BusPlanner	Version: 1.1
Design Description	Date: 11-11-2016

Distributed **S**oftware **D**evelopment:
BusPlanner
Design Description

Revision History

Date	Version	Description	Author
11-11-2016	1.0	Initial draft	Team
04-01-2017	1.1	Changes in technologies used, database model removed	I. Agosti

Contents

1 INTRODUCTION

1.1 Purpose of this document

The purpose of this document is to give an overview of the project design i-e functionalities, Class diagram, Software architecture, Graphical User Interface and Database Model.

1.2 Document organization

The document is organized as follows:

- Section 1, Description and definitions describes the content of the design document.
- Section 2, Background and objectives describes the purpose of the project and the algorithm used for reducing the number of buses per route and the time needed to do a scheduling.
- Section 3, High level system architecture describes the system's main components such as MySQL, the BusPlanner application and the user simulation.
- Section 4, Software architecture describes the system's architecture layers (presentation layer, business logic layer and data access layer) and the technologies used to implement them (HTML, CSS, Bootstrap, PHP and MySQL).
- Section 5, Graphical User Interface presents the web pages models the users will interact with.
- Section 6, Database model describes the format of the tables needed to store and retrieve data.

1.3 Intended Audience

The intended audience of this document is composed of:

- Development team, as a guidance line during development and also for the team to ensure that they have the same level of understanding on the project design, architecture and technologies used.
- Stakeholders, whose needs will be taken under consideration in the design and architecture of the project.

1.4 Scope

This document will provide an overview about the project design, architecture and technologies used during the development of the project. It also presents the database model. The document will be a guideline for the development team during the development process, saving them a lot of time.

1.5 Definitions and acronyms

1.5.1 Definitions

Keyword	**Definitions**
User	A person who requests for bus by being from bus stop.
Fleet Manager	Who owns the buses. He/she wants to know the utilization of buses and scheduling of buses.
User Request	Information generated with timestamps for the scheduling purpose.
Algorithm	A method used to enhance the scheduling process which is static as well as dynamic.
Sprint	A repeatable work cycle which is also known as iteration.

1.5.2 Acronyms and abbreviations

Acronym/abbreviation	**Definitions**
UI	User Interface
GUI	Graphical User Interface
MDH	Mälardalens Högskola, Västerås, Sweden
POLIMI	Politecnico di Milano, Milan, Italy
QA	Quality Assurance
DSD	Distributed Software Development

2 BACKGROUND AND OBJECTIVES

2.1 Overview

The purpose of this project is to help the city of Johannesburg with the bus planning process which, as of now, lacks of efficiency and effectiveness.

For more information: http://www.fer.unizg.hr/_download/repository/Project_Plan%5B12%5D.pdf

2.2 High level description of the functionalities

The BusPlanner project is based on an algorithm that simulates user requests around the city of Johannesburg. These requests are identified by the two bus stops where the user wants to get on and off the bus. Based on this information, the algorithm is able to identify which route reaches both the user's starting and end point, and then assign the user to the bus already covering that route or assign a new bus to that route if the one already covering it is full.

In this way the process efficiency will be improved and the time needed to do a scheduling will be reduced. Also, the users' waiting time will be dropped from hours to minutes.

In a real world users will interact with the system by sending requests for a bus, related to a specific position. They will also be able to view the buses' location in the city, thanks to the mapping service the system will make use of.

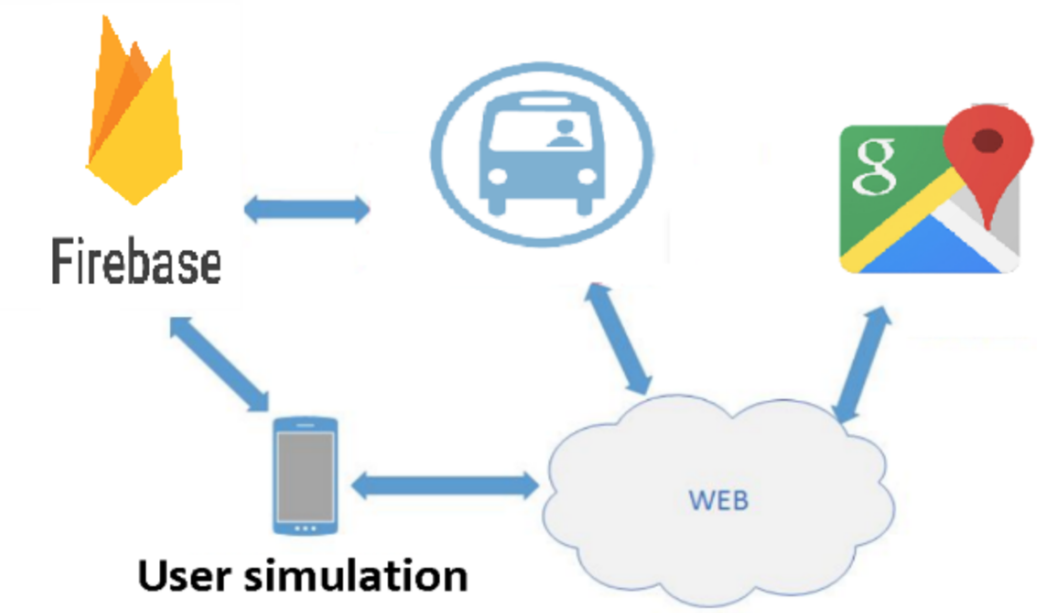
On the other hand, bus drivers will receive a notification for each user request, along with all the related information.

Fleet managers are the company's resource managers: they manage buses, drivers and routes and they have access to all the information related to the past rides.

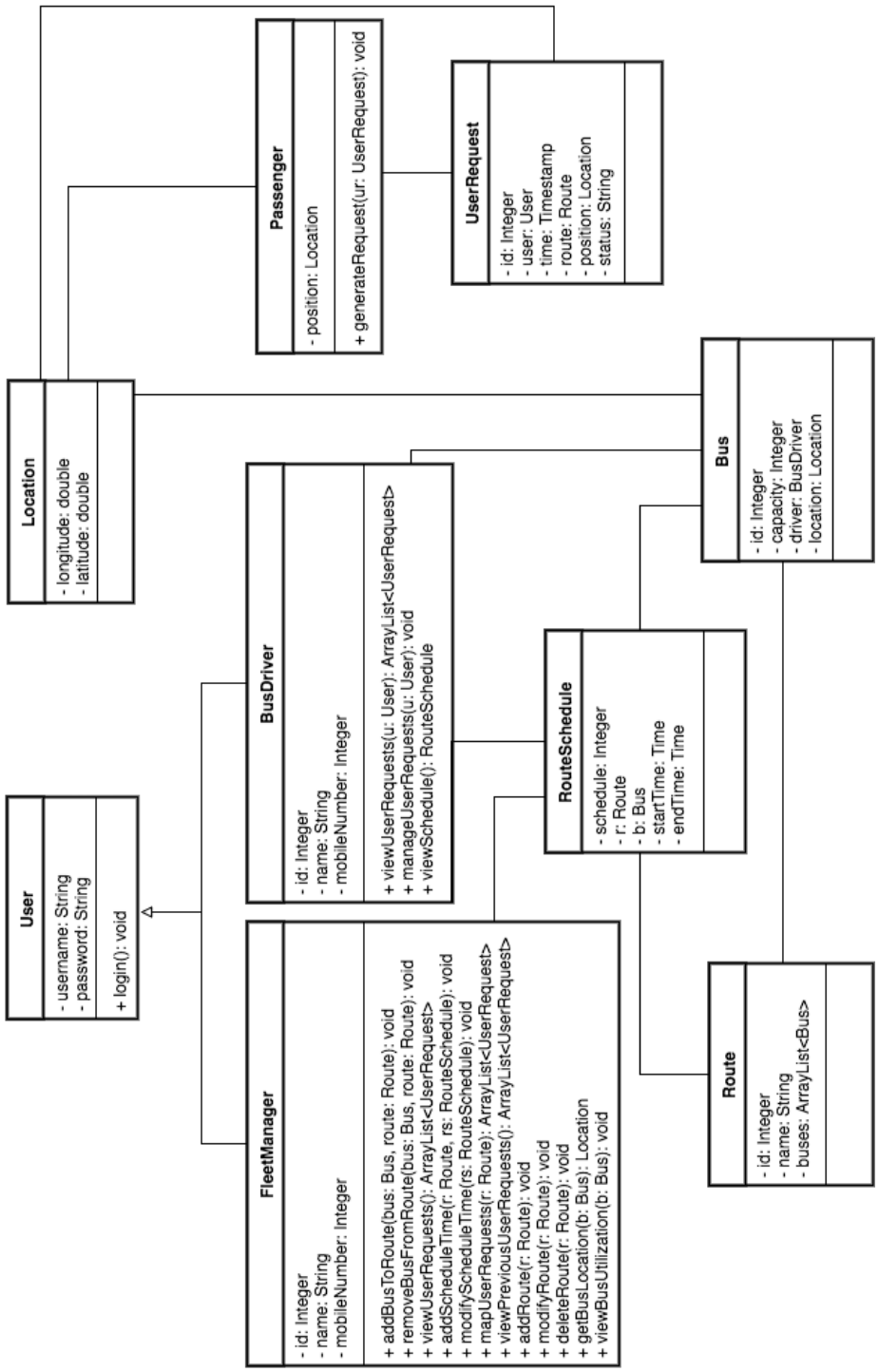
3 HIGH LEVEL SYSTEM ARCHITECTURE

The system's main components will be:

- **Firebase**, for the storage of all the information related to both the BusPlanner application and the users' requests.
- **BusPlanner application**, through which all kind of users (fleet managers, bus drivers, passengers) will interact with the system and will be able to use it over the WEB.
- **Mapping service**, to keep track of the position of both buses and users.
- **User simulation**, to simulate users' requests from a certain position.



3.1 Class diagram



4 SOFTWARE ARCHITECTURE

4.1 System architecture layers

The application consists of three layers:

- **Presentation Layer**: contains the components that implement and display the user interface and manage user interaction.
- **Business Logic Layer**: is used as an intermediary for data exchange between the presentation layer and the data access layer. Business entities, or business objects, encapsulate the business logic and data necessary to represent real world elements. The business layer's goal is to minimize the complexity by separating tasks into different areas of concern.
- **Data Access Layer**: includes methods which can interact with the database. When methods are called, it will connect to the database by using query and will return the corresponding results.

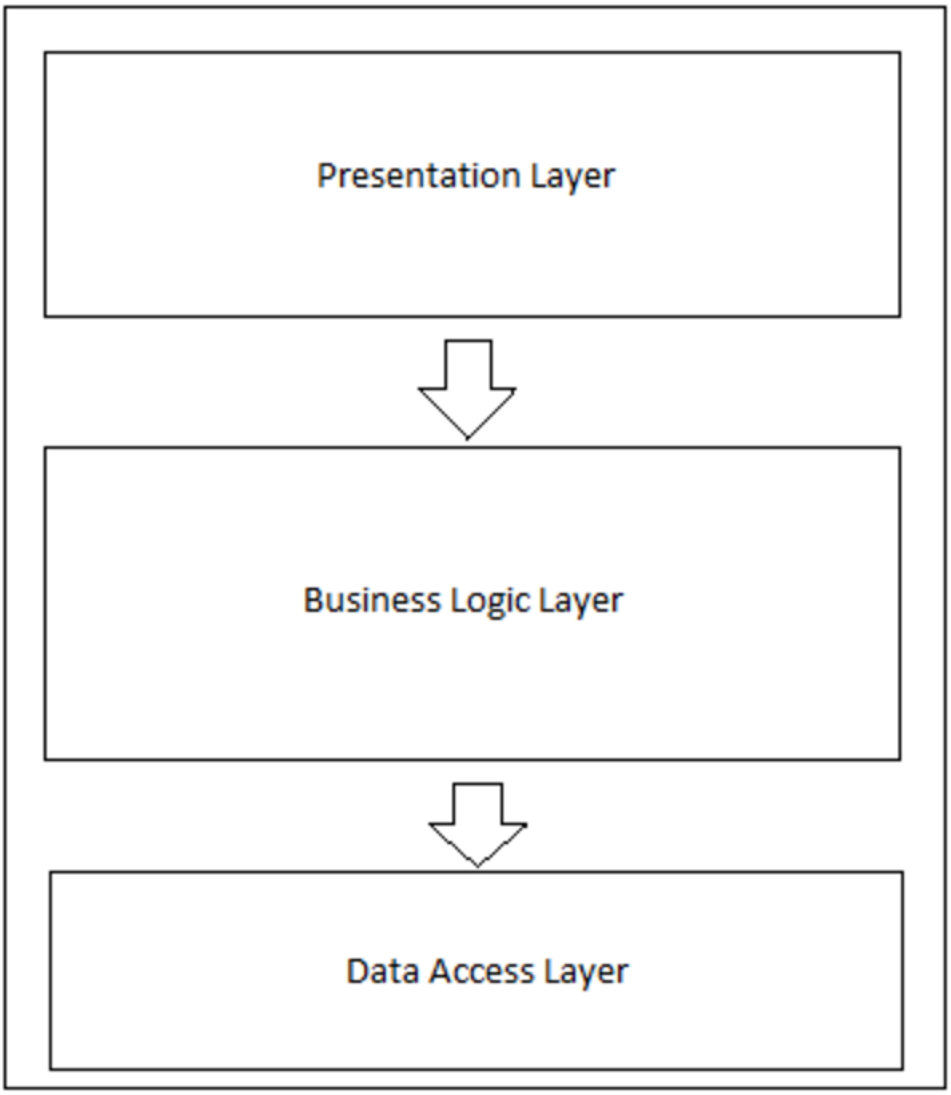


Figure 1: Three-layered architecture

With a three-layered architecture, improving the modularity of the application will be simple and this can make it easier for developers to extend features in the future. Distinguishing among the different layers allows the development team to program according to the interfaces, thus allowing an easier distribution of the work.

4.2 Technologies used

- **Back-end and database**: JavaScript and Python will be used to build the back-end of the application and the database will be based on Json files stored in Firebase.
- **Front-end**: the application will use HTML, CSS and Bootstrap for front-end.

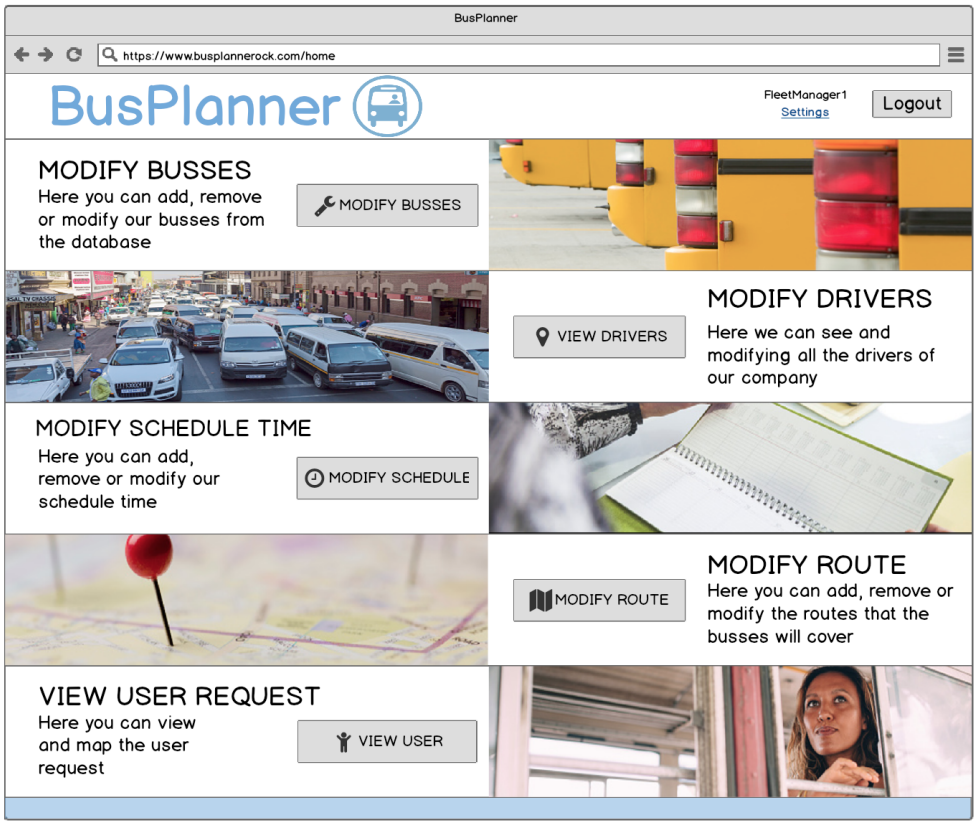
Figure 2: The technologies in the three-layered architecture of the application

5 GRAPHICAL USER INTERFACE

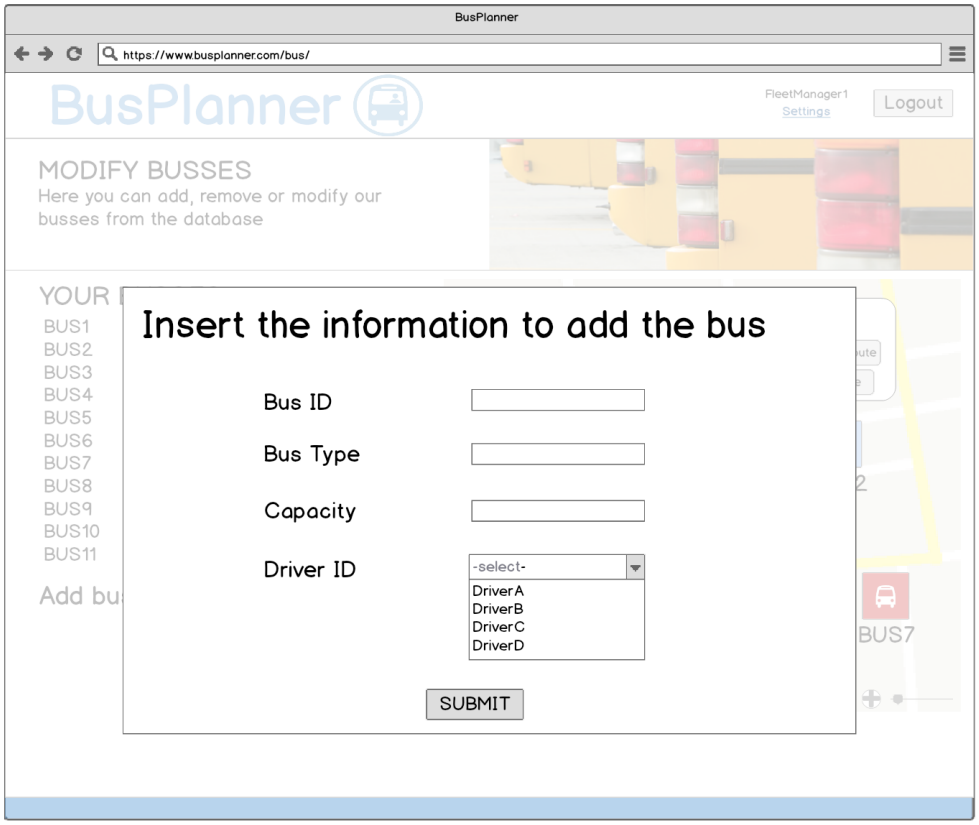
The following mockups show a brief overview of the systems interface. The main goal is to make it easy for the users to interact with the application. The applications user are fleet managers and bus drivers; these actors have two different interfaces.

5.1 Fleet manager's interface

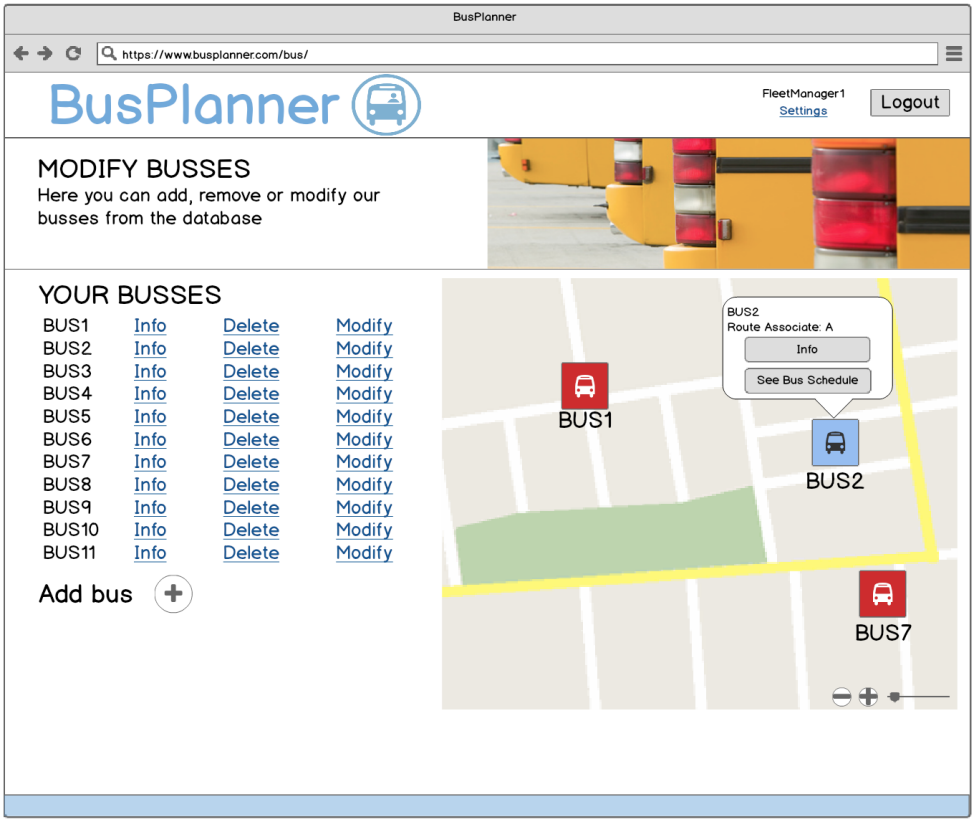
- Fleet manager's homepage:



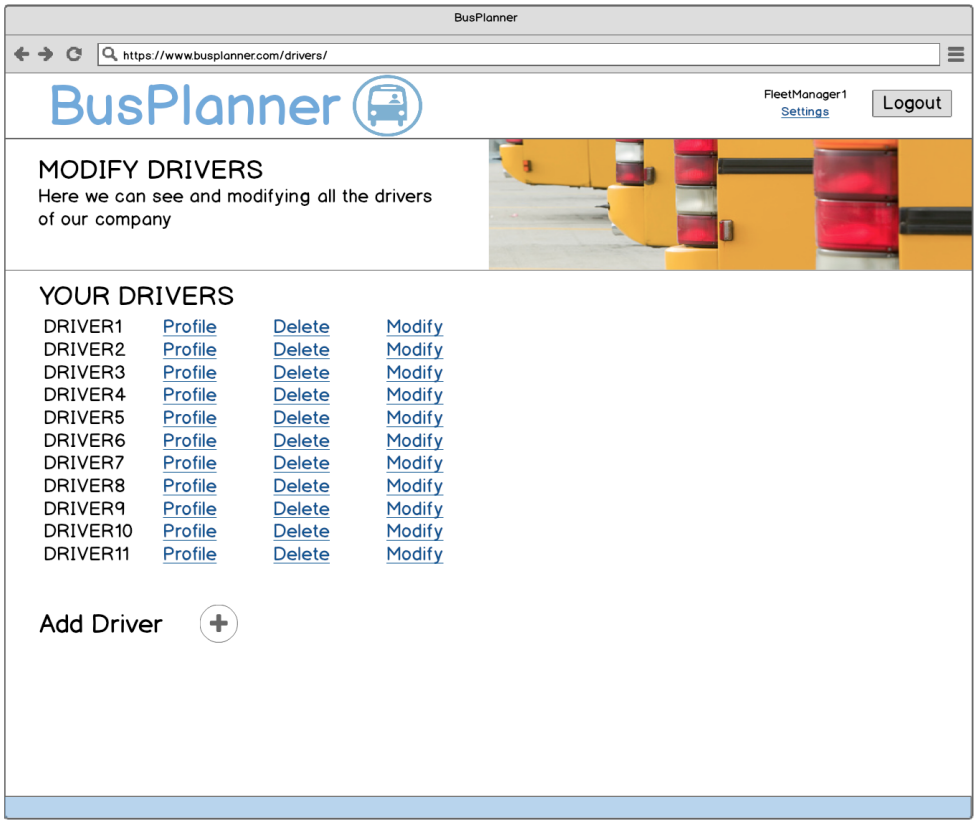
- Adding a bus:


BusPlanner
https://www.busplanner.com/bus/
BusPlanner
FleetManager1
Settings
Logout
MODIFY BUSSES
Here you can add, remove or modify our busses from the database
BUS1
BUS2
BUS3
BUS4
BUS5
BUS6
BUS7
BUS8
BUS9
BUS10
BUS11
Insert the information to add the bus
Bus ID
Bus Type
Capacity
Driver ID
-select-
DriverA
DriverB
DriverC
DriverD
SUBMIT

- Buses' list:


BusPlanner
https://www.busplanner.com/bus/
BusPlanner
FleetManager1
Settings
Logout
MODIFY BUSSES
Here you can add, remove or modify our busses from the database
YOUR BUSSES
BUS1 Info Delete Modify
BUS2 Info Delete Modify
BUS3 Info Delete Modify
BUS4 Info Delete Modify
BUS5 Info Delete Modify
BUS6 Info Delete Modify
BUS7 Info Delete Modify
BUS8 Info Delete Modify
BUS9 Info Delete Modify
BUS10 Info Delete Modify
BUS11 Info Delete Modify
Add bus
BUS2
Route Associate: A
Info
See Bus Schedule
BUS1
BUS2
BUS7

- Modify drivers:



5.2 Bus driver's interface

- Driver's login:

- Driver's homepage: