Design Document: bike4share

Maffioli Sara, Papale Lorenzo,
Stucchi Lorenzo & Vaghi Federica

June 9, 2019

Contents

1 Introduction

The aim of the project is to develop a web application called “bike4share” for a bike sharing service in order to guarantee to the user a web service to search for the availability of bikes and free stalls around his position or the desired location in the city.
The web application must ensure the technician to have a private section showing analytic graphs and statistics on the bike flow in real time. The main used technologies are shown in Table 1

PostgresSql	Used in order to to manage the databases
Python	Perform the outputs, build the map and manage the map’s background
Flask	Set for the visualisation of the statistics

Table 1: Technologies overview

1.1 Purpose

This document is a generic Technical Design Document for “ bike4share”. The purpose of this document is to outline the technical design of the web application system and provide an overview for the used databases and software.
It is mainly intended to assist the customer that has invested in sustainable mobility (Municipality of Milan) and the technicians who have to monitor the system in order to work on data and extract the needed information.

1.2 Scope

The final output of the “ bike4share” project will be a client–server computer program which the client, including the user interface and client-side logic, runs in a web browser. The system will allow the user to interact with a web page in which different functions are offered to different type of users (Table 2). The Basic Functions allow the not registered users to perform some elementary activities. The Premium Functions are in addiction of the Basic Function and they are accessible for the users after the registration, once the log-in phase is successfully completed, the registered user can access to different functionalities that can help the usability of the app. A specific section of the application is devoted to the technician’s use. The technicians are requested to complete the registration (with a specific secret code given by the master of the service) and log-in to have access to their specific section. Here it’s possible, in addition to all the other functionalities, to perform some analysis and graphical statistics with specific tools.

BASIC FUNCTION	• Showing the position of all the bike stalls • ID Number of bikes stalls per each selected station • Number of the stalls for each selected station • Manually typing of the address or select the position on the map
PREMIUM FUNCTION	• all the previous basic function are included • Display the position of the user on the map • Help the users to visualise the nearest station with respect to their position • Showing the list of all the available stations nearby • Manually typing of the address or select the position on the map
TECHNICIAN AREA	• All the previous premium function are included • Plotting of the stations median bikes availability per day of the week • Plotting of the stations median bikes availability per day of the month • Plotting of the percentage of bikes availability per day • Plotting of the percentage of bikes availability per month • Bike availability trend per weekend • Real-time data previous week • Real-time data previous month • Move over the map

Table 2: "bike4share" functionalities overview

The main advantages of "bike4share" is the possibility of planning the usage of a bicycle for the daily mobility in a sustainable way, in order to avoid delay, stress and unexpected events while there are important delivery or scheduled appointment. The main goal is to provide to the customers, users and technician the best services as possible in terms of system performance and functionalities in order to help them with their daily objectives. Thanks to the premium functionalities it's possible to avoid different organisational problems. Even if the users don't allow to access to the position because of privacy reasons it's however possible to use the "bike4share" web application thanks to the manual typing of the addresses. The log-in, e-mail address and password request are able to reduce the security risks for the users and the main risk related to the usage of this application in case of a security hack will be related to the acquisition of data regarding the positions of the users.

1.3 Definitions, acronyms and abbreviations

DB:database
DBMS: database management system
b4s: bike4share
HW: hardware
SW: software
DD: Design Document

1.4 Document Organisation

This document is organised in different sections like described in Table 3.

Introduction	Provides information related to this document
Design Overview	Describes the approach, architectural goals and constraints
System Architecture	Describes s the various system components and their integration
Data Design	Outlines the design of the DBMS and non-DBMS files
Detailed Design	Describes the proposed design in detail
Human-Machine Interface	Describes the proposed interface

Table 3: Document organisation

2 Design Overview

This section briefly introduce the system context and design, and discuss the background to the project

2.1 Approach

The "bike4share" web application is created and extended in multiple phases over the course of the project (Table 4).

Requirements Phase	Describing the candidate architecture to be validated
System Design Phase	Describes the approach, architectural goals and constraints
Prototype Phase	Evolutionary Prototype and the architectural foundation is created
Construction Phase	Technical procedure for the tasks is defined
Training Phase	Test of the final product

Table 4: Phases of the project's development

2.2 Background Information

The customer has invested in sustainable mobility by installing a number of bike stalling stations around the City. The implemented bike sharing system must be monitored with the "bike4share" web application for assessing the bikes flow both in space and time and, eventually, for designing future system improvement actions. Also, the customer has the need to inform in real time bike sharing users about the status of each stations.

2.3 Constraints

TECHNICAL CONSTRAINTS	• PostgreSQL • Python-Flask • Python-Bokeh • Python data analysis and plotting • A vector map of the stalling stations (shapefile)

DELIVERY CONSTRAINTS	• First version of requirement analysis document on the 29th of April in 2019. • First version of the Design Document and Test Plan are scheduled on the 26th of May in 2019. • Implementation, test report and updated document scheduled on the 09th of June in 2019.

Table 5: Constraints overview

2.4 Design Trade-offs

The trade-off method used in the development of the "bike4share" application is based on the functional and quality requirements of the predicted system, a number of scenarios was created in order to represent both the day-to-day use, and the intended use of the Premium Functionality introduced (ref cap 2.8.1). Using these scenarios, different functional parts are then extracted from architecture descriptions prepared for each of the use-case scenarios. Since the evaluation is conducted at the architecture level, the different parts can cover measures such as the number of active data repositories, passive data repositories, persistent and non-persistent components, data links, control links, logical groupings, styles and patterns and violations of the intended architecture. These metrics are collected by the developer of the whole system that estimate complexity, impact and effort for each of the scenarios for each of the architectures using an existing architecture as a point of reference (for example the BikeMi services). Based on the collected metrics it is then possible to compare the architecture alternatives and based on that select the most appropriate architecture for the improvement of the system. The alternative architectures are assessed with mainly respect to robustness but they are also compared for reliability, maintainability, interoperability, portability, scalability and performance.

2.5 User Characteristics

There are different kind of users. Every user can interact with different information about the bike stalls placed in Milan (Table 6).

REGISTERED USER	A user that is registered yet and that wants to access the "b4s" web application services in order to use the premium functions available on the application. It could be a person that uses this service frequently.
NOT REGISTERED USER	A user that isn't registered yet and that wants to access the "b4s" web application services in order to use the basic functions available on the application, it could be a person that doesn't use this service frequently or a new user that wants to start an approach to this service and try it for the first time
TECHNICIAN	A person who is authenticated thanks to specific credential given by the master of the service and who has been granted supervising permissions. He/she is allowed to view, analyse, access statistics and aggregated information about the "b4s" web application services.

Table 6: User-type description

2.5.1 User Problem Statement

In this section the main problems that a user could have by experiencing the "b4s" web application are described. It is possible to identify the following problems with respect to the type of event that is occurring .

USER'S PROBLEM	TYPE OF EVENT
REGISTRATION PHASE	• User name already existed: the username already exists and it is proposed to change the chosen username • E-mail already existed: the e-mail already exists, so an error is shown and it is proposed to change the chosen e-mail • Incorrect e-mail address: the format of the mail is wrong so the system requires to correct it and to insert a valid one • Invalid secret-key(only technician's problem):it can be solved by contacting the administrator

LOG-IN PHASE	• Forgot password:the user can follow the recovery password procedure • Localisation doesn't work: so the system provides to the logged user the possibility of manually inserting the needed address and find the position or adding a point in his position • Maps providers crash • Server doesn't work

Table 7: User's problem overview

3 System Architecture

Our web application "bike4share" will work thanks to several components supporting it. We can recognise three levels of the system: User, Software and Data, as is showed in figure 1.

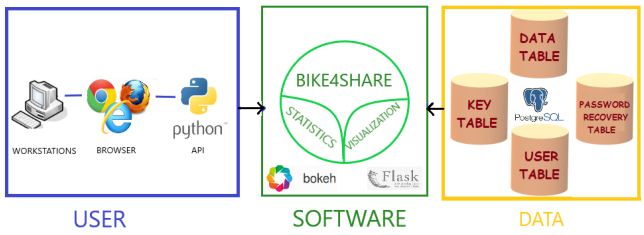

Figure 1: Overall view of "bike4share" system architecture

Starting from the right and considering the inputs, Data are retrieved from four Databases, one for the registered users and technicians, one for the information regarding the data of the bike stations, one for the password recovery procedure and one for the secret key created for the first log-in of the technician. The outputs of "b4s", instead, are the visualisation of a map showing the bike stations with the relative information (bikes and free stalls) and the visualisation of statistics (only for technicians)and they are showed in the middle of the Figure 1. On the left there is the section regarding to the user, capable of viewing the map to obtain the information, and the technician, who can analyse data and statistics. Both the actors can perform their purposes accessing the page through the browser.

3.1 Hardware Architecture

As seen in the previous paragraph, the Bike4share is designed as a distributed system: a three-layers architecture. A graphical description is shown in the figure 2.

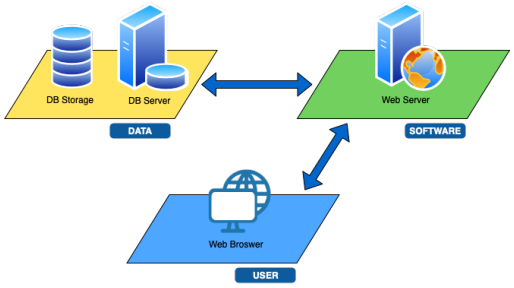

Figure 2: Overall view of "bike4share" hardware architecture

- Database Server: This component, which includes also the DB storage, provides all the data when requested by the application (with SQL queries), allowing it to accomplish its functionalities.
- Web Server: It is focused on the workflow and it is responsible for the communication with both the Database Server and the User.
- Web Browser: The role of this component is crucial for what concerns the interface, the user information retrieval (for registration and login) and the presentation of the information needed by the User and by the Technician.

It is important to consider the fact that in the programming phase the architecture of the system is centralised. Since the system involves the use of a single machine, the role of all the components is played by the machine itself as shown in the figure 3.

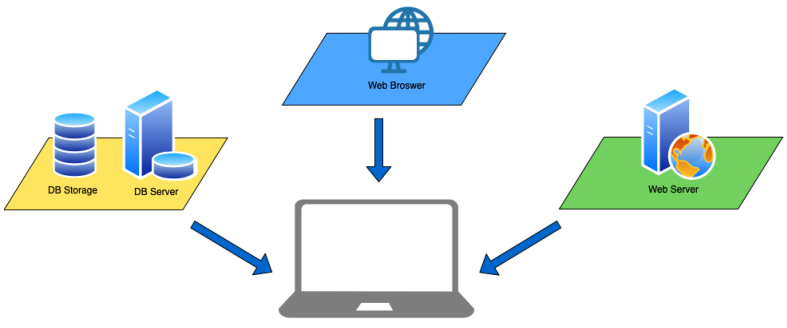

Figure 3: "bike4share" hardware architecture during system creation phase

3.2 Software Architecture

The software architecture is mainly based on the implementation of the statistics thanks to Bokeh and the visualisation of the obtained results through flask. The programming language is Python and in particular the libraries and the tools that are used are:

Packages Name	Version	Description
Pandas	0.24.2	It provides high-performance, easy-to-use data structures and data analysis tools for the Python programming language
Geopandas	0.4.1	It combines the capabilities of pandas and shapely, providing geospatial operations in pandas and a high-level interface to multiple geometries to shapely random function
Psycopg2	2.8.1	The most popular PostgreSQL DB adapter for the Python programming language. It was designed for heavily multi-threaded applications. It features client-side and server-side cursors, asynchronous communication and notifications, many Python types are supported out-of-the-box and adapted to matching PostgreSQL data types; adaptation can be extended and customized thanks to a flexible objects adaptation system
Sqlalchemy	1.3.2	It is the Python SQL toolkit and Object Relational Mapper that gives application developers the full power and flexibility of SQL. It provides a full suite of well known enterprise-level persistence patterns, designed for efficient and high-performing database access, adapted into a simple and Pythonic domain language
Geoalchemy2	0.6.1	It provides extensions to SQLAlchemy for working with spatial DBs and it is focused on PostGIS
GeoJSON	2.4.1	Is a format for encoding a variety of geographic data structures that supports the following geometry types: Point, LineString, Polygon, MultiPoint, MultiLineString, and MultiPolygon. Geometric objects with additional properties are Feature objects. Sets of features are contained by FeatureCollection objects

Bokeh	1.1.0	It is an interactive visualisation library its goal is to provide elegant, concise construction of versatile graphics, and to extend this capability with high-performance interactivity over very large or streaming datasets
Flask	1.0.2	It is a micro web framework written in Python that supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools
Werkzeug	0.14.1	Werkzeug is a comprehensive WSGI web application library
psycopg2	2.8.1	psycopg is the most popular PostgreSQL adapter for the Python programming language. At its core it fully implements the Python DB API 2.0 specifications
PgAdmin	4.5	It is a management tool for PostgreSQL and derivative relational databases such as EnterpriseDB's EDB Advanced Server
Leaflet	1.4.0	It is the leading open-source JavaScript library for mobile-friendly interactive maps, it has all the mapping features most developers need.It works efficiently across all major desktop and mobile platforms, can be extended with lots of plugins and it is an easy to use API
Leaflet-knn	0.1.0	In order to compute the distance between the points and returning the nearest one it is used the external functions called leaflet-knn with the relative license (https://github.com/mapbox/leaflet-knn)
Leaflet Control Geocoder	1.8.2	This package allows to insert manually an address returning the position on the map

Table 8: Software architecture packages and tools

The structure of the software is based on the following python script and reusable functions as it is shown in the table 9.

NAME	DESCRIPTION AND FUNCTIONS
createSchema.py	It allows the creation of the different table used by the "bike4share" web application. Thanks to the SQL query and the pgAdmin DBs connection it is possible to access, create, updated, extract data and fill the different DBs which interact with the user side in order to guarantee the consistency with the latest version of the data
realtime_data.py	It allows the data analysis in real-time thanks to the connection with the DBs
func.py	It is a function that allow to send an email to the user in order to send back an e-mail containing the recovery code for the forgot password procedure. it'll take the e-mail address and the random code of the users automatically from the DBs. for the generation of the code the following function is used: • Key_generator: it is a function defined in order to automatically generate some random password that will be given to the technician directly from their chief or by our customer (the buyer of the system, the Municipality of Milan)
Statistics.py	It is the section of "bike4share"software development that allows to creation of plots, representing the different evaluated statistics. The principal functions are: • pd.to_datetime: Pandas function to convert string date time into Python date time object • df_bike.groupby: it allows to split the data into groups based on some criteria • ColumnDataSource: it allows to store the data to use in the bokeh graph • figure: it allows to create a new figure in Bokeh • callback: it allows to upload the graph • getPointCoords: it allows to retrieve coordinates from the geodataframe

main.py	It is the main part of "bike4share"software development, it allows to log-in, access,register, make some statistics and visualise the map in which the bike and stalls information thanks to the direct connection to the DBs with the following main function: • get_dbConn: it allows the connection with the DBs • close_dbConn: it allows the disconnection from the DBs • load_logged_in_user: it allows to check if the user is inserted yet in the user's DB and make the consequent operations • index: it allows the map visualisation • register: it allows the registration of the user • log_in: it allows the log-in of the pre-registered users • forgotpassword: it allows to recovery the forgotten password both for the users than the technician • set_new_password: it allows both the user and the technician to set a new password and store it in the DB instead of the old forgotten one • log_out: it allows the log-out of the users • tech_reg: it allows the registration of the technician • bash_command: it allows the bokeh application running on port 5006 to be accessed at port 5000 by Flask • statistics: it allows Flash to read what is the bokeh server and to have access to the statistics.
downloadStation.py	It allows to download the data about the station from the DB and to save it to a GeoJSON file and transform it into a javascript variable.

Table 9: Software structure overview

3.3 Communications Architecture

The Communication diagram, fig 4, is a diagram that shows the interactions between elements at run-time. They are used to visualise inter-object relationships. In the schema in fig. 4 is possible to notice how the createSchema python code is linked with the DB and how it can interact in the creation, modification and update of the dataframe.

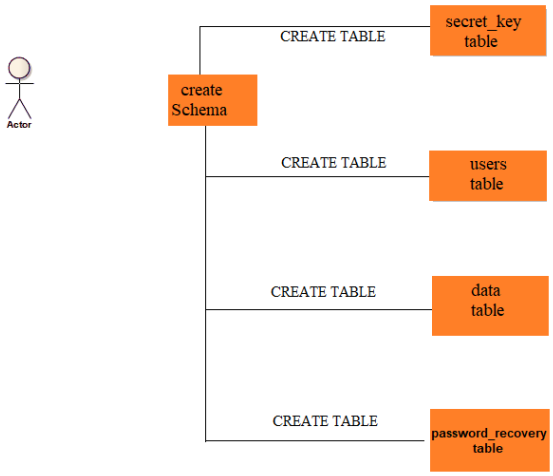

Figure 4: Communication diagram related to the createSchema module

In the figure 5 is possible to see how the different modules of the "b4s" application are linked, what kind of function are used and how they interact between themselves.

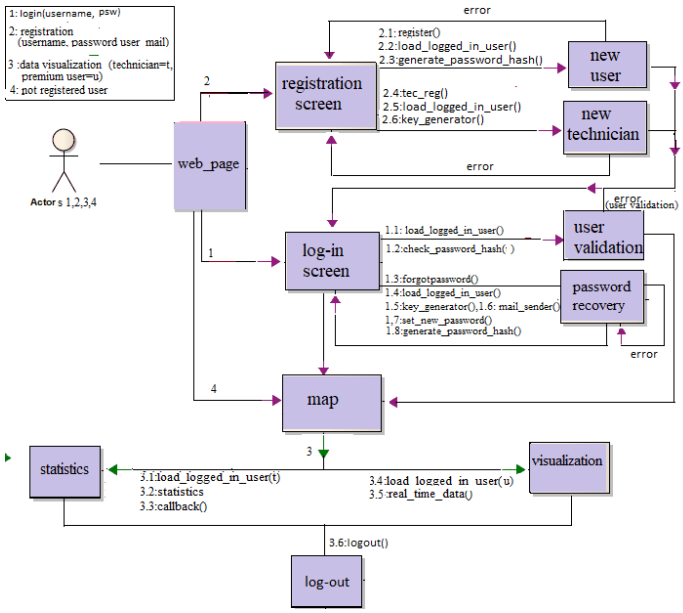

Figure 5: Communication diagram related to the webpage module

4 Data Design

The DBMS chosen for this purpose is PgAdmin4 and the DB is mainly composed by different tables:

Key_list	This table contains the random generated passwords for the first registration of the technician,once that one of those password is been used it'll be deleted.
User_bike	This table contains the credentials of all the registered users in order to allow them to perform the log-in procedure. It's checked, filled and updated every time that a new user try to make the registration
Stations	This table contains the all the data related to the stations information, e.g. the address, the stalls...
password_recovery	This table contains all the needed information in order to perform the forgot password procedure

Table 10: DBMS tables components

4.1 Database Management System Files

In this section will be described how the DB will be designed. In the fig 6 is shown the suggested mental approach to the "b4s" application and the relationship with the corresponding DB tables. The users can navigate starting from the web application and have to think about the different choices to do in relationship with their purposes.

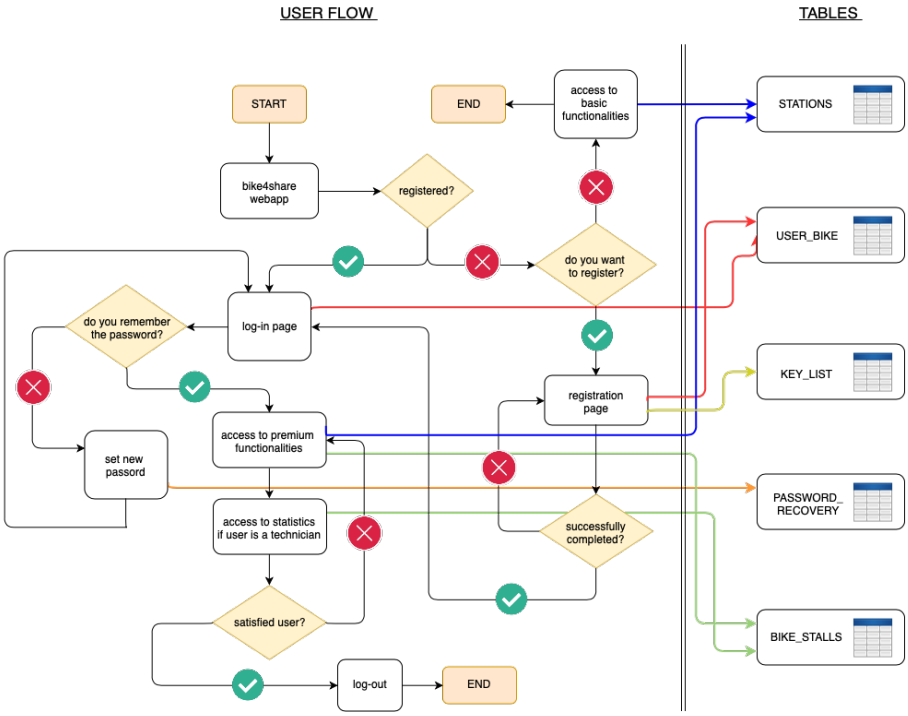

Figure 6: Representation of the flow of the events related to the usage of the b4s

The table 11 will show the main attributes that describe the DB tables.

TABLE NAME	TABLE COMMENTS	COLUMN NAME	DATA TYPE	DATA LENGTH
KEY_LIST	Random technician passwords	Id_key secret_key	int(PK) varchar	- 35
STATIONS	Information regarding stations	ID BIKE_SH INDIRIZZO STALLI	int text text int	- - - -

		LOCALIZ LATITUDINE LONGITUDINE	text double double	- - -
USER_BIKE	Information regarding users	user_id user_name user_password user_type	int(PK) varchar varchar varchar	- 255 255 255
PASSWORD_ RECOVERY	Information regarding users who have forgotten their password	id_psw psw_recovery user_name	int(PK) varchar varchar	- 11 255

Table 11: Database structure

5 Detailed Design

5.1 Software Detailed Design

In this section the different module composing the "bike4share" web application will be described and commented.

5.1.1 Module 1: CreateSchema

In this module it is possible to create the different table stored in the DB.

CreateSchema: Processing The first operation that is made in this module is the dropping of the existing table in order to avoid inconsistency of data. After that it's possible to create the tables (key_list, user_bike and password_recovery) with all the needed parameters. The key_list table is filled thanks to the function key_generator() that is able to generate different random password composed by chars, numbers and special characters value for a defined number of time; in this way is possible to check the quantity of allowed technician and to check if their password are used yet. Also the table stations, with the information about the position of the stalls is created into the database. After the availability for every station is added in to table bike_stalls.

Local data structures The main structures used to store data are data frames.

5.1.2 Module 2: downloadStation

This module is used to download the data of the stations directly from the server, it is automated and it's imported by the main module.

downloadStation: Processing It implies the creation of a geojson file with the information about the position and the number of stalls for every station.

5.1.3 Module 3: realtime_data

This module is used to download the data of the availability of bikes for each station directly from the server, it is automated and it's imported by the main module.

realtime_data: Processing It implies the creation of a geojson file with the information about the number of free stalls and bike available for every station.

5.1.4 Module 4: Main

This module is the main module of the "b4s" system used to create the general schema of the website in order to connect the different sections: map, login, logout, registration, forgot password, set new password and statistics.

main: Processing The first point consists in the creation of the application instance. Then different functions are implemented to establish the connection with the DB such as get_dbConn() and close_dbConn(). The different sections are connected through the functions shown in Table 9.All the URLs are defined as routes in the Flask application through the app.route decorator.

main: Local data structures The main structures used to store data are data frames and lists.

5.1.5 Module 5: statistics

The aim of this module is to create different statistics on the base of the data retrieved by the DB, from tables 'bike_stalls' and 'stations'.

statistics: Processing The first operation that is made in this module is the connection to the DB in order to store the tables, above mentioned, of the data respectively in two different data frames. In this way is possible to extract the information needed to compute statistics, e.g. day and month. Functionalities to retrieve real time data have been implemented too. The data can be stored in the data frame and group by on the base of the different criteria as the median and the total availability of bikes. Using Bokeh functions it is possible to plot these statistics and to design the relative widgets.

statistics: Local data structures The main structures used to store data are data frames and lists.

6 Human-Machine Interface

The user interface is easy and intuitive, fig 7. When the webpage is opened, a map appears on the screen. The map shows the bike stations (as markers) in terms of position in space, with the possibility to interact with it; clicking on them, the user can immediately know what is the address and the capacity (total number of stalls)

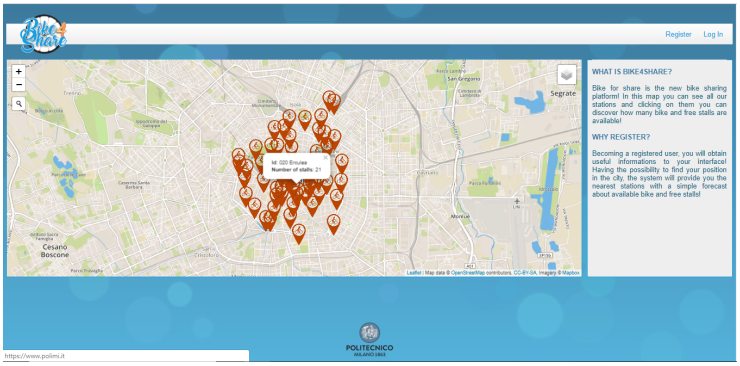

Figure 7: homepage overview

6.1 Interface Design Rules

As design rule, the webpage is provided with the constant presence of the navigation bar on the top of the page. the webpage is distinguished between the user-side and the technician-side, in the specific it contains the following buttons for the users:

- Bike4share (returning the homepage)
- Register
- Log In

This design rule can help the user in navigating on the webpage with the possibility to return to the home page if something goes wrong.

Figure 8: Detail representing the navigation bar

For the technician there is also a button called "Technical Area" that allows them to access at their private session and to have access to the statistics.

Figure 9: technical area overview

6.2 Inputs

Because of the diversity of the users there will be different inputs requests. These inputs will be used in order to provide to the user the "b4s" services.

NOT REGISTERED USER	inputs are not required; the user accesses the "b4s" web application in order to visualise the map and the relative stalls position or the total number of bikes
REGISTERED USER	inputs are required for the different procedures: • registration: username, password, e-mail, authorisation to GPS access • log-in: username, password • forgot password: username, e-mail • set new password password: username, received code, new password
TECHNICIAN	inputs are required for the different procedures: • registration: username, password, e-mail, secret key, authorisation to GPS access • log-in: username, password • forgot password: username, e-mail • set new password password: username, received code, new password

Table 12: Input requirements

All these input data will be stored in the DB (the password will be hashed for security purpose) and will be checked in the DB.

6.3 Outputs

The "b4s" web application provides several outputs, based on the type of user who is logged in.

NOT REGISTERED USER	The output provided by are: • Map visualisation • Position of the stations • Total number of the stalls • Name and ID of the station Map visualisation the user is allowed to know the address of each station and it's capacity.
REGISTERED USER	The output provided are: • Map visualisation • Position of the stations • Manual inserting of the user's position • Automatic localisation • List of the nearest stalls • Total number of the stalls • Number of free bikes available • Number of free stalls available • Name and ID of the station

TECHNICIAN	The output provided are: • Map visualisation • Position of the stations • Manual inserting of the user's position • Automatic localisation • List of the nearest stalls • Total number of the stalls • Number of free bikes available • Number of free stalls available • Name and ID of the station • Statistic visualisation (Table 2)

Table 13: input requirements

6.4 Navigation Hierarchy

In this section is possible to find a diagram, fig 10, of the navigation hierarchy where is shown how a user can move through the interfaces.

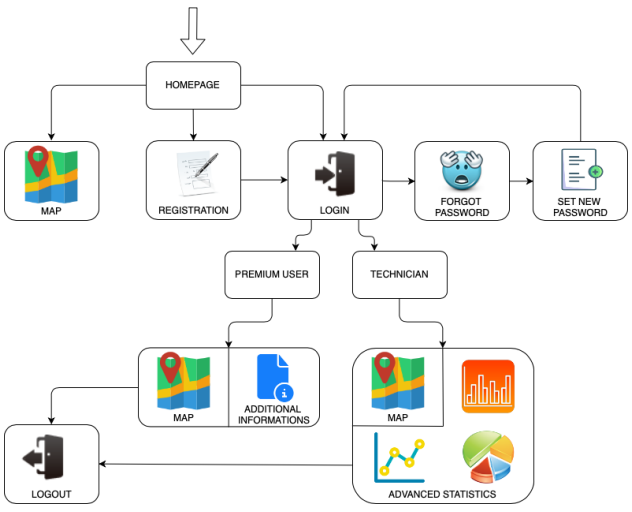

Figure 10: Diagram of the navigation hierarchy