

BusPlanner	Version: 1.0
Design Description	Date: 2016-11-11

Distributed Software Development:

BusPlanner

Design Description



Revision History

Date	Version	Description	Author
2016-11-11	1.0	Initial draft	Team

Contents

1	INTRODUCTION	4
1.1	Purpose of this document	4
1.2	Document organization	4
1.3	Intended Audience	4
1.4	Scope	5
1.5	Definitions and acronyms	5
1.5.1	Definitions	5
1.5.2	Acronyms and abbreviations	5
2	BACKGROUND AND OBJECTIVES	6
2.1	Overview	6
2.2	High level description of the functionalities	6
3	HIGH LEVEL SYSTEM ARCHITECTURE	7
3.1	Class diagram	8
4	SOFTWARE ARCHITECTURE	9
4.1	System architecture layers	9
4.2	Technologies used	11
5	GRAPHICAL USER INTERFACE	12
5.1	Fleet manager's interface	12
5.2	Bus driver's interface	16
6	DATABASE MODEL	17

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	Mlardalens Hgskola, Vsters, 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 aims to reduce the number of buses per route and the time needed to do a scheduling. The users' waiting time will thus be dropped from hours to minutes.

Users will interact with the system 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.

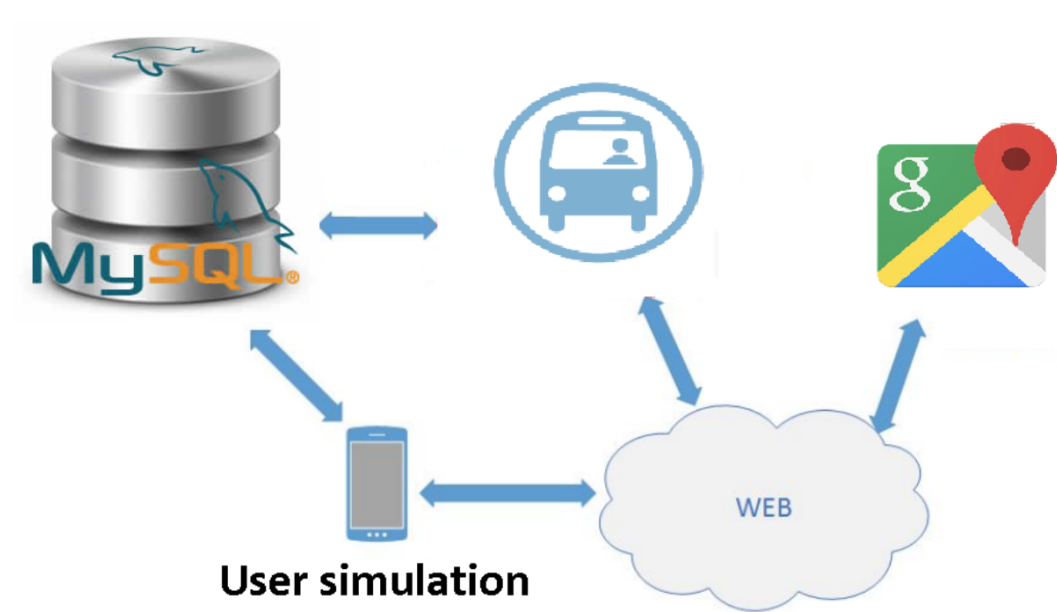
On the other hand, bus drivers will receive a notification for each user request, along with all the related information.

Fleet managers' duty will be to assign each bus to a specific route, in order to satisfy the maximum number of requests in a minimum period of time.

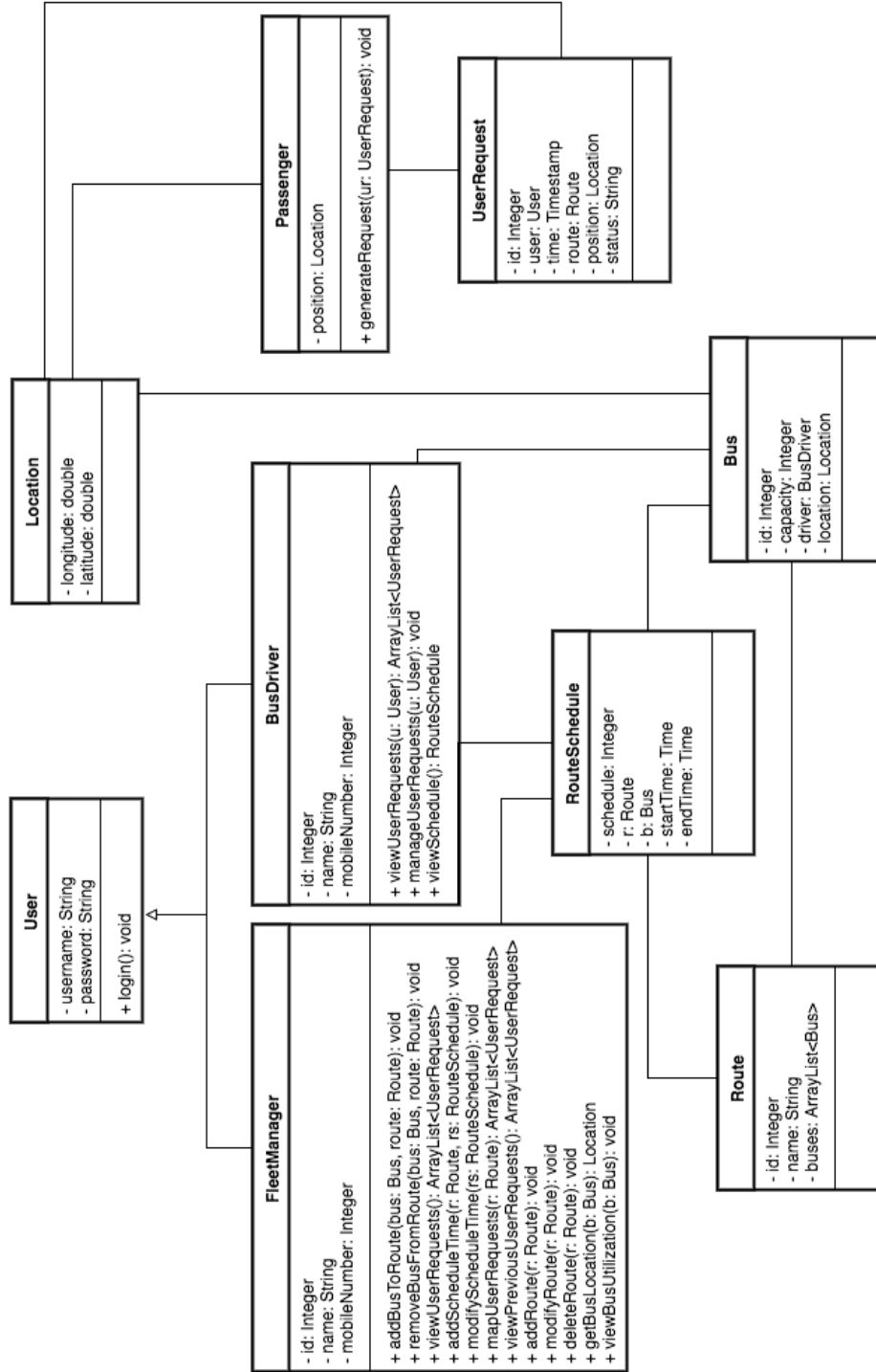
3 HIGH LEVEL SYSTEM ARCHITECTURE

The system's main components will be:

- **MySQL**, 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.

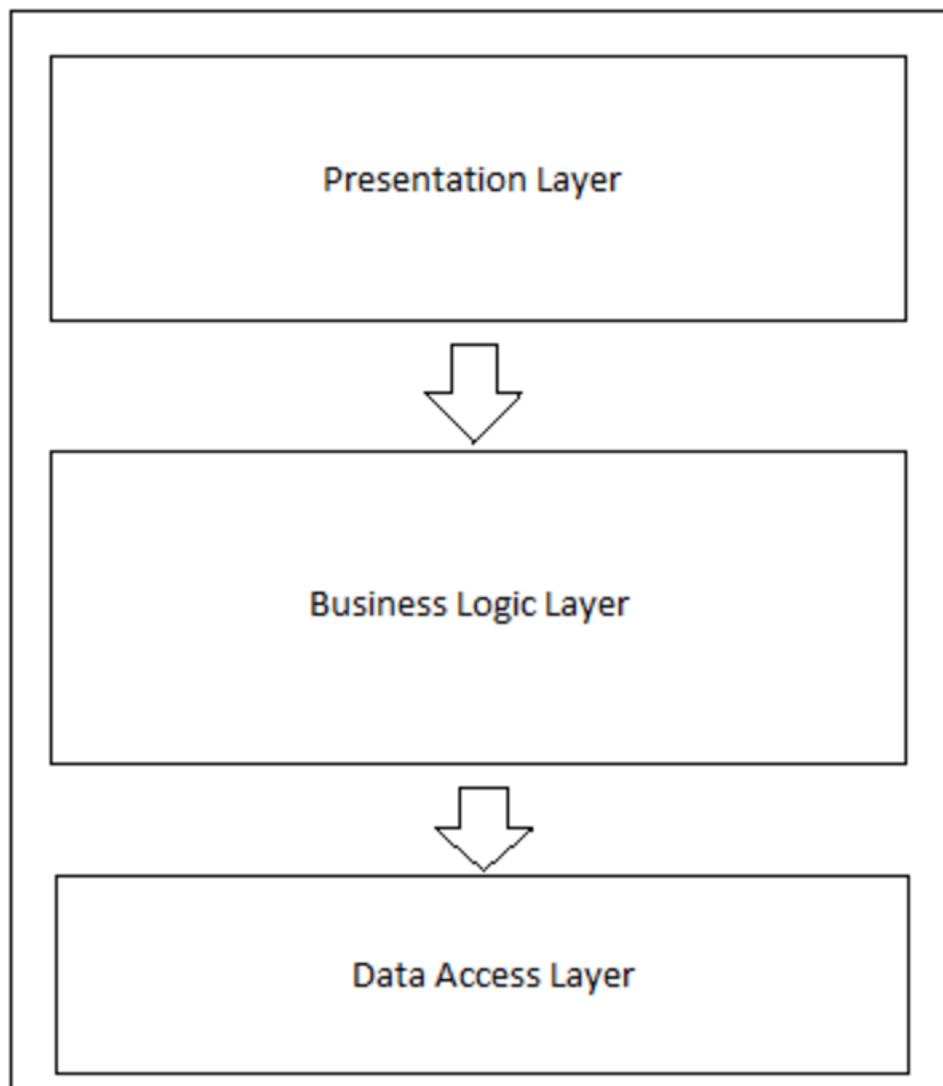


Figure 1: Three-layered architecture

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:** PHP will be used to build the back-end of the application and the database used in this application will make use of MySQL.
- **Front-end:** the application will use HTML, CSS and Bootstrap for front-end.

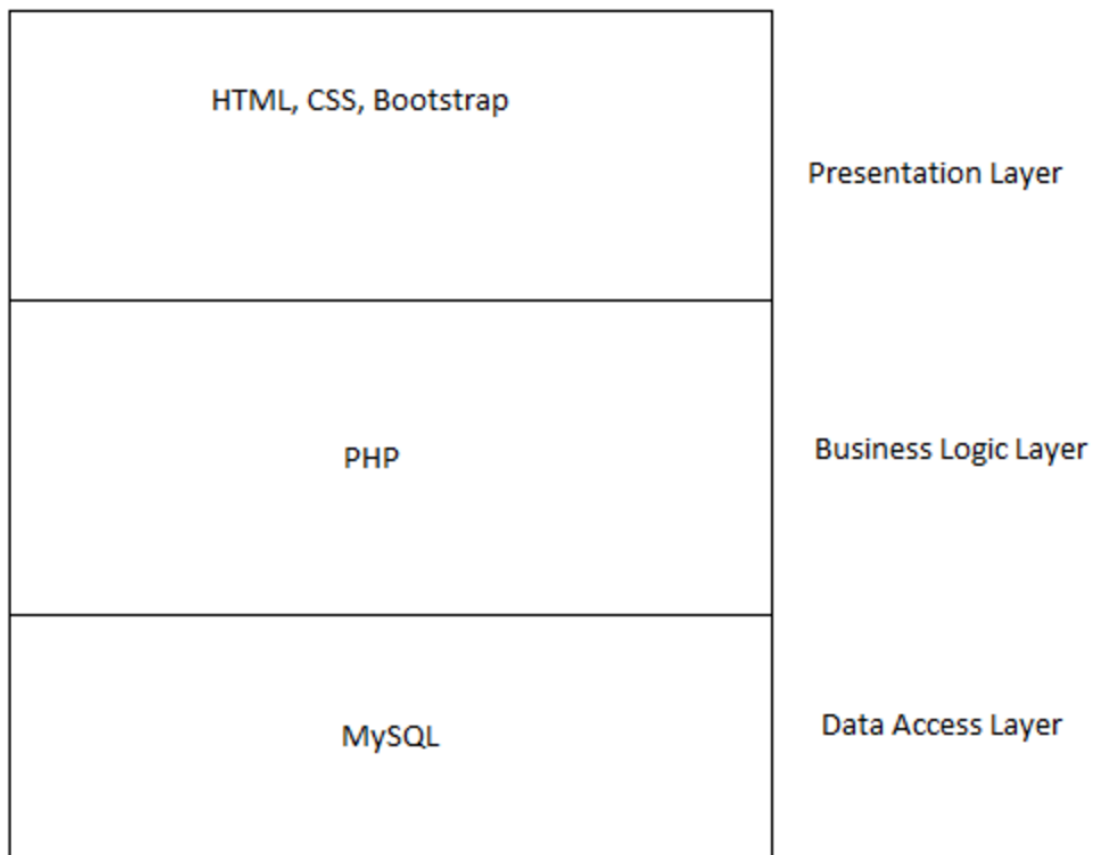


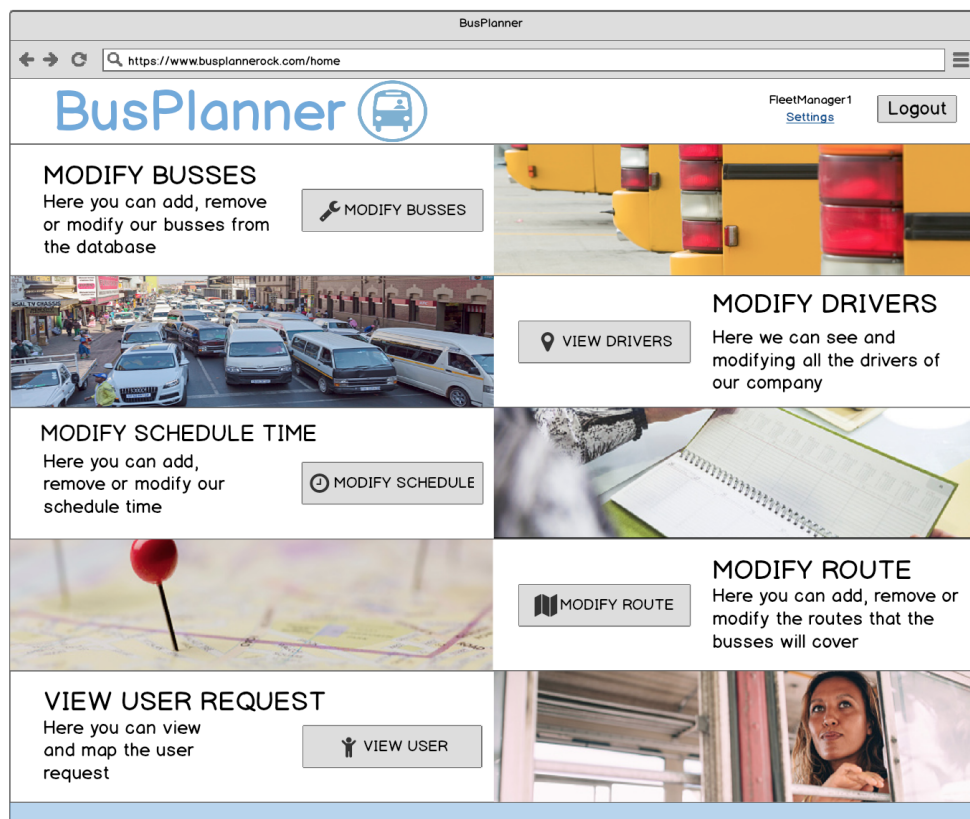
Figure 2: The technologies in the three-layered architecture of the application

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:

The screenshot shows a web browser window with the address `https://www.busplanner.com/bus/`. The page title is "BusPlanner". In the top right corner, there is a user profile "FleetManager1" with a "Settings" link and a "Logout" button. The main heading is "MODIFY BUSES" with the subtext "Here you can add, remove or modify our busses from the database". On the left, under "YOUR BUSES", there is a list of bus IDs from BUS1 to BUS11. Below this list is a link "Add bus". A modal form titled "Insert the information to add the bus" is displayed in the center. This form contains four fields: "Bus ID" (text input), "Bus Type" (text input), "Capacity" (text input), and "Driver ID" (a dropdown menu with options "-select-", "DriverA", "DriverB", "DriverC", and "DriverD"). A "SUBMIT" button is located at the bottom of the modal. In the background, a map is visible with a red bus icon labeled "BUS7".

BusPlanner

FleetManager1
Settings Logout

MODIFY BUSES
Here you can add, remove or modify our busses from the database

YOUR BUSES

- BUS1
- BUS2
- BUS3
- BUS4
- BUS5
- BUS6
- BUS7
- BUS8
- BUS9
- BUS10
- BUS11

Add bus

Insert the information to add the bus

Bus ID

Bus Type

Capacity

Driver ID

- DriverA
- DriverB
- DriverC
- DriverD

SUBMIT

BUS7

- Buses' list:

BusPlanner

FleetManager1
Settings

Logout

MODIFY BUSESSES

Here you can add, remove or modify our busses from the database

YOUR BUSESSES

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

- Modify drivers:

BusPlanner

← → ↻

https://www.busplanner.com/drivers/

≡

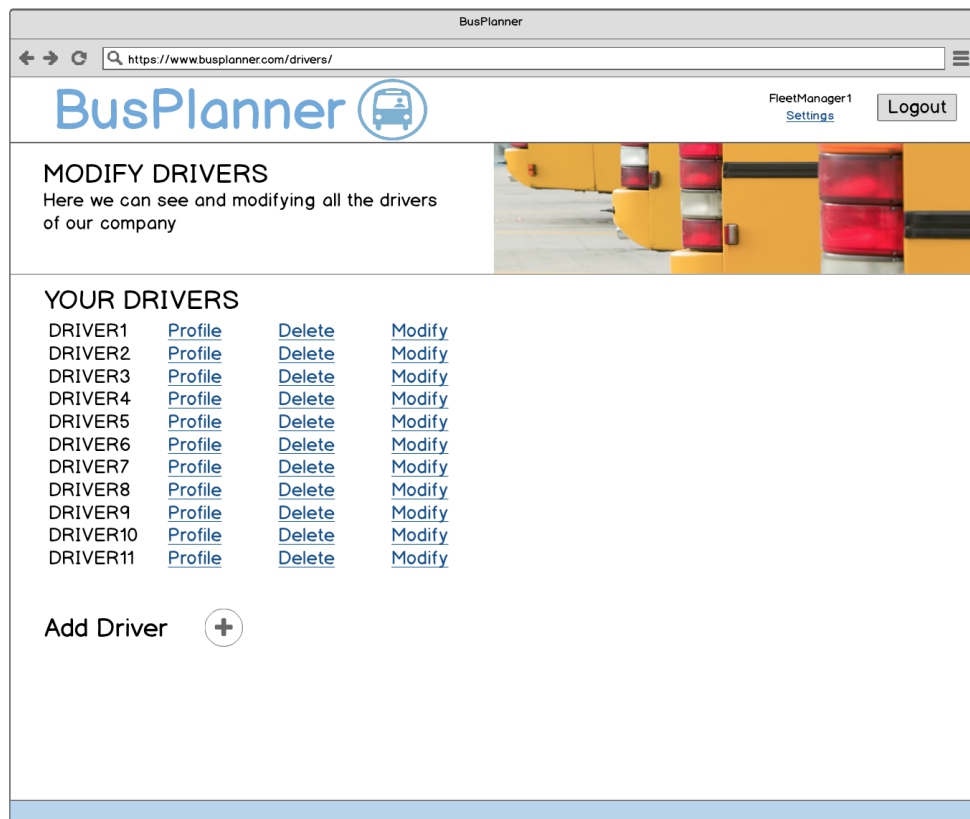
BusPlanner

FleetManager1
Settings

Logout

MODIFY DRIVERS

Here we can see and modifying all the drivers of our company



YOUR DRIVERS

DRIVER1	Profile	Delete	Modify
DRIVER2	Profile	Delete	Modify
DRIVER3	Profile	Delete	Modify
DRIVER4	Profile	Delete	Modify
DRIVER5	Profile	Delete	Modify
DRIVER6	Profile	Delete	Modify
DRIVER7	Profile	Delete	Modify
DRIVER8	Profile	Delete	Modify
DRIVER9	Profile	Delete	Modify
DRIVER10	Profile	Delete	Modify
DRIVER11	Profile	Delete	Modify

Add Driver

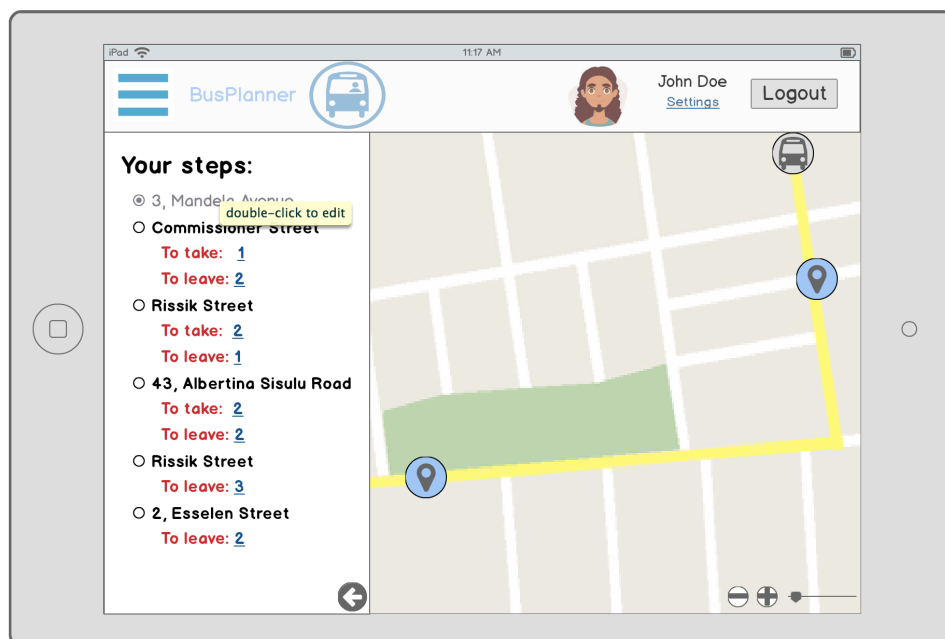
+

5.2 Bus driver's interface

- Driver's login:



- Driver's homepage:



6 DATABASE MODEL

Route

Route_id	integer	PK
Route_name	varchar	

Login

Login_id	varchar	PK
User_name	varchar	
Password	varchar	
User_type_id	integer	

UserType

User_type_id	integer	PK
User_type	varchar	
Login_id	integer	FK

UserRequest

User_id	integer	PK
Timestamp	timestamp	
Route_id	integer	FK
Latitude	float	
Longitude	float	
Status	varchar	
Starting_schedule	integer	

Bus

Bus_id	integer	PK
Bus_type	varchar	
Driver_id	integer	FK
Bus_capacity	integer	
Longitude	float	
Latitude	float	

Driver

Driver_id	integer	PK
Driver_name	varchar	
Mobile_number	varchar	

FleetManager

Manager_id	integer	PK
Manager_name	varchar	
Mobile_number	varchar	

UserSchedule

User_schedule_id	integer	PK
User_id	integer	FK
Driver_id	integer	FK
Bus_id	integer	FK
Route_id	integer	FK
Status	varchar	

RouteSchedule

Route_schedule_id	integer	PK
Start_time	timestamp	
Finish_time	timestamp	
Bus_id	integer	FK
Route_id	integer	FK
Start_point	varchar	
End_point	varchar	
Timestamp_departure	timestamp	
Timestamp_arrival	timestamp	
Number_onboarding	integer	
Number_debording	integer	
Number_current	integer	