Twiss, a car sharing application

Analysis and Design Document

Student: Andrei Gog

**Group: 30433**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <15/04/2018> | <1.0> | Updated the **Elaboration – Iteration 1.1** section of the document. | Andrei Gog |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

I. Project Specification 4

II. Elaboration – Iteration 1.1 4

1. Domain Model 4

2. Architectural Design 4

2.1 Conceptual Architecture 4

2.2 Package Design 4

2.3 Component and Deployment Diagrams 4

III. Elaboration – Iteration 1.2 4

1. Design Model 4

1.1 Dynamic Behavior 4

1.2 Class Design 4

2. Data Model 4

3. Unit Testing 4

IV. Elaboration – Iteration 2 4

1. Architectural Design Refinement 4

2. Design Model Refinement 4

V. Construction and Transition 5

1. System Testing 5

2. Future improvements 5

VI. Bibliography 5

# Project Specification

# 

# Elaboration – Iteration 1.1

# Domain Model

The application should have multiple classes in order to store the data gathered from the database and in order to store new data in the database. Thus we will need classes like User which will manipulate regular and common information about any user, also Driver and Passenger which will extend the regular user class. The application will have the applet class which will be used to create the UI of the application that will be accessible through the browser, as well as some way to connect to the database server and manage data.

# Architectural Design

## Conceptual Architecture

The system will use the Client-Server Design-Pattern. This architectural design is a a [distributed application](https://en.wikipedia.org/wiki/Distributed_application) structure that partitions tasks or workloads between the providers of a resource or service, called [servers](https://en.wikipedia.org/wiki/Server_(computing)), and service requesters, called [clients](https://en.wikipedia.org/wiki/Client_(computing)). Often clients and servers communicate over a [computer network](https://en.wikipedia.org/wiki/Computer_network) on separate hardware, but both client and server may reside in the same system. A server [host](https://en.wikipedia.org/wiki/Host_(network)) runs one or more server programs which share their resources with clients. A client does not share any of its resources, but requests a server's content or service function.

This architecture was chosen because of the need of multiple people using the application at once. Thus, the server will contain the database for the application’s information storing through the business logic.

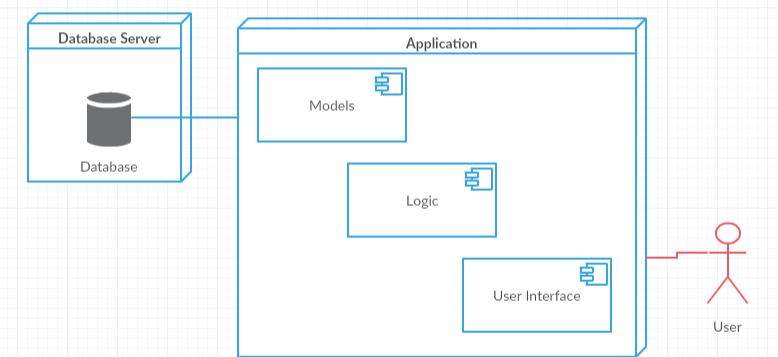
## Package Design

## Component and Deployment Diagrams

* **Deployment Diagram**

# 

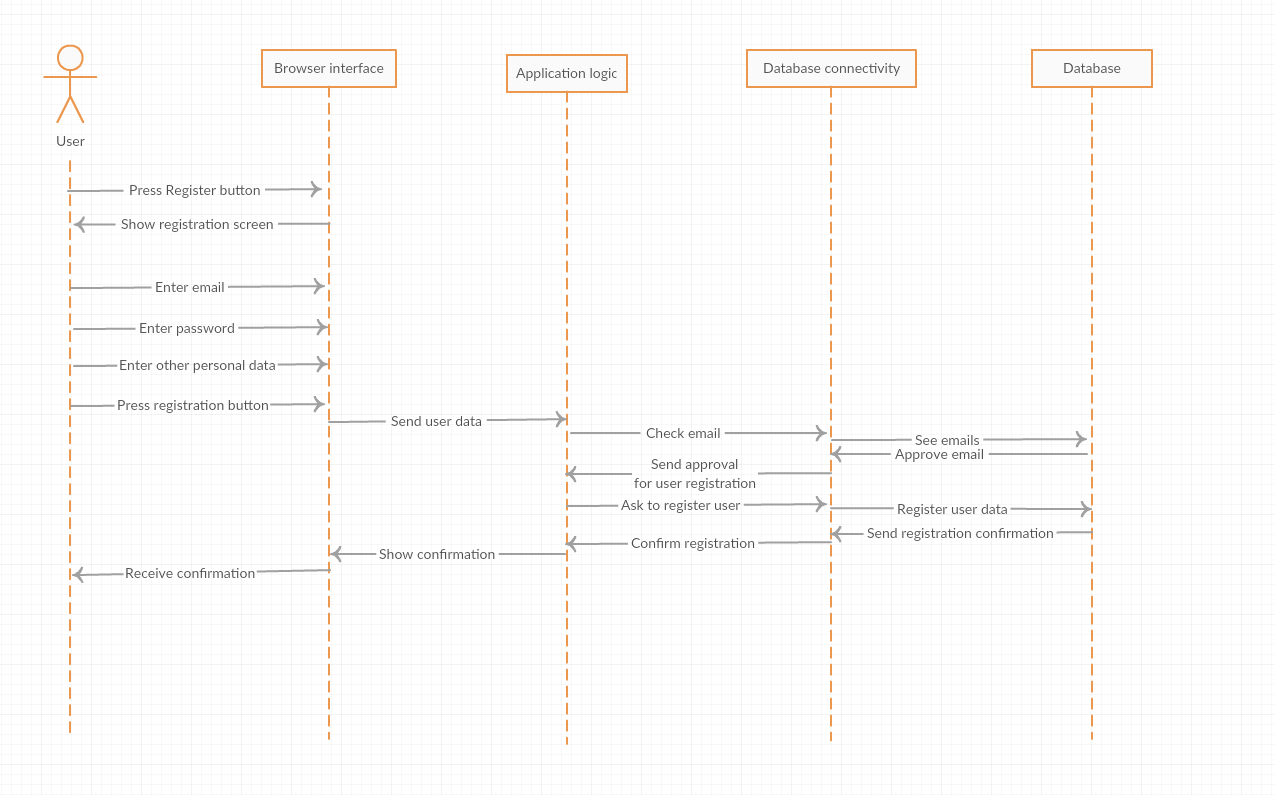
* **Component Diagram**

****

# Elaboration – Iteration 1.2

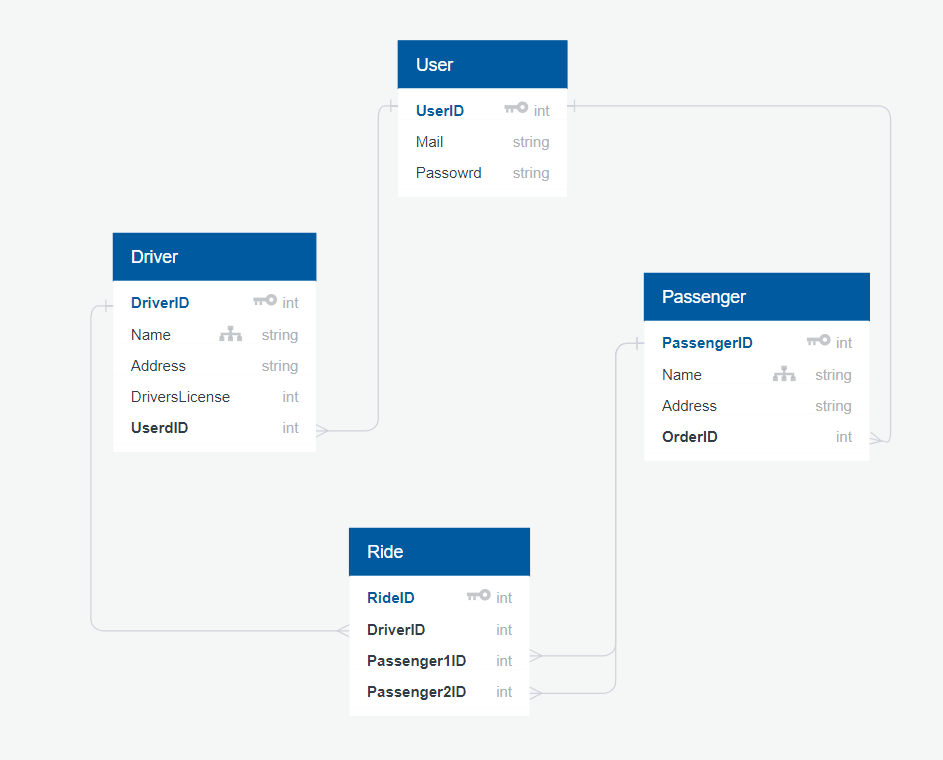
# Design Model

## Dynamic Behavior

**

## Class Design

# Data Model



# Elaboration – Iteration 2

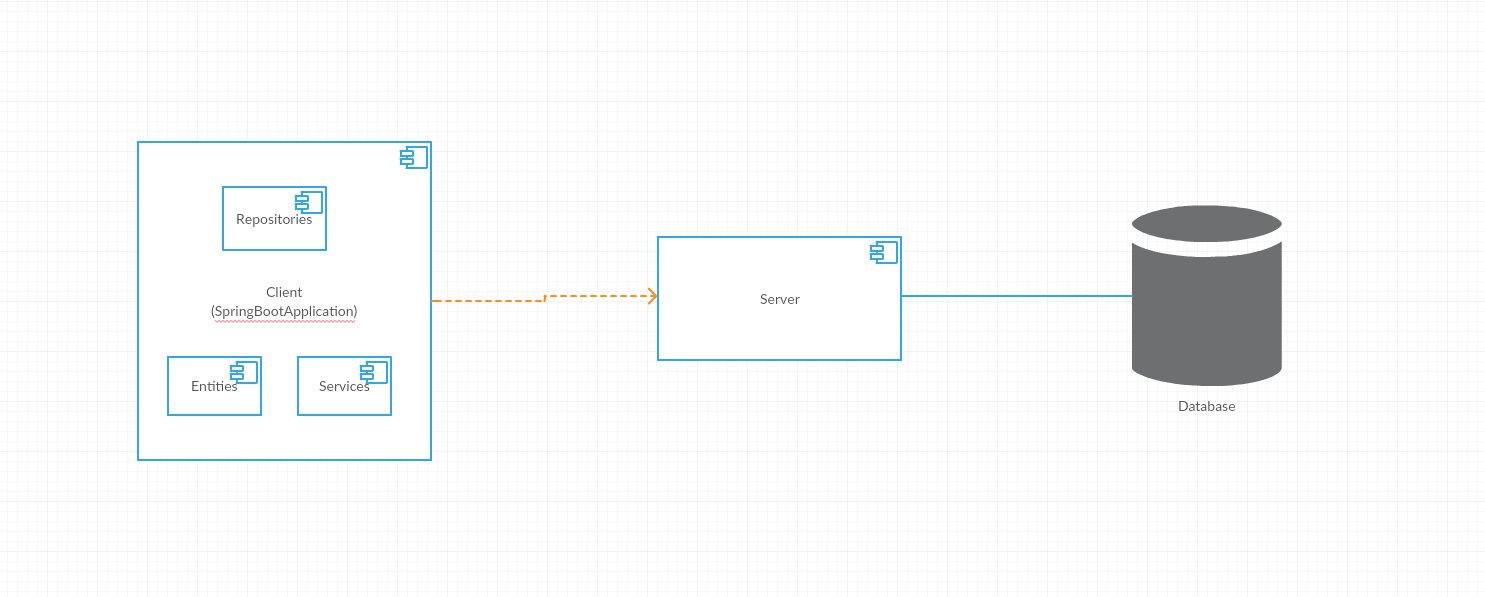
# Architectural Design Refinement

**Conceptual architecture**

The system will be modeled after the Client-Server Architecture Design. This architectural design is a a [distributed application](https://en.wikipedia.org/wiki/Distributed_application) structure that partitions tasks or workloads between the providers of a resource or service, called [servers](https://en.wikipedia.org/wiki/Server_(computing)), and service requesters, called [clients](https://en.wikipedia.org/wiki/Client_(computing)). Often clients and servers communicate over a [computer network](https://en.wikipedia.org/wiki/Computer_network) on separate hardware, but both client and server may reside in the same system. A server [host](https://en.wikipedia.org/wiki/Host_(network)) runs one or more server programs which share their resources with clients. A client does not share any of its resources, but requests a server's content or service function. This architecture was chosen because of the need of multiple people using the application at once. Thus, the server will contain the database for the application’s information storing through the business logic.

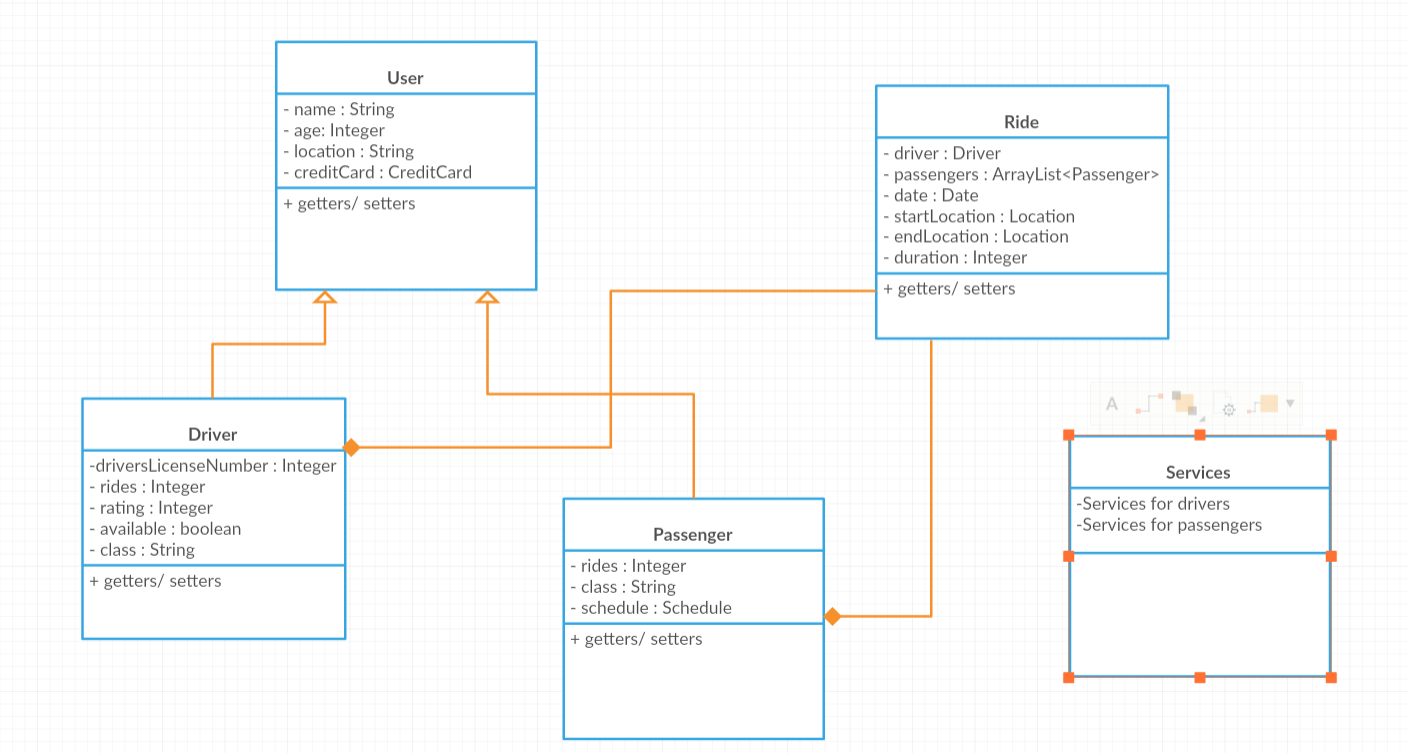
The only thing that is to add to this description of the conceptual architecture is the fact that the application will be modeled using the Spring Framework. This means that the application will automatically be a client-server one. While the Server part will be automatically created and handled by the Spring Framework, the client part will be created by the designer and is modeled using the Spring MVC (integrated MVC architecture in the Spring Framework). The View part of the application is created using Thymeleaf with HTML files.

**Component Diagram**



# Design Model Refinement

## 



This is the updated UML Class Diagram.

# Construction and Transition

# System Testing

Since the system it is still at its first version the only testing that has been done is the human testing, being exposed to different scenarios like the Use Case scenarios and the output was as the expected one. In future developments system testing will be done using frameworks like Mokito.

# Future improvements

Twiss is a freshly developed platform and it has enough room for various improvements and new modules. Some of the main ideas are integrating a platform for online payment, a chat for a better communication between the driver and the passenger. Also for a better and accurate location it would have been a proper solution to use the Google Maps API.

# Bibliography

[1] <https://en.wikipedia.org/wiki/Client%E2%80%93server_model>