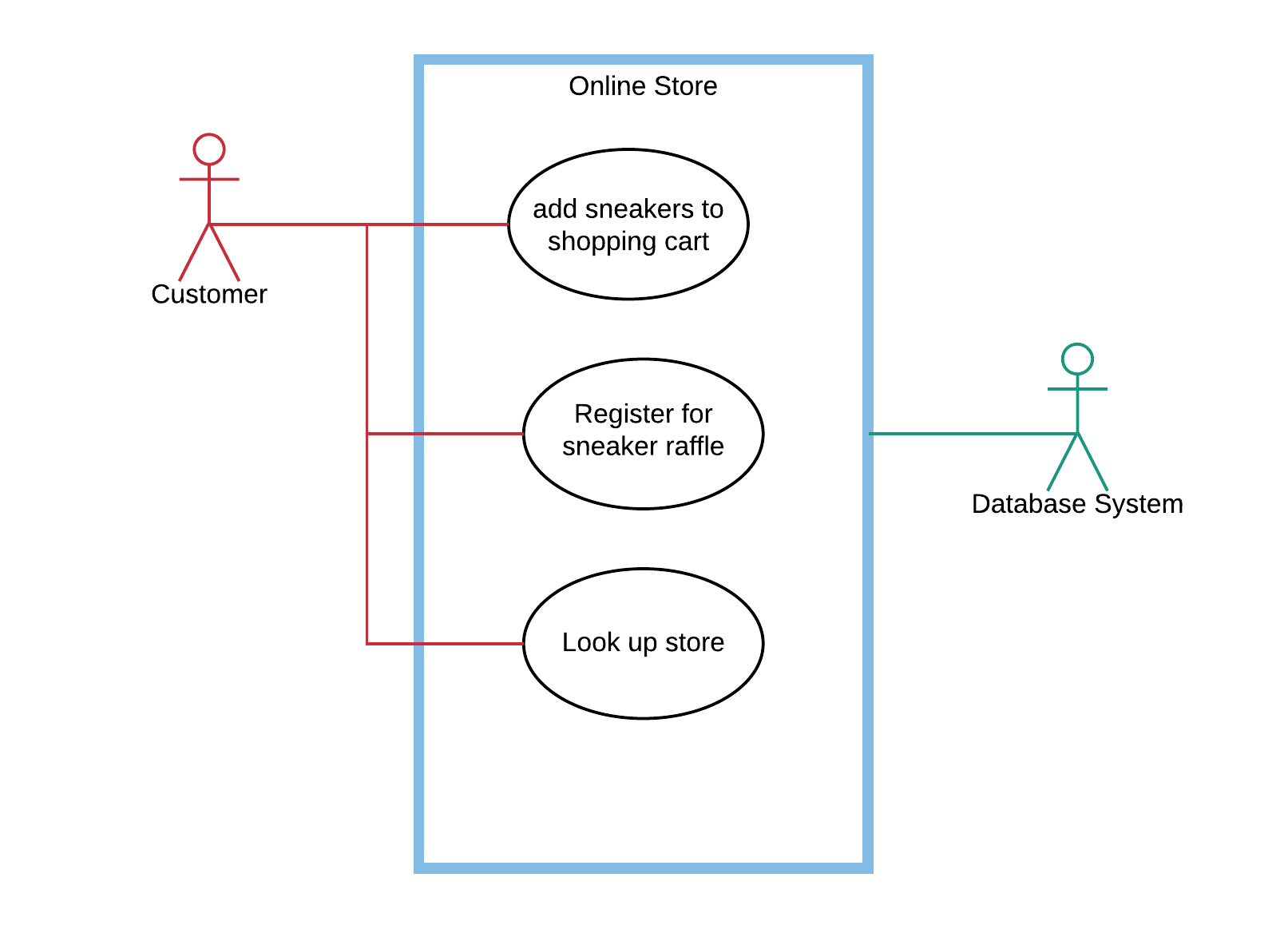
# Overview

Greatest Chicago sneaker online shop that lets the user:

* Register for a sneaker raffle
* Add sneakers to shopping cart
* Look up sneaker stores by zip code

# Requirements

## Use Case



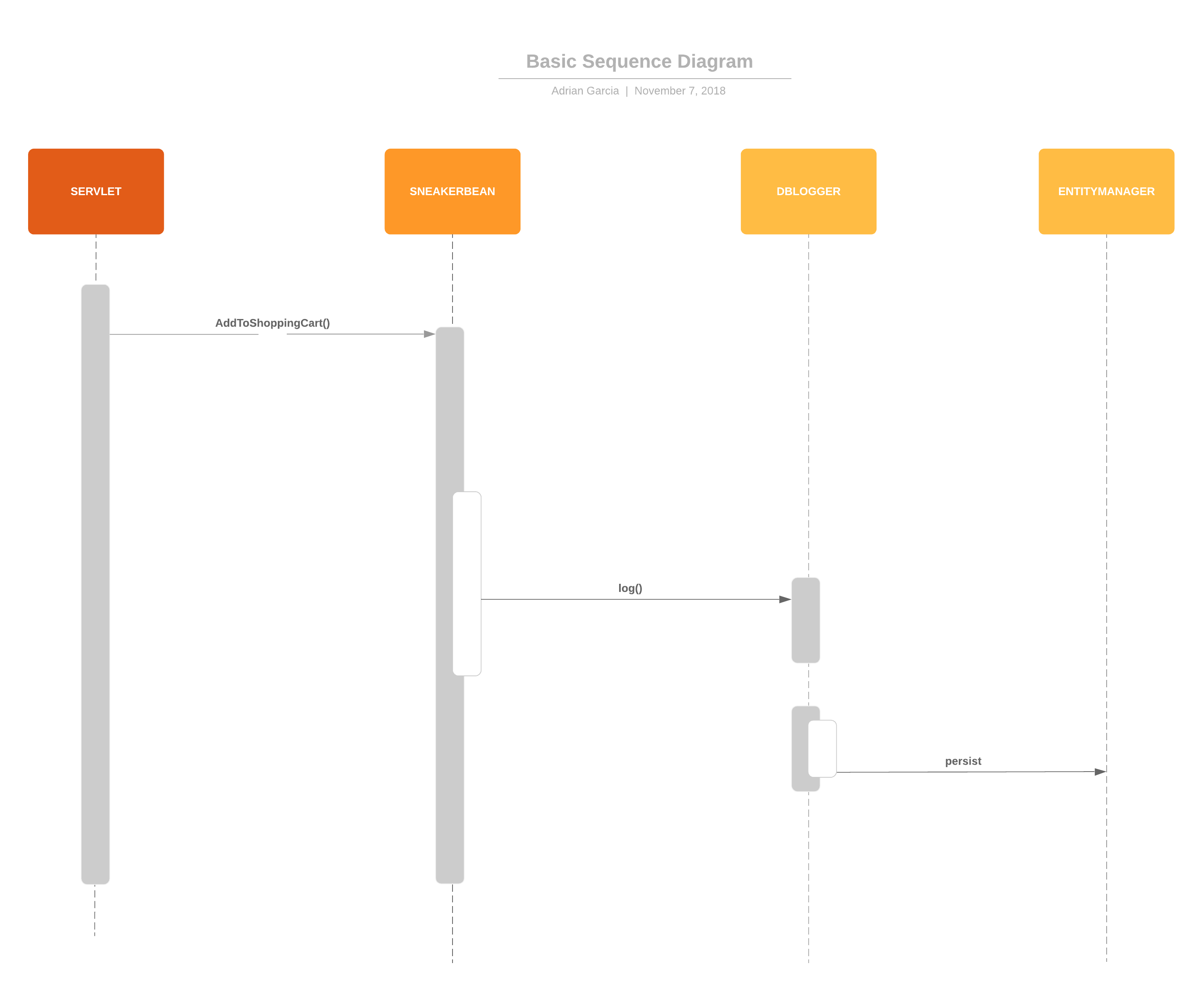
## Description of problem

Sneaker online store that supports human interaction using Web. Users can Register for a sneaker raffle, add sneakers to shopping cart, look up sneakers store by zip code.

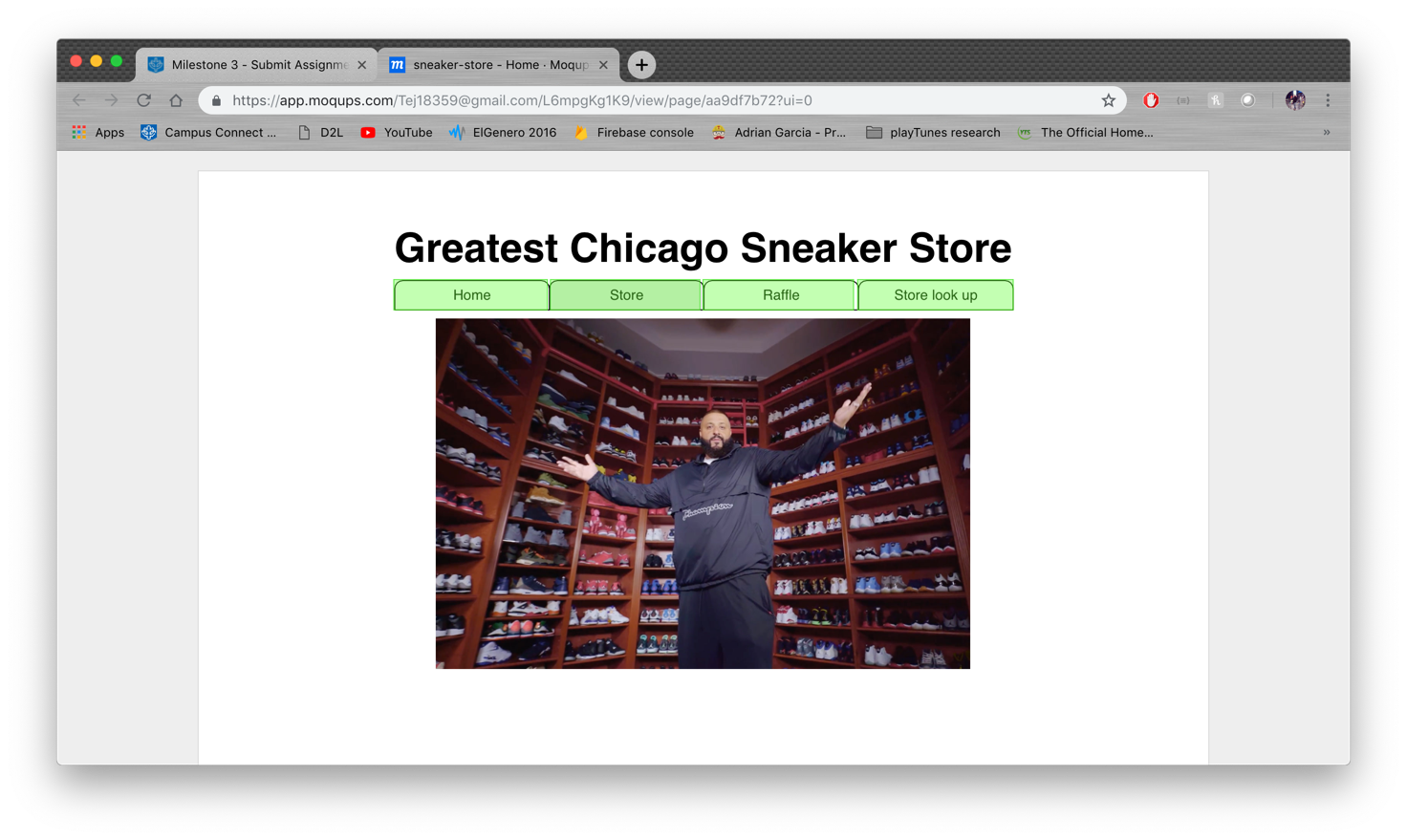
# Design

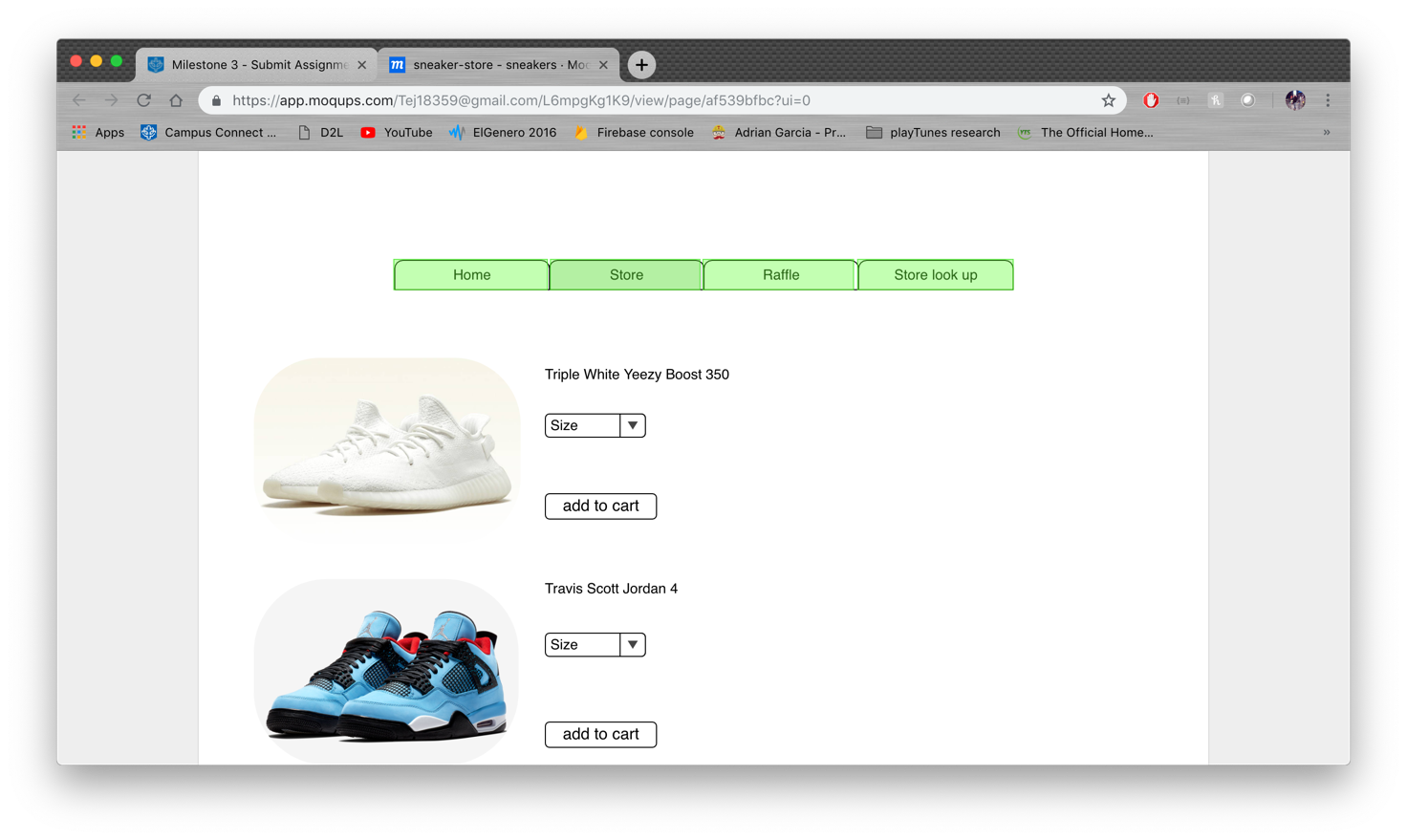
## Sequence of major functionality

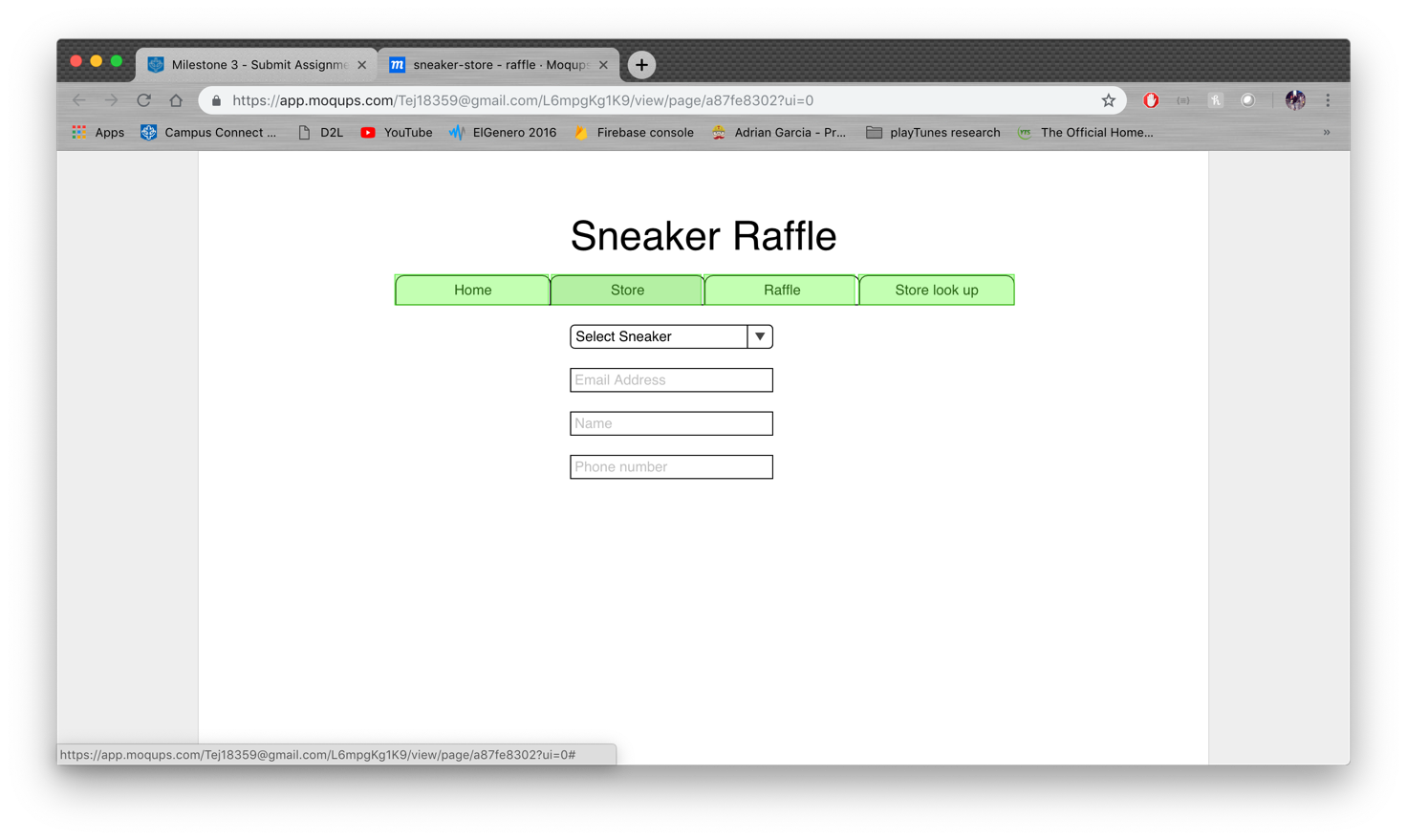
### Web UI (Common case)

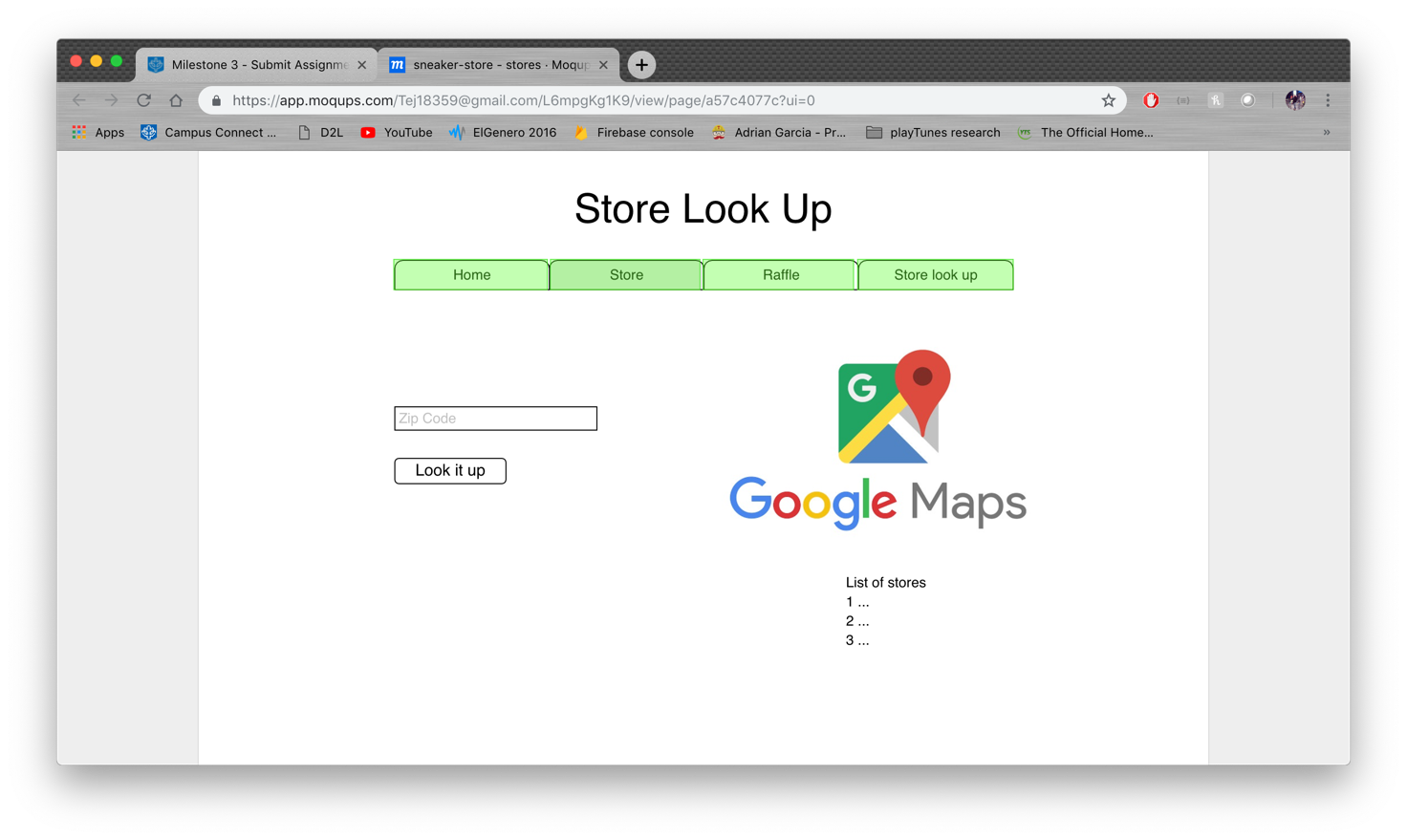


## Web UI Mockup









## Table layout

**NoSQL Document**

Sneaker Store table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Name | Street | City | Zip |
|  |  |  |  |  |
|  |  |  |  |  |

There one document to fulfill the requirement for the registration of the sneaker raffle.

**SQL Tables**

|  |
| --- |
| **Sneaker** |
| Id |
| Name |
| Brand |
| Color |
| Size |
| Type |
| Price |
| Gender |

|  |
| --- |
| **Customer** |
| Raffle Id |
| Shoe Id |
| Name |
| Email |
| Phone number |

There are two tables to fulfill the requirement for the registration of the sneaker raffle.

Sneaker table

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Name | Brand | Color | Size | Type | Price | Gender |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

Shopping Cart

|  |  |  |
| --- | --- | --- |
| Order number | Sneaker Id | Quantity |
|  |  |  |
|  |  |  |

There are only two tables that fulfill the requirement of adding sneakers to a shopping cart

## Deployment

All the classes were packaged into one deployment unit to keep things simple.



Queue: CalculatorQ

Data source: jdbc/sample

# Discussion of how your design met the requirements

This sample project did not meet the functional requirements but was created to validate the component interaction for different technology pieces.

This also serves as a reference document for other projects of basic expectation of the course project work

# Discussion of lessons learned

**SQL Database decision**

I made the decision of using a SQL database for the shopping cart functionality and for the store finder. The reason why I decided of using the SQL database for shopping cart was to ensure consistency. On the same note with the sneaker raffle it will allow to join the sneaker table with the customer table, this will find all the customer information with the specific shoe they fill the raffle ticket for.

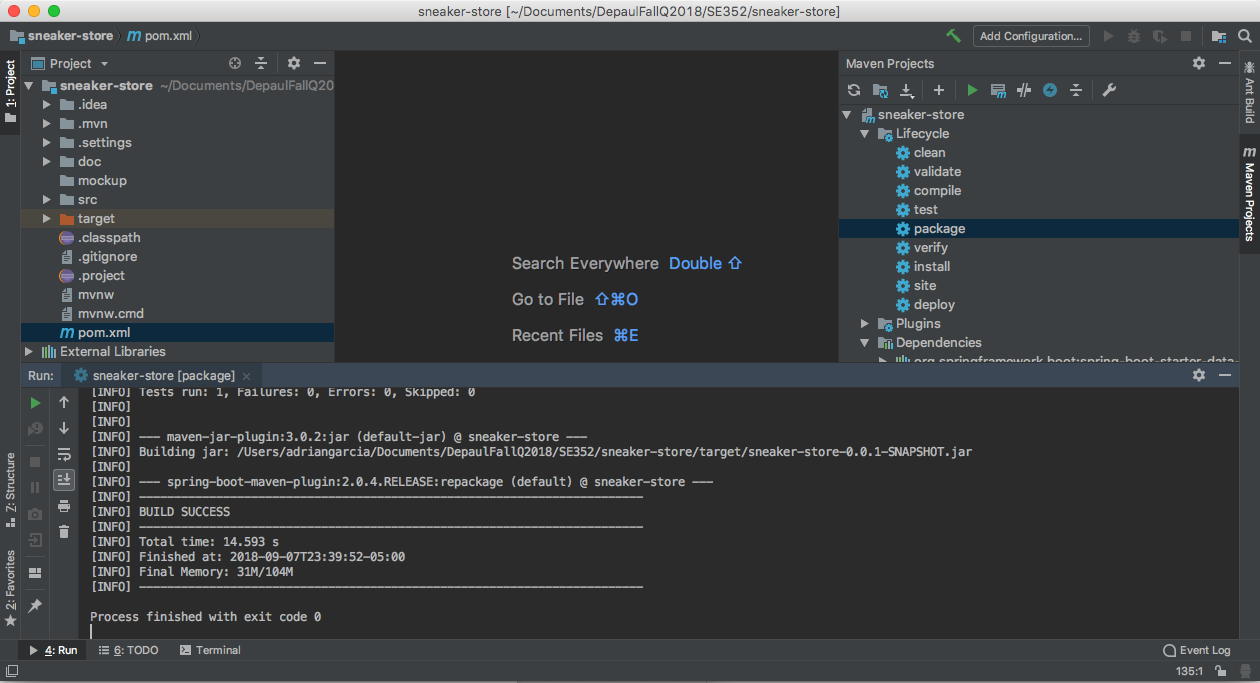
**NoSQL Decision**

I made the decision of using mongoDB as my database for the feature of store look up for the reason that it would be quick if the user just passes a zip code, it can find the documents that have the same zip code O(n) time.

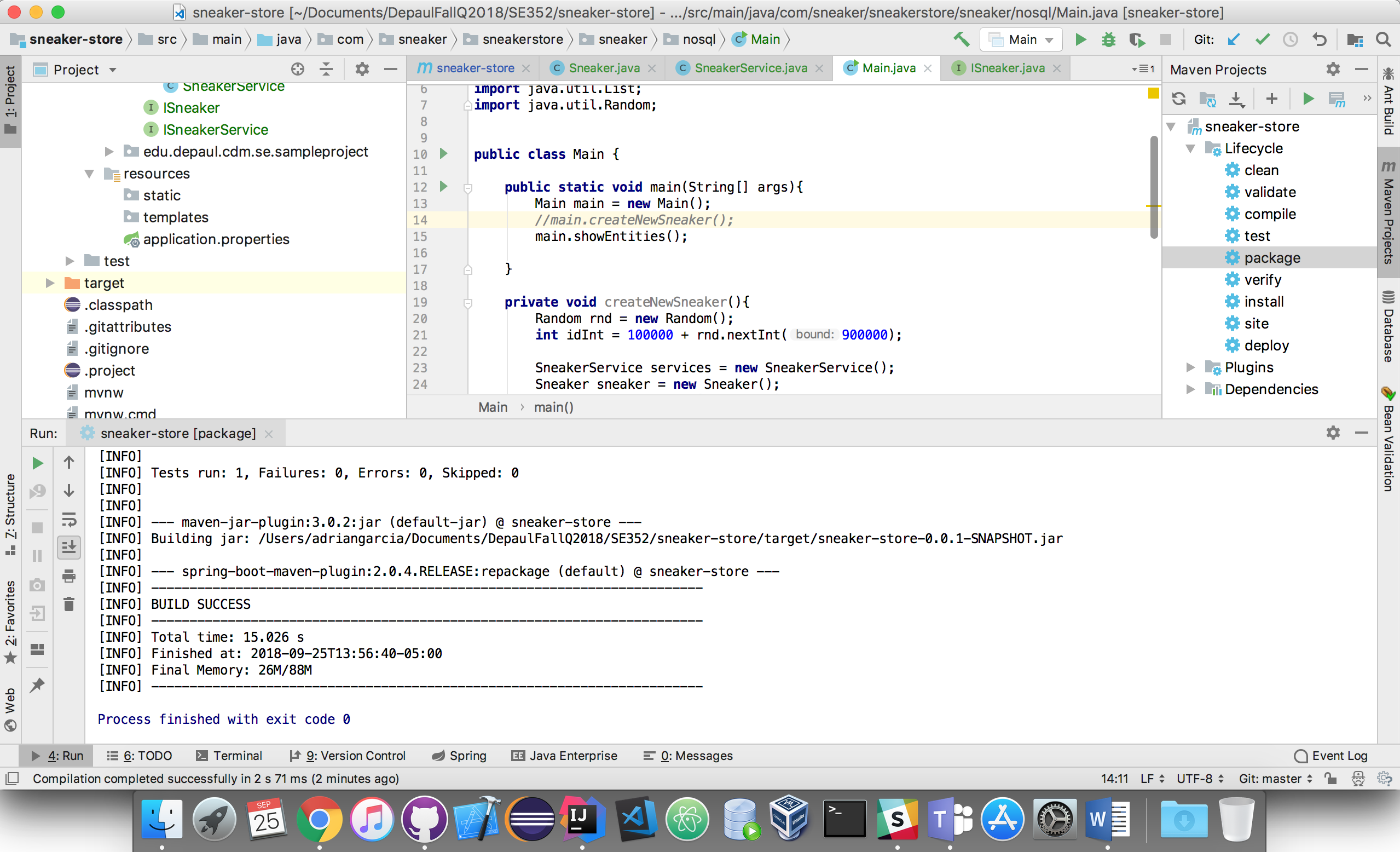
# Decision Log

|  |  |  |  |
| --- | --- | --- | --- |
| **Problem** | **What was decided** | **Alternatives considered** | **Rationale** |
| Which IDE to use | IntelliJ | Eclipse | IntelliJ had all the components integrated in one UI |
| SQL Database | PostgreSQL | MySQL | Open source and I’ve worked with PostgreSQL |
| NoSQL Database | MongoDB | Fire Base | Fast access to Database and can modify the data from database UI |
| Code repo | GitHub | Local Machine | I would be able to have access to the code in any computer |
|  |  |  |  |
|  |  |  |  |

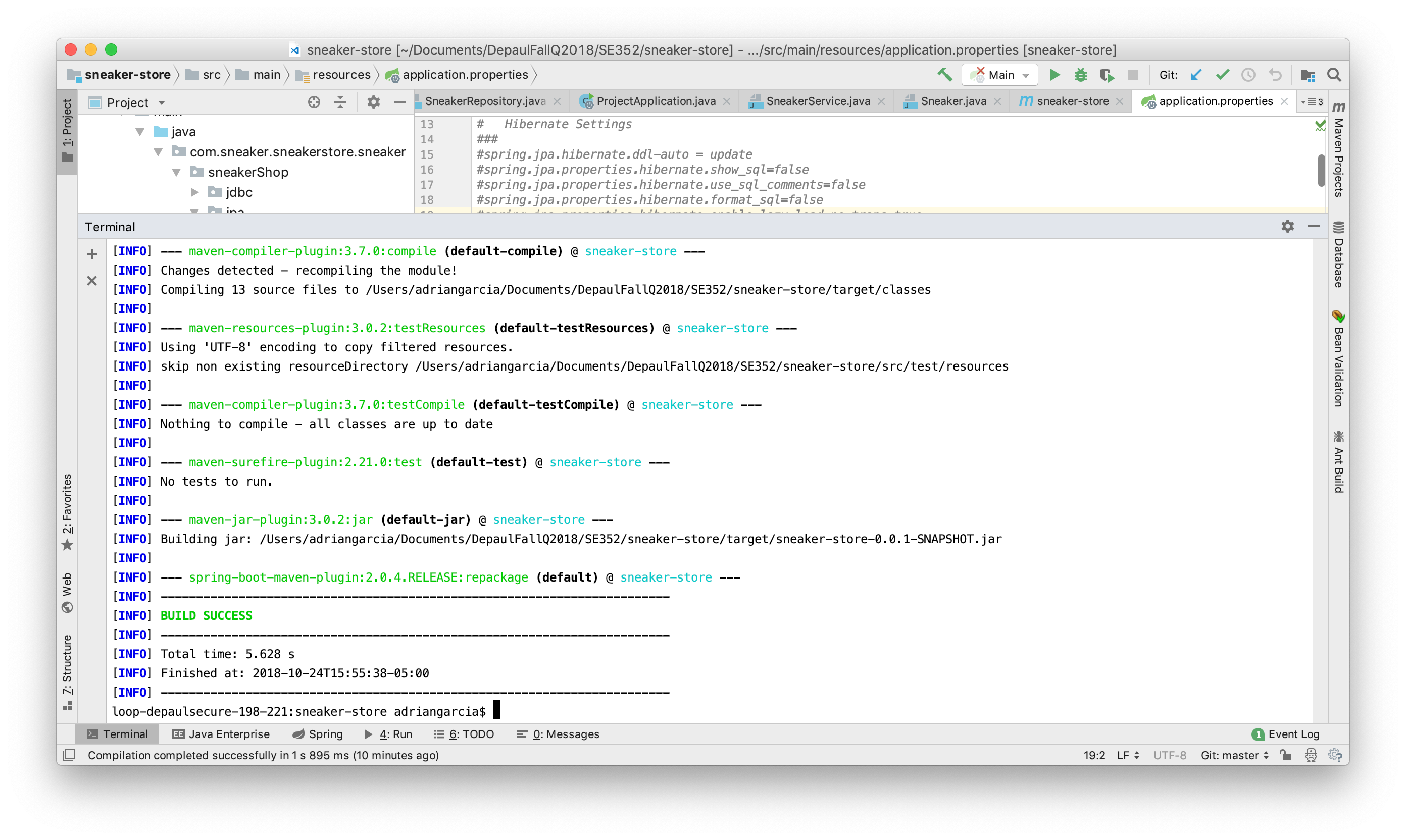
# Milestone 1 Screenshot



# Milestone 2 Screenshot



# Milestone 3 Screenshot



# Milestone 4 Screenshot

