POLITECNICO
MILANO 1863

Software Engineering Project
Lorenzo Amici, Marina Ranghetti,
Marta Rossi, Yinyao Zhang

Requirement Analysis and Specification Document

Deliverable: RASD
Title: Requirement Analysis and Verification Document
Authors: Lorenzo Amici, Marina Ranghetti, Marta Rossi, Yinyao Zhang
Version: 2.0
Date: 05-June-2019
Download page: https://github.com/MarinaZen/SoftwareEng.git

Contents

1.Introduction

The goal of this project is to develop a web-app for managing bike-sharing's service information.
Bike-sharing is a short-term bicycle rental service available in different urban locations characterized by bicycle transit.
The service is accessible through applied technology (smart cards and/or mobile phone). In particular, it offers 24h/7 service.
The development of this web application has been commissioned from Comune di Milano that will eventually release privileged permissions to obtain statistical information and other utilities. In the following a more precise description is provided.

1.1World Phenomena

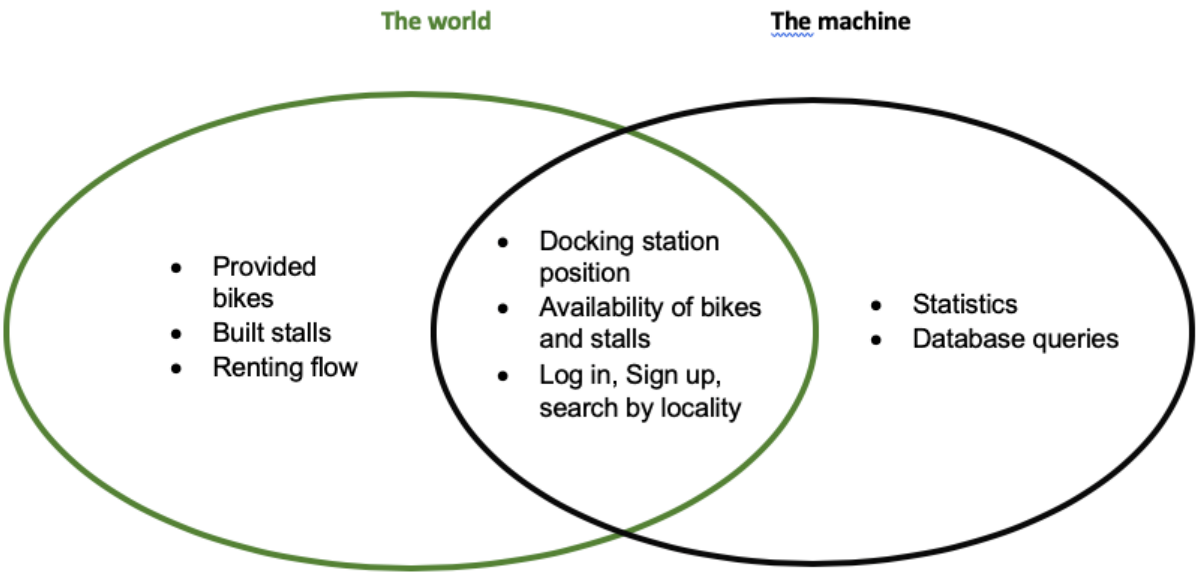


Fig.1: Wold and Machine Phenomena Scheme

World Phenomena:

- **Provided Bikes**

  Comune di Milano has provided a sufficient number of bikes to satisfy the necessity of the citizen.

- **Built Stalls**

  Comune di Milano has provided a sufficient number of stalls in strategic area of the city to satisfy according to the number of bikes.

- **Renting Flow**

  Movement of bikes in each station during the day.

Shared Phenomena:

- **Docking Station Position**

  Exact position in a specific reference system of the docking station.

- **Availability of bikes and stalls**

  The movement of bikes in each station during the day produces a different availability of bikes and stalls in every docking stations.

- **Log In, Sign Up, Search by attribute**

  Procedures the users can apply, particularly register to our app filling each field with his data and then log in.
  Once open the app he can search each docking station by name of the locality.

Phenomena located entirely in the machine:

- **Statistics**

  Having the flow of bikes in each stations during the day, it is possible to provide a full vision of this movement.

- **Database Queries**

  Retrieving data from the database (post, user data, bikes data etc...).

1.2 Scenario

Bob is common user. He wants to hire a bike for going from position A to position B. Particularly, he wants to discover if there are some docking stations near him with available bikes. Lastly when he will reach the destination, he would be sure that there are docking stations with free stalls near his position.

Alice is a registered user. She has recently used the bike-sharing app and would like to leave a comment about the service. Moreover, she wants to check the bike flows of her usual station.

John heard about this application and would like to join as a new registered user.

1.3 Use Cases

User connects to the homepage

Actors:

- Common User
- Customer

Entry condition:

- The information about the app and bike sharing service are available

Flow of events:

- User navigates to the About/Homepage section

- User can read the description and aim of the app

Exit condition:

- User closes the web app

Exceptions:

- App crash, the app provides no data loss

User reads blog's posts

Actors:

- Common User
- Customer

Entry condition:

- Blog section is available

Flow of events:

- User navigates to the blog section
- User can read the posts in the blog (starting from the recent)

Exit condition:

- User closes the web app or navigates to another section

Exceptions:

- App crashes

Visualization number of free bikes/stalls

Actors:

- Common User
- Customer
- GPS
- Map service

Entry condition:

- User wants to retrieve information about the bike service and navigates to the appropriate section

Flow of events:

- User opens the map page section
- (optional) User updates his initial position
- User can visualize the nearest docking stations
- A pop-up shows containing information about the station (bikes, stalls)
- (optional) User can insert input and final destination.
- (optional) GPS can navigate the user to the final destination

Exit condition:

- User closes the web app or navigates to another section

Exceptions:

- The GPS is not working
- App crashes

THE FOLLOWING USE CASES ARE RELATED ONLY TO A REGISTERED USER

Registration

Actors:

- Customer

Entry condition:

- Customer has to fill the registration form

Flow of events:

- Customer inserts username (unique)
- Customer inserts mail (unique)
- Customer confirms his mail
- Customer inserts password (length condition applied)
- Customer inserts confirm password
- Customer inserts check PIN (provided by Comune di Milano)

Exit condition:

- Customer closes the app
- Customer submits the form

Exceptions:

- A warning alert is shown if the customer is already registered
- A warning alert is shown for each field wrongly compiled
- App crashes

Login

Actors:

- Customer

Entry condition:

- Customer fills the login form

Flow of events:

- Customer inserts the mail
- Customer inserts the password
- (optional) Customer checks the "remember me" checkbox

Exit condition:

- Customer closes app
- Customer successfully performs the login

Exceptions:

- A warning alert is shown if the customer isn't registered

- A warning alert is shown for each field wrongly compiled
- App crashes

Retrieve password

Actors:

- Customer

Entry condition:

- Customer forgot his password for the login

Flow of events:

- Customer clicks on "I forgot my password"
- Customer inserts the email he used for registration
- Customer checks his mailbox for the assistance email
- Customer clicks the link on the received email
- Customer inserts the new password ho chose
- Customer confirms the new password

Exit condition:

- Customer closes the app
- Customer successfully submits the new password

Exceptions:

- A warning alert is shown if the customer isn't registered
- A warning alert is shown for each field wrongly compiled
- App crashes

Updating Profile

Actors:

- Customer

Entry condition:

- Logged-in customer wants to change profile settings

Flow of events:

- Customer access to the account section
- (optional) Customer changes his username
- (optional) Customer changes his email
- (optional) Customer updates his profile picture

Exit condition:

- Customer closes the app
- Customer submits the changes
- Customer changes page

Exceptions:

- A warning alert is shown if the chosen username is already taken

- A warning alert is shown if the chosen email is already taken
- App crashes

Adding posts

Actors:

- Customer

Entry condition:

- Logged-in customer enters the section for creating a new post

Flow of events:

- The new post section opens
- Customer inserts the title
- Customer inserts the post content

Exit condition:

- Customer closes the app
- Customer submits the post
- Customer changes page

Exceptions:

- A warning alert is shown if the customer doesn't insert a title
- A warning alert is shown if the customer doesn't insert any content
- App crashes

Updating posts

Actors:

- Customer

Entry condition:

- Logged-in customer wants to change the content or title of an existing post

Flow of events:

- Customer access to the blog section
- Customer clicks on his own post
- Customer clicks the update button
- (optional) Customer inserts a new title
- (optional) Customer updates the content

Exit condition:

- Customer closes the app
- Customer submits the post changes
- Customer changes page

Exceptions:

- A warning alert is shown if the customer doesn't insert a title (empty field)
- A warning alert is shown if the customer doesn't insert content (empty field)

- A warning alert is shown if the customer clicks on a post that doesn't belong to him
- App crashes

Deleting posts

Actors:

- Customer

Entry condition:

- Logged-in customer wants to delete an existing post

Flow of events:

- Customer access the blog section
- Customer clicks on his own post
- Customer clicks the delete button
- (optional) Customer confirms the deletion
- (optional) The post is canceled

Exit condition:

- Customer closes the app
- Customer submits the deletion form
- Customer changes page

Exceptions:

- A warning alert is shown if the customer clicks on a post that doesn't belong to him
- App crashes

Displaying statistical information

Actors:

- Customer
- GPS
- Map service

Entry condition:

- Logged-in customer wants to visualize statistics about the docking stations

Flow of events:

- Customer access to the map section
- Customer clicks on a station
- A pop-up is shown containing information about the station (bikes, stalls)
- Customer clicks on the statistic button
- Customer is redirected to the page containing statistics information

Exit condition:

- Customer closes the app

Exceptions:

- The login button is shown instead of statistic button if the customer is not logged-in

- App crashes

2.Overall description

Here is the description of the functionalities of our web-app.
First it is necessary to distinguish between users that are/are not registered.
Both can visualize a map with all the docking stations: by selecting one of them the user can come to know the station's exact position, the number of available bikes and stalls and other information.

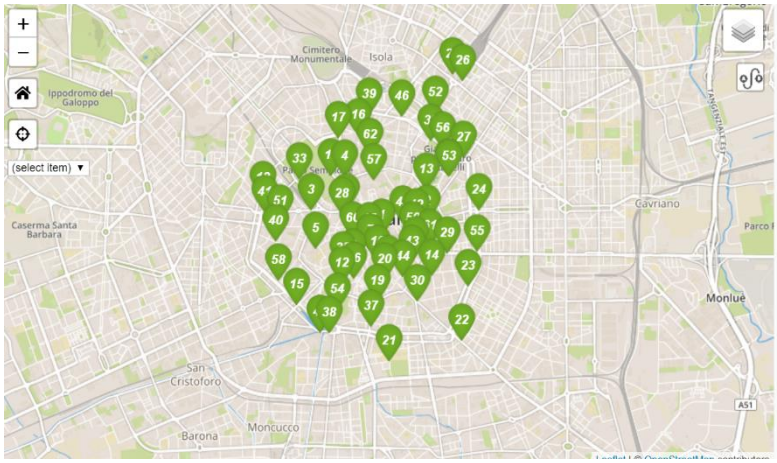


Fig.2: Overview of the map embedded in the web application.

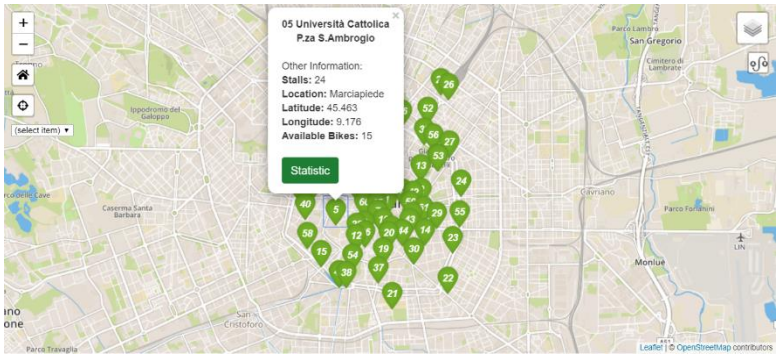
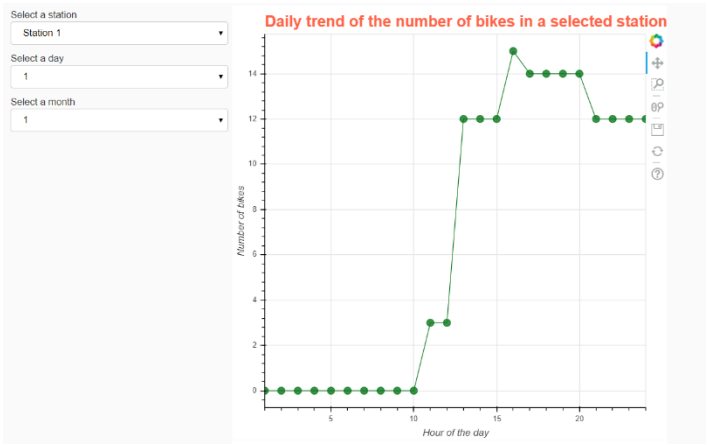


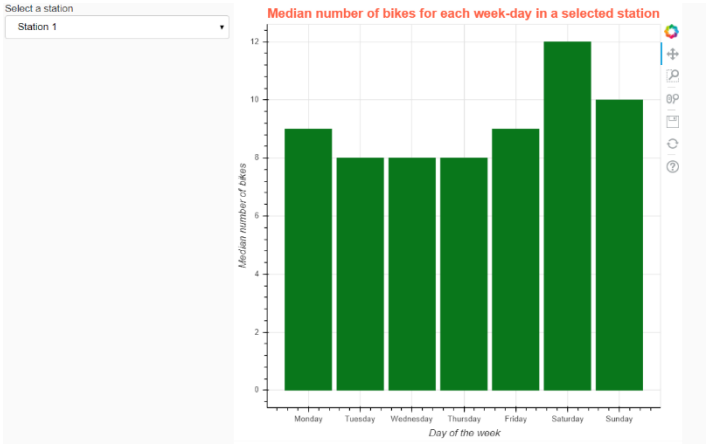
Fig.3: Example of a popup displaying information of a station.

The registered user will be able to and participate to the blog and visualize statistical information. In particular, a logged-in user can visualize three main interactive plots:

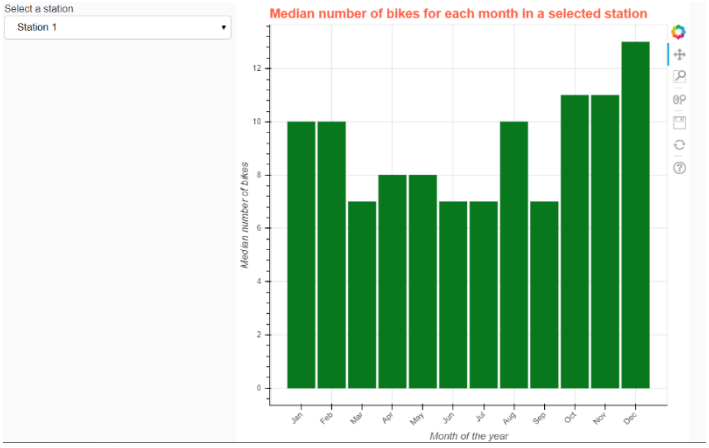
- The first one provides 3 dropdown menus in order to select the station number, the month and the day, and outputs the bike flow in that station for 24 hours



- The second one provides a dropdown menu in order to select the station number and outputs the median number of bikes for that station in each day of the week



- The third one provides a dropdown menu in order to select the station number and outputs the median number of bikes for that station in each month of the year



3. Specific requirements

3.1 Technical requirements

The system should be implemented in:

- Python
- HTML, CSS, JS
- Easy interface
- English languages
- Software for managing databases

3.2 Non-functional requirements

- The system should be available 24h/7
- The system should be updated in real time
- Supportability on mobile phones

3.3 Functional requirements

- The system shall allow multiple users to access a same service at the same time
- The system shall allow users to access the basic services (availability of bike and stalls) without any registration needed
- The system shall allow registered users to enter the privileged area
- The system shall allow registered user to see statistics about bike movements

4. Effort Spent

Marina Ranghetti: RASD, DD and web application (website and map)
Marta Rossi: RASD, DD and web application (website and code comments)
Lorenzo Amici: RASD, DD, statistics and their implementation
Yinyao Zhang: DD, testing of use cases and code comments