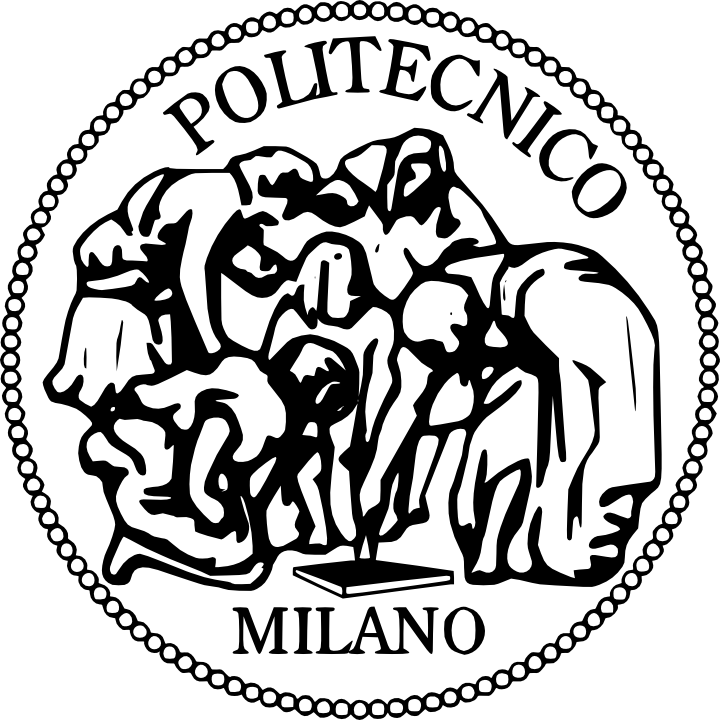
**Oleggo**

**Design Document**



Falci Angelo Gallego García Julián David

Politecnico di Milano January 22, 2018

CONTENTS

[1. Chapter 1 - Introduction - 5 -](#_Toc504431221)

[1.1. Purpose - 5 -](#_Toc504431222)

[1.1.1. Intended audience - 5 -](#_Toc504431223)

[1.2. Scope  - 5 -](#_Toc504431224)

[1.3. Definitions, acronyms, abbreviations Reference documents  - 6 -](#_Toc504431225)

[1.3.1. Definitions - 6 -](#_Toc504431226)

[1.3.2. Acronyms - 6 -](#_Toc504431227)

[1.3.3. Abbreviations - 6 -](#_Toc504431228)

[1.4. Goals - 7 -](#_Toc504431229)

[1.5. Architecture and tool chosen - 7 -](#_Toc504431230)

[1.6. Overview - 8 -](#_Toc504431231)

[2. Chapter 2 - Architectural Design - 9 -](#_Toc504431232)

[2.1. Overview - 9 -](#_Toc504431233)

[2.2. Architectural Style and Patterns - 9 -](#_Toc504431234)

[2.3. High level components and their interaction - 11 -](#_Toc504431235)

[2.4. Deployment View - 11 -](#_Toc504431236)

[2.5. Runtime view - 12 -](#_Toc504431237)

[2.6. Component interfaces - 14 -](#_Toc504431238)

[2.7. Use case functional requirements analysis - 14 -](#_Toc504431239)

[2.7.1. Insert new book - 14 -](#_Toc504431240)

[2.7.2. Update bookmark - 15 -](#_Toc504431241)

[2.7.3. Insert a note - 16 -](#_Toc504431242)

[2.7.4. Search word definition - 16 -](#_Toc504431243)

[2.7.5. Change state of book - 17 -](#_Toc504431244)

[2.7.6. See all favorite notes - 17 -](#_Toc504431245)

[2.7.7. See General dictionary - 18 -](#_Toc504431246)

[2.7.8. See the notes of specific book - 18 -](#_Toc504431247)

[2.7.9. See the words definitions of a specific book - 18 -](#_Toc504431248)

[2.7.10. Remove a word - 19 -](#_Toc504431249)

[2.7.11. Remove a note - 19 -](#_Toc504431250)

[2.7.12. Remove a note from favorite notes - 19 -](#_Toc504431251)

[2.7.13. Add a note to favorite notes - 20 -](#_Toc504431252)

[2.7.14. Modify an existing note - 20 -](#_Toc504431253)

[2.7.15. Search a specific word in dictionary - 20 -](#_Toc504431254)

[2.8. Sequence diagrams - 21 -](#_Toc504431255)

[2.8.1. Add new books - 21 -](#_Toc504431256)

[2.8.2. Update bookmark - 22 -](#_Toc504431257)

[2.8.3. Add new note - 23 -](#_Toc504431258)

[2.8.4. Search word definition - 24 -](#_Toc504431259)

[2.8.5. Change the state of a book - 25 -](#_Toc504431260)

[3. Chapter 3 - Class diagram - 26 -](#_Toc504431261)

[3.1. Database - 28 -](#_Toc504431262)

[3.2. Pages - 28 -](#_Toc504431263)

[4. Chapter 4 - User Interface Design - 30 -](#_Toc504431264)

[4.1. Home Page - 30 -](#_Toc504431265)

[4.1.1. Library - 30 -](#_Toc504431266)

[4.1.2. *Favorite Notes* - 30 -](#_Toc504431267)

[4.2. Book - 32 -](#_Toc504431268)

[*4.2.1.* *Information* - 32 -](#_Toc504431269)

[*4.2.2.* *Notes* - 34 -](#_Toc504431270)

[4.2.3. Dictionary - 35 -](#_Toc504431271)

[4.2.4. Update progress screen - 36 -](#_Toc504431272)

[4.2.5. Reading Book - 37 -](#_Toc504431273)

[4.2.6. Dictionary - 38 -](#_Toc504431274)

[4.3. New Entry - 39 -](#_Toc504431275)

[4.4. New Book - 41 -](#_Toc504431276)

[4.5. Side Drawer menu - 42 -](#_Toc504431277)

[4.6. Complementarty screens - 43 -](#_Toc504431278)

[5. Requirements Traceability - 44 -](#_Toc504431279)

[5.1. Software System Attributes - 44 -](#_Toc504431280)

[5.1.1. Reliability - 44 -](#_Toc504431281)

[5.1.2. Availability - 44 -](#_Toc504431282)

[5.1.3. Security  - 44 -](#_Toc504431283)

[5.1.4. Maintainability - 44 -](#_Toc504431284)

[5.1.5. Usability - 44 -](#_Toc504431285)

[5.1.6. Portability - 45 -](#_Toc504431286)

[6. Software used - 45 -](#_Toc504431287)

[7. References - 45 -](#_Toc504431288)

# Chapter 1 - Introduction

## Purpose

This document is intended to help understand and evaluate design and prototyping steps taken into account for the development of the mobile application call “Oleggo” (named from olè exaltation of being good at something and leggo from the verb leggere Italian word for read), it explains both the application domain and the system for further understanding. It explains the features of Oleggo, the main components we used and the interface we implemented for the user.

This project is implemented as an outcome of the course “Design and Implementation of mobile applications”.

### Intended audience

This document is realized for all people participating in and/or supervising the project:

* Expected audience of this document is the developers and anyone who in-tends to develop on this program
* Developers who can review project's capabilities and more easily understand where their efforts should be targeted to improve or add more features to it (it sets the guidelines for future development).
* Project testers can use this document as a base for their testing strategy as some bugs are easier to find using this document. This way testing becomes more methodically organized.
* End users of this application who wish to read about what this project can do.

## Scope

The goal of Oleggo is to create an application that can be used by students to take notes or look for a word in a fast way to reduce the dead times. An other advantage is that the students can have always with them their notes and a personal dictionary with the words that were more complicated for them. They also can save like favorite the most important note in order to find them in the home page.

But it can also be used by anyone that want to save in own phone all books he/she is reading or read in the past and, for each book, save the notes inserted or word searched and also the bookmark. To take notes with Oleggo will be fast mostly because it is possible insert note or word using a speech recognized alternatively to write it.

## Definitions, acronyms, abbreviations Reference documents

### Definitions

**Application:** A computer program designed for a specific task or use.

**Architecture: <**system> fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution

**Queue:** In programing:

**a.** A sequence of stored data or programs awaiting processing.

**b.** A data structure from which the first item that can be retrieved is the one stored earliest.

**Request:** The act or an instance of asking for something.

**Requirement:** something that is needed or that must be done. Something that is necessary for something else to happen or be done.

**System:** A group of interacting, interrelated, or interdependent elements forming a complex whole, especially:

**a.** A network of structures and channels, as for communication, travel, or distribution.

**b.** A network of related computer software, hardware, and data transmission devices.

**User:** a person or thing that uses something.

**Tier:** a level or grade within the hierarchy of an organization or system.

### Acronyms

**GPS:** Global Positioning System.

**UML:** Unified Modelling Language.

**IEEE:** Institute of Electrical and Electronics Engineers.

**RASD:** Requirements analysis and specifications Document.

**EIS:** Enterprise information systems.

**JEE:** Java enterprise edition.

### Abbreviations

**App:** Application.

## Goals

The system has to provide these features:

* [G1] User can insert a new book inserting the ISBN by writing it or reading it using the bar-code reader in the app.
* [G2] For each book inserted, user can insert and modify a bookmark.
* [G3] For each book inserted user can insert a note by writing it or using speech recognize, and he/she can also add the page of the book. The note will be added to the “main active” book (explained further) .
* [G4] For each book inserted user can search the meaning of a word and add it in her/his personal dictionary. The word will be added to the “main active” book so user have a global dictionary and many little dictionary for each book.
* [G5] User can change the state of a book in “passive”, “active” (book user is reading and can change bookmark), “main active” (user can have only one main active book, every word searching or notes added will be connected to this book).
* [G6] User can choose to see all favorite notes or all notes connected to a certain book.
* [G7] User can choose to see all words in the dictionary or all words connected to a certain book.

## Architecture and tool chosen

To develops Oleggo we used the tool Nativescript.

This tool can be used for cross-platform development, so our app could be run in Android or IOS system (at the moment the changes necessary to IOS interface hasn’t been done, this will be added to future work). It is possible code for Nativescript in JavaScript or in TypeScript, for Oleggo we chose the first. Other languages used for the graphics are XML and CSS.

*Why NativeScript?*

There are a lot of motivations.

The main reason is that it offers many powerfully plugins to implement things, like speech recognition, that in other framework are hard to do (we will show all the plugins used).

An other reason is for the graphics. NativeScript offers a really easy implementation of the components for developing the user interface also due to the fact that we have previously worked with Javascript and XML for develop of websites allows an easy way to implement the graphics for the application.

Another motive is due to the fact that Nativescript allows you to upload the changes live on the emulator or cellphone where the app is running shorts the time of developing in a consider way.

Finally, they are libraries and templates for mobile application developed by the community that allows us to implement complex interfaces that without would take much more time. We combine some of them to implement beautiful graphics components that, without these templates, are very hard to do so beautiful (for example the side drawer menu).

## Overview

This document is divided in four parts with clean and non-ambiguity description of the whole system:

* **Chapter 1:** brief description of the idea, goals and objective of the document.
* **Chapter 2:** Explanation of the main functionalities with constraints, assumptions and hardware dependencies.
* **Chapter 3:** Specifications of the system functionalities and non-system functionalities possible scenarios.
* **Chapter 4:** Use cases and UML models

# Chapter 2 - Architectural Design

## Overview

In this section it is provided detailed information of the product structure and the implementation of the general structure. In the following sections it is provided details of why it was decided the architecture for the system, what are the characteristics of the architecture that make it the chosen one

Then it continues talking about the general components of the architecture and the subcomponents of each component and their interactions. Following by information of all the functions that do each subcomponent.

After the primary information on how it is constituted the architecture, its explained how all these components work to make the system to work effetely.

The chapter finish with an explanation about the interface and how some components interact with it.

## Architectural Style and Patterns

The system will be developed by using a general client-server 3-tier Architecture. This is because this is a service that its expect to have a lot of simultaneous request from visitors, passenger and Taxi drivers. Now it is present a general description of the architecture proposed for the system and furthermore, also a small description for each of the components.

The design of the architecture is based on a client-server with a fat client and fat server, this decision was made because with the actual capacities of the cellphones and computers, we want to take advantage of it and make part of the calculus and processing on the device of the costumer to reduce the time of respond of the application, improve the performance and to be able to respond all the simultaneous request in the minor time possible. For the system was decided a 3-tier distributed architecture to distribute the operations on visualization, logic and database, each tier is described as follows:

**Client tier:** This is an architectural layer whose job is translating the user actions and presenting the output of tasks and results into something the user can understand, in this case, in My taxi service is a web and mobile application.

**Business Logic tier:** This is an architectural layer whose job is coordinate the application, processes commands, perform calculations, make logical decisions and evaluations. It also moves and processes data between the client and EIS tiers.

**EIS tier:** This is an architectural layer whose job is to provide an abstract interface to information storage mechanism, is in charge of storing and retrieving information from the database.

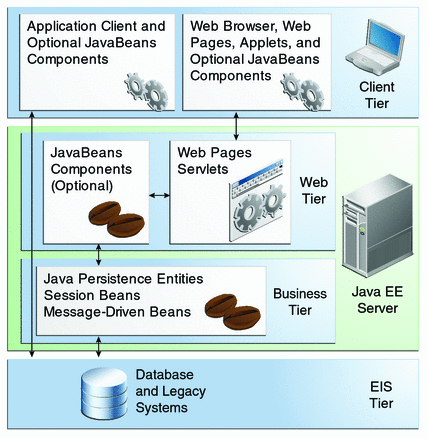
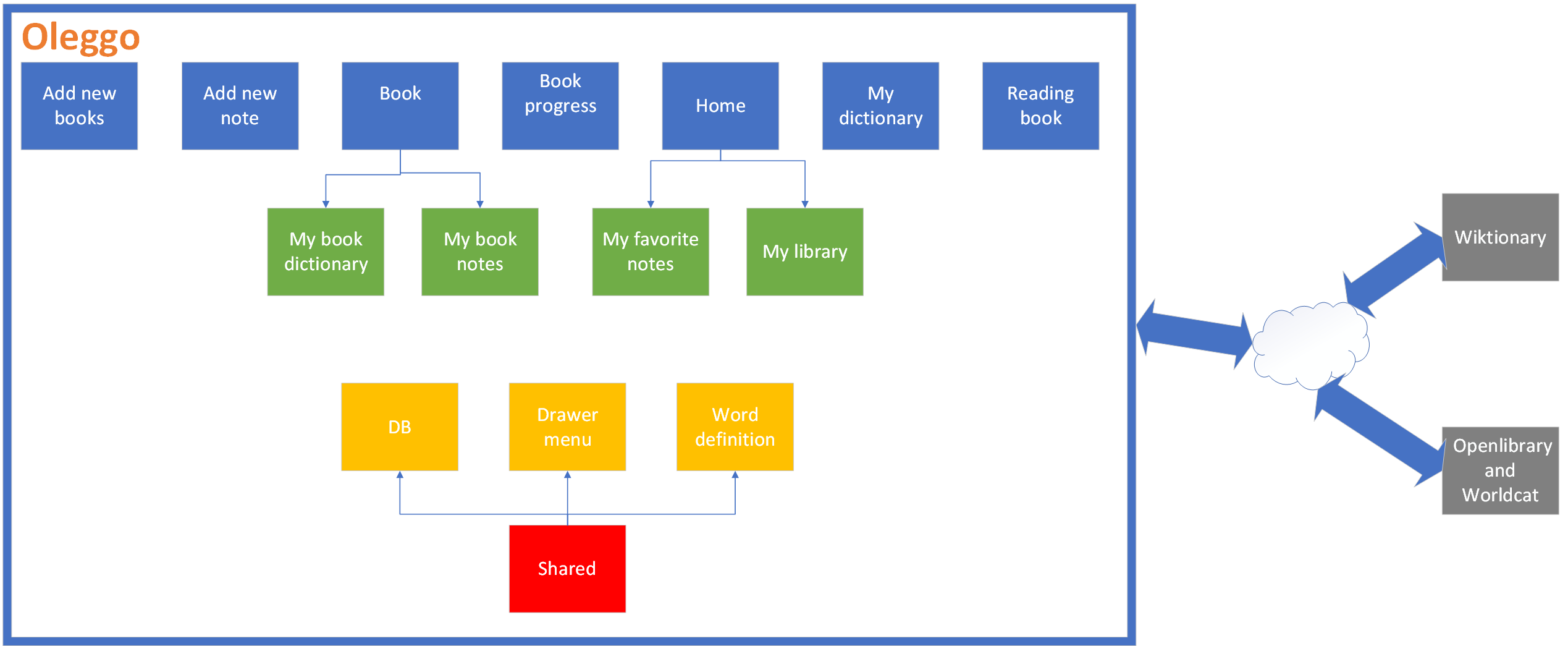


Figure 1. System architecture [1].



We will see all blue and green component in the chapter “User Interface”.  
The shared component aren’t visible for the user, they are only used by the pages above:

DB → contain all the query to the database, in this way if we modify something to database we must modify only this class

Drawer Menu → it is the hide menu that appear in every page if user click button on the left-top or shift the finger from left to right

Word Definition → it’s include a set of pages that manage the searching of words in Wikitionary

## High level components and their interaction

As described before, the system has 3 high level components that are the tiers of the architecture:

-The client.

-Business logic.

-EIS.

These components manage all the operations made on the system asynchronously, this are started by a request (signal) from the tier that wants to start an operation to the the tiers that needs to accomplish the operation.

The first component, **the client**, has the possibility to start communication directly with the two other tiers, with logic when is trying to pull a request for a taxi/Search a costumer or when the driver wants to change his status, and with the EIS, when the client wants to manages his account information.

The second component, **logic**, manages all the queues of the city, checks the position of every taxi using the GPS of each driver mobile. When the client component sends a signal of status change from a driver, it also is in charge to add or remove the taxi to the queue and send a signal to the EIS to save the changes. Keeps the drivers in the correct queue depending of the position. When the client component sends signals of request a taxi, assigns a taxi to each passenger and sends them the current location of the taxi that accepted the ride when they are waiting the arrival, this is possible by sending a signal to the Client component to displayed on the user interface.

**The EIS tier** is the component that has access and manages all the information about the passengers and drivers, including also the queues information, this tier is entirely reactive from the signals from the logic or client components.

## Deployment View

The following diagram is the deployment view of the system, it shows the relations that are between the components and subcomponents, giving the idea of the subcomponents that are need it to do a task on the system, dependencies and the route through the subcomponents that follows actions in the system.

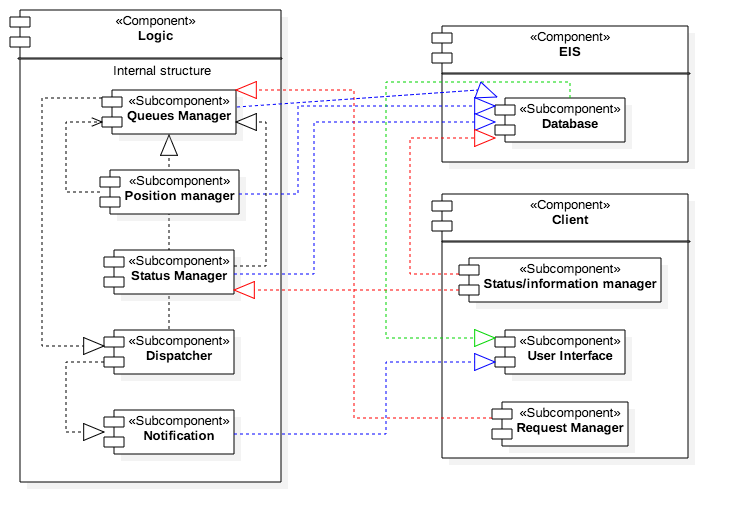


Figure 2. deployment View.

## Runtime view

In the following figure is shown the possible actions that can be done in the system. With the actions or functionalities that can be done in the system shown in the image and based on the deployment view, its possible to understand more about the architecture of the system with the help of the sequence diagrams that explain how functionalities are accomplish by the combination of work of the subcomponents. In the following numerals are explained by sequence diagrams each one of these functionalities.

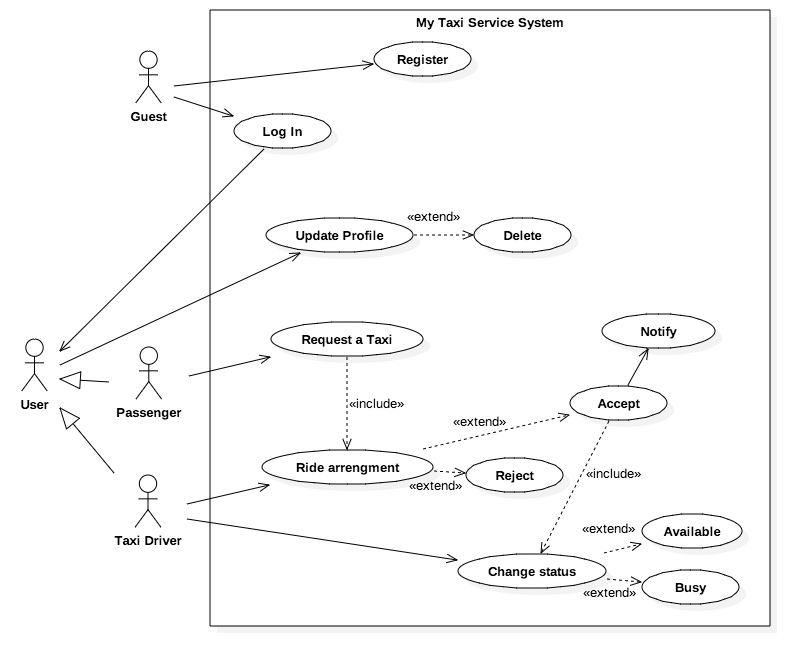


Figure 3. User interactions with the system.

## Component interfaces

**Client Tier**: Is composed of the XHTML pages that the normal user will see or cellphone interfaces. This is strictly related to the logic Tier, that is the one that provides all the info displayed here.

**Logic Tier**: Includes Web servlets and the web beans. This tier receives the requests of the client tier and has some beans which listen to these events and display data regarding the user requests. They retrieve the information. This tier also has all the logic underlying our application; it is responsible of communication with Client Tier and EIS Tier.

**EIS Tier**: Is composed of the entity beans and database of the system that I call “database” in general, the beans represent the connection to the database, which connects to the Database, to insert, update, delete, select. The database itself is composed of tables composing the database we generated from the needs of the project.

## Use case functional requirements analysis

This part of the document shows how users interact with Oleggo in order to reach the different goals. We will see also the flow of actions the user needs to do but we not see the system internals.

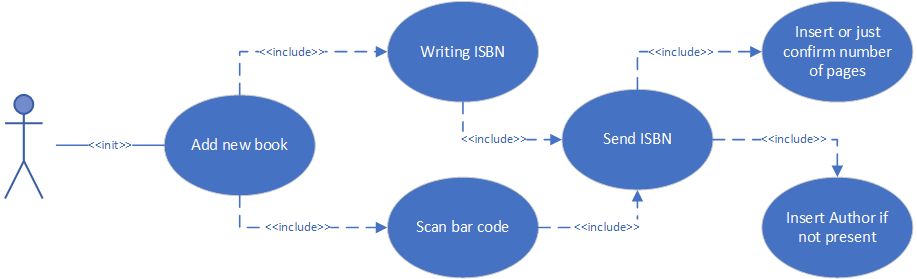
We can identify only one group of actors involved in this application:

* Users – who had downloaded and installed the application Oleggo

For the most important, and complex, functions there is the graphics representation of user case.

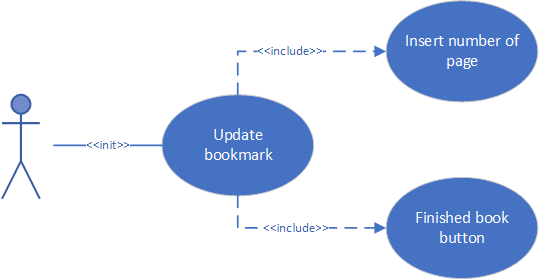
### Insert new book

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | Insert a new book using ISBN |
| **Event flow** | Press New Book voice in the drawer menu  Insert the ISBN of book writing it or scanning the bar code  Press Send ISBN  If author is not present in online library insert it and continue  Confirm (or insert if it isn’t present) the number of page in the book and continue |
| **Output conditions** | Book is inserted |
| **Exception** | User insert a not valid number of pages (for example insert letters instead of numbers)  Insert a not valid ISBN |



### Update bookmark

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | Update the bookmark of a book |
| **Event flow** | Press Home or Reading Books in the drawer menu  Press in the book we want to modify bookmark  Press the circle button at left side  Insert a number and press update progress or press “I have finished button” |
| **Input conditions** | Book already added |
| **Output conditions** | Bookmark updated  Book state changed and become “read” |
| **Exception** | / |



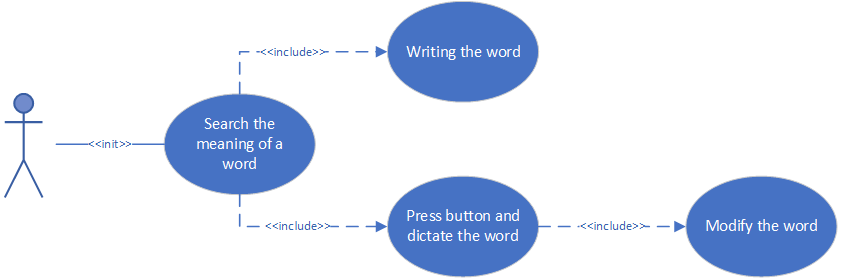
### Insert a note

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | Insert a note connected to “main active” book |
| **Event flow** | Press New Note in the drawer menu  Insert the note writing it or using speech recognized (pressing “press and speak” and speak) and eventually modify it  Optionally it possible insert a number of page  Press add quote |
| **Input conditions** | One book is “main active” |
| **Output conditions** | Quote added and connected to the “main active” book (on default the quote isn’t favorite) |
| **Exception** | If there isn’t any “main active” book the quote isn’t added |



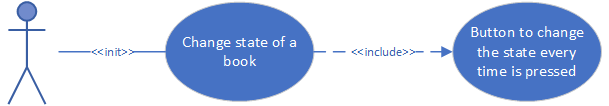
### Search word definition

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | Search and save the meaning of a word connected to “main active” book |
| **Event flow** | Press New Note in the drawer menu  Insert the word writing it or using speech recognized (pressing “press and speak” and speak) and eventually modify it  Press look for the word |
| **Input conditions** | One book is “main active” |
| **Output conditions** | Word’s meaning saved and connected to the “main active” book |
| **Exception** | If there isn’t any “main active” book the word isn’t added |



### Change state of book

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | Change the state of a book (passive, active, main active or read) |
| **Event flow** | Press Home or Reading Books in the drawer menu  Press in the book you want to change the state  Press in the big, center button more times to change state in “main active”, “active” or “passive”  If you finish the book see the “update bookmark” table above |
| **Input conditions** | Book already added |
| **Output conditions** | The state of the book is changed |
| **Exception** | / |



### See all favorite notes

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | See all favorite notes |
| **Event flow** | Press Home→My Favorite Notes in the drawer menu |
| **Input conditions** | There is at least one favorite note |
| **Output conditions** | Show all favorite notes |
| **Exception** | / |

### See General dictionary

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | See all words searched and saved |
| **Event flow** | Press My Dictionary in drawer menu |
| **Input conditions** | There is at least one word |
| **Output conditions** | Show all words in the dictionary with their meaning in alphabet order |
| **Exception** | / |

### See the notes of specific book

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | See all notes connected to a specific book |
| **Event flow** | Press Home or Reading Books in the drawer menu  Press in the book you want to see the connected notes  Press My Notes in the tab menu on top of the page |
| **Input conditions** | Book and notes for this book added |
| **Output conditions** | Show all notes connected to a specific book |
| **Exception** | / |

### See the words definitions of a specific book

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | See all words, and their meaning, connected to a specific book |
| **Event flow** | Press Home or Reading Books in the drawer menu  Press in the book you want to see the connected notes  Press My Dictionary in the tab menu on top of the page |
| **Input conditions** | Book and word searched for this book added |
| **Output conditions** | Show all words and their meaning connected to a specific book |
| **Exception** | / |

### Remove a word

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | Remove a word from dictionary |
| **Event flow** | Go in the page to see all words in dictionary or only the words connected to a specific book (see the tables above for details)  Press remove button near the word we want remove |
| **Input conditions** | Word already searched |
| **Output conditions** | If you are in global dictionary you remove from dictionary all words equal the word you remove  If you are in the dictionary connected to a specific book you remove only the word connected with this book (it means that if a word is connected to 2 books you remove only one occurrence of this and the word is present yet in the global dictionary) |
| **Exception** | / |

### Remove a note

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | Remove a note |
| **Event flow** | Go in Home→My Favorite Notes or in the notes of a specific book (see the tables above for details)  Press delete button near the note you want to remove |
| **Input conditions** | Note already inserted |
| **Output conditions** | The note is removed |
| **Exception** | / |

### Remove a note from favorite notes

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | Remove a note |
| **Event flow** | Go in the notes of a specific book (see the tables above for details)  Press “not favorite” button near the note you want to remove from favorite notes |
| **Input conditions** | Word already searched and added to favorites |
| **Output conditions** | The note is removed from favorite notes |
| **Exception** | / |

### Add a note to favorite notes

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | Remove a note |
| **Event flow** | Go in Home→My Favorite Notes  Press “favorite” button near the note you want to remove from favorite notes |
| **Input conditions** | Note already added |
| **Output conditions** | The note is removed from favorite notes |
| **Exception** | / |

### Modify an existing note

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | Modify an existing note |
| **Event flow** | Go in Home→My Favorite Notes or in the notes of a specific book (see the tables above for details)  Modify what you want in the note you want to modify  Press the button save near the chosen note |
| **Input conditions** | Note already added |
| **Output conditions** | The modification of this note is saved |
| **Exception** | / |

### Search a specific word in dictionary

|  |  |
| --- | --- |
| **Actor** | Users |
| **Goal** | Search a specific word in the global dictionary |
| **Event flow** | Go in global dictionary to see all the words (see table above for further details)  Insert in Search box on top of the page the word you want to search the meaning or a part of this word |
| **Input conditions** | Words already searched |
| **Output conditions** | Show all the word that included what you wrote in the search box |
| **Exception** | / |

## Sequence diagrams

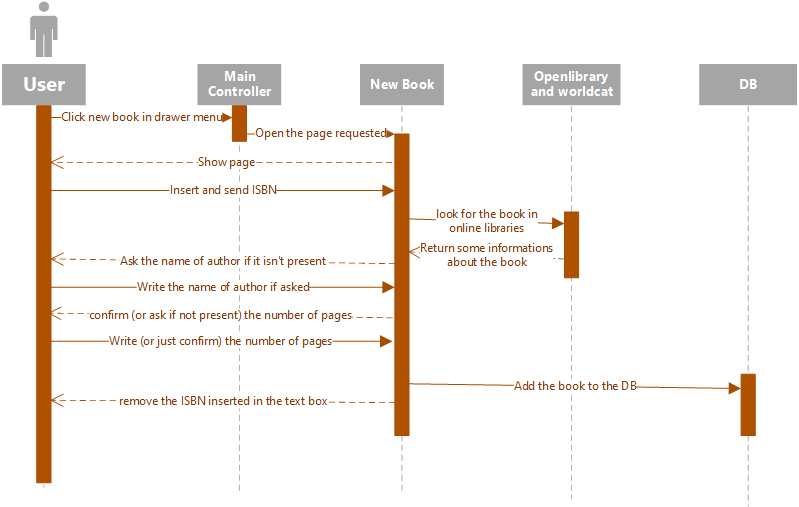
In this chapter we will model the flow of logic within our system in a visual manner that will enable us to document and validate our logic for further analysis and design purpose. We will present the most important interactions between the system and the user.

### Add new books

User can add a book press “new book” voice in drawer menu. Then user can add ISBN writing it or scanning bar code.

Online libraries don’t have all information about books, for this reason sometimes app ask to the user to insert the name of the author. It also asks to confirm, or insert if it isn’t available, the number of pages that contain the book.

When the book is added an alert advice user and the text box with ISBN will be cleaned.

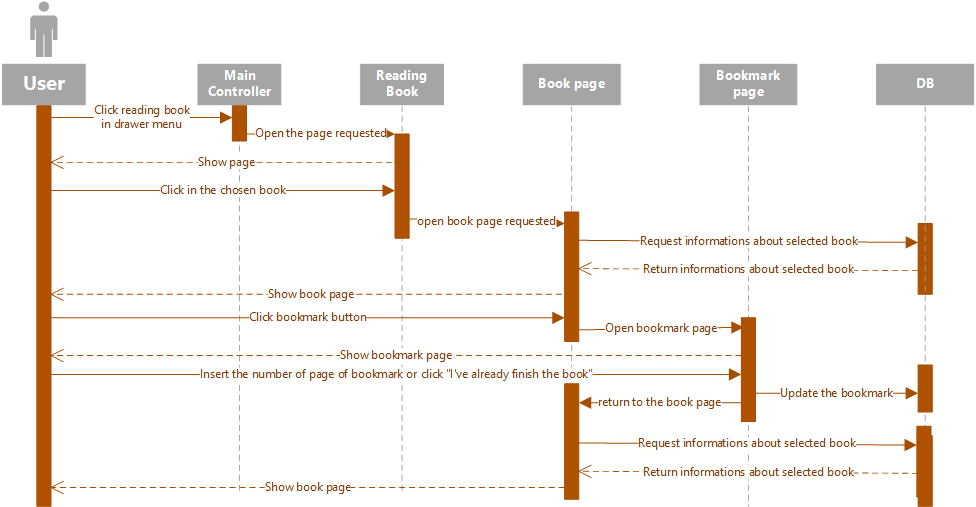


### Update bookmark

In the personal page of book user user can update the bookmark of a specific book.

User must click the little button at left and insert the number of page where put the bookmark.

Finally press “update progress” to confirm the new bookmark or “I have finished already the book”. If user insert a number greater than the number of pages that has the book the bookmark is set to the last page of book.



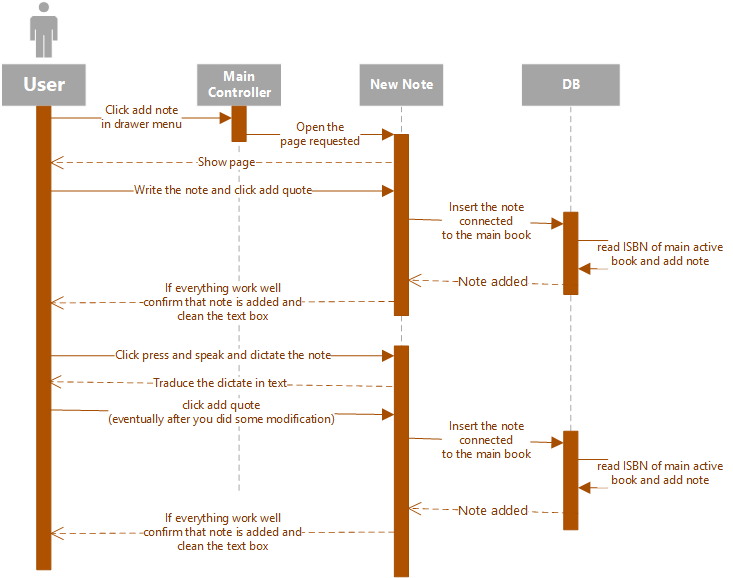
### Add new note

User can add a note pressing “new note” voice in drawer menu.

It can write the note or dictate it using speech recognize. After dictate the note user can modify the text that appear after the translation from voice to text.

If there isn’t a “main active” book note can’t be added to database.

If the note is added successfully an alert advice the user and the text box will be cleaned.



### Search word definition

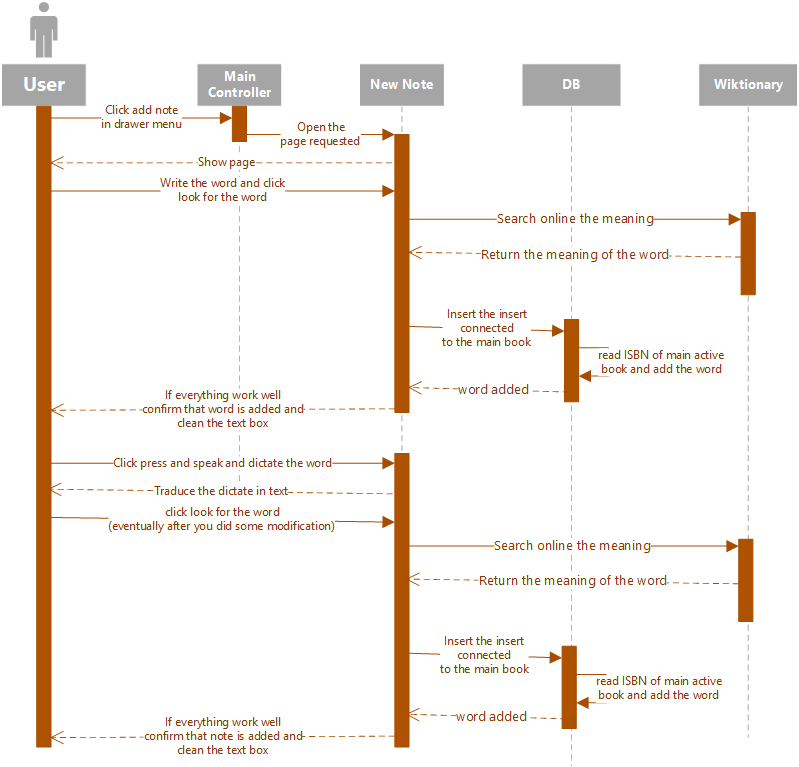
User can search a word pressing “new note” voice in drawer menu.

It can write the word or dictate it using speech recognize. After dictate the word user can modify the text that appear after the translation from voice to text.

If there isn’t a “main active” book word can’t be added to database.

If the word is added successfully an alert advice the user and the text box will be cleaned.

Finally, if user look for a “strange” word, or a word that doesn’t exist in Wiktionary, the word is saved in dictionary without the meaning.



### Change the state of a book

In the personal page of book user user can change the state of the book.   
It can be:

* passive → book doesn’t appear in “reading book” page
* active → book appears in “reading book” page
* main active → book appears in “reading book” page and if user add new note or word they will be connected to this book; only one book for times can be “main active”



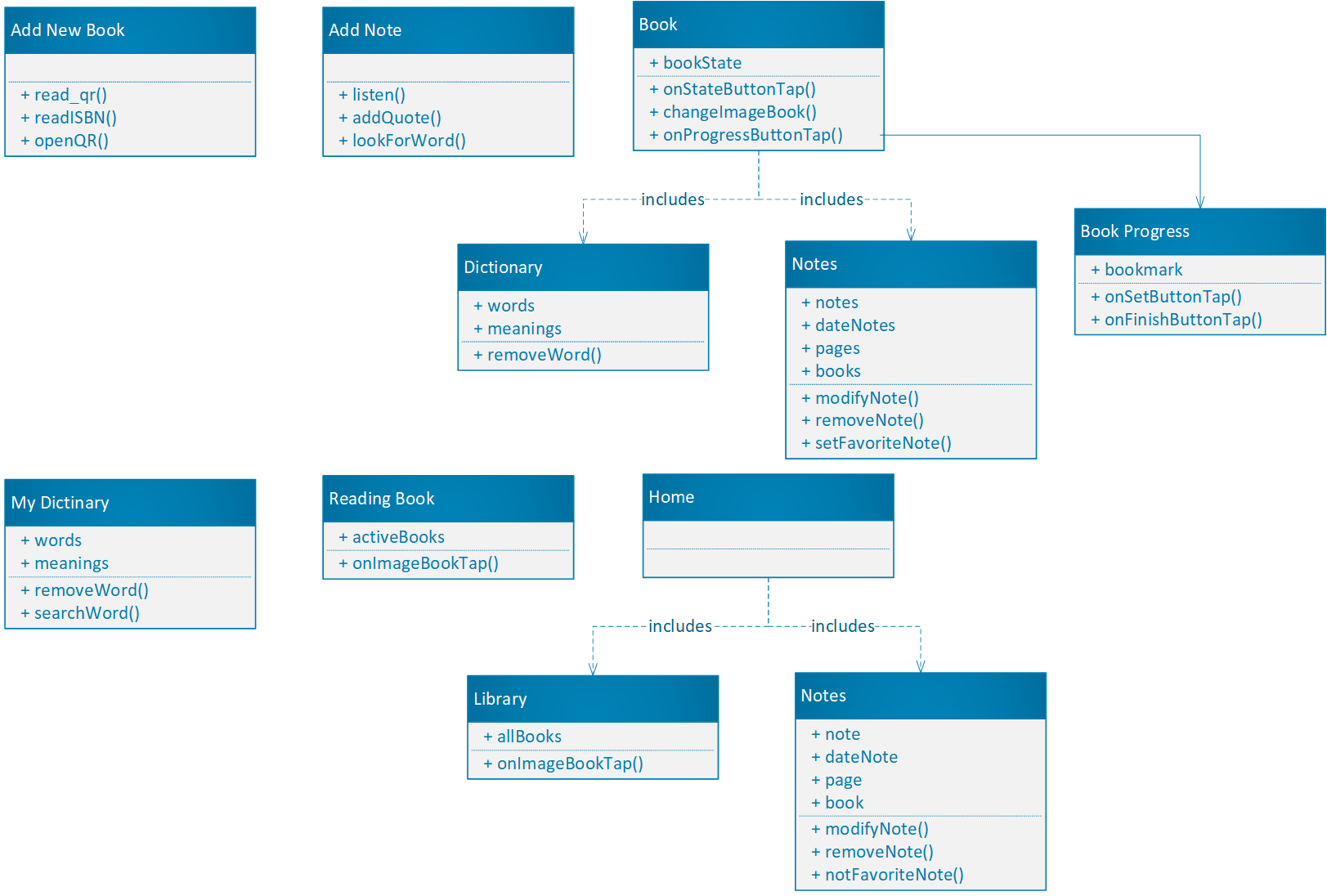
# Chapter 3 - Class diagram

In this part of document, we will see the structure of the database and details about the tables architecture.

Then we will see how is structured the entire project. For NativeScript is used to use the MVVM pattern (Model View View Model) where:

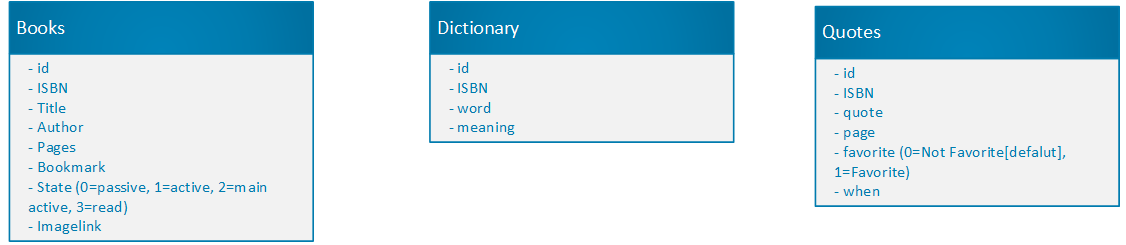
* *Model*: The model defines and represents the data. Separating the model from the various views that might use it allows for code reuse.
* *View*: The view represents the UI, which in NativeScript is written in XML. The view is often data-bound to the view model so that changes made to the view model in JavaScript instantly trigger visual changes to UI components.
* *View Model*: The view model contains the application logic (often including the model), and exposes the data to the view. NativeScript provides a module called 'Observable', which facilitates creating a view model object that can be bound to the view.

In Oleggo we use MVVM but with some modifications. We preferred to have a folder for each page of our app and sub folders if a page included more pages. We choose this because each page use data in different ways and in order to have a more organization and scalability because someone can work only to a page without modify the other. It’s mean that each view has only one model and each model has only one view but in order to don’t have a different class in which database is called we create shared components that include all possible queries need in the pages and every page can use it when want to interact with the database. In this way if we want modify something, like the name or type of the columns, in the database we must work only in this class.



## **Database**

The database is used to organize all the information important for the application for the management of the books and notes, for that reason we created a local database using sqlite and we divided the information into tree tables one for the books, one for the dictionary and one for the quotes.



## Pages

In the model graph we can see all pages that has Oleggo. For each page it is visible:

* Title.
* Visible data in that page.
* Methods that user can call interacting with that page.

Moreover, in each page is possible use drawer menu shifting the fridge from left to right or pressing the button on top left.

Now, for each page, we see a brief description:

* **Home →** This is the first window appear when user open Oleggo, it includes two pages selectable using the tab menu. In “Library” we can see all books inserted in Oleggo in alphabet order independently of their state. In the other page, “Notes”, we can see all favorite notes inserted in Oleggo order by the date of insertion.
* **Add New Book →** In this page we have ad edit text where we can input the ISBN of book we want inserted. We can also use the bar code reader pressing an other button to read the ISBN.
* **Add Note →** In this page we can add a note connected to the “main active” book. We can write the note or using the speech recognition (and eventually modify the text translated from the voice). When the note is added successfully an alert advices us and the text box will be cleaned while nothing happen if we don’t have a “main active” book.
* **Book →** In the book page we can see the information about a single book, the page where is the bookmark , how many notes and word are connected to this book and the state of this book. Moreover we can change the image of book from this page, it’s useful if the online library didn’t found any image. The last thing we can do in this page is open the “Book Progress” page.

Then from this page we can move to other two pages using the tab menu on the top of the page, “Dictionary” and “Notes”. The first page contains all words in dictionary connected to this book and we can chose to remove one of them (but if this word is connected to other words in the global dictionary we will continue to see it). The second page contain all notes connected to this book, in this page we can do the same things we can do in the “Notes” page included in “Home”. The only difference is that here we can add a note in the preference notes rather than remove it from preference notes.

* **Book Progress →** In this page we can modify the page where is the bookmark and we can set the “read” state for book when we finish to read it.
* **My Dictionary →** In this page we have all words added in the every book with its meaning order in alphabetic way. Moreover we can search a specific word or a set of words writing a part of it or them. Finally from this page we can remove a word definitely, it’s means that if we remove here a word we will not find them in the “Book” page because we remove each occurrence in the database of this word.
* **Reading Books →** In this page we can see the list of all “active” book and, if we press in a book, we can reach the “Book” page with all information about the page chosen. This page is useful because we can see only the “active” book and not all book added in the past and find fastly a specific book we are reading in this moment.

# Chapter 4 - User Interface Design

To see all the interfaces of the system, refer to the RASD document section 3.1. Where are shown and explained all the interfaces for the passengers and the taxi drivers, presenting the Web version and the one for the mobile app.

In the next figure it’s seen the general operation of the system, for any visitor, the idea with this is to give a general idea of the flow that can be done in the system, from the time that a visitor uses the system for first time until when the user erases his account, passing by the request of a taxi or the modifying of his personal information.

In this chapter we will see the user interface of each page showing how use the functionalities of Oleggo we saw in the previous chapters.

## Home Page

This is the first page that user see when open Oleggo. It is divided in two sub-pages using a tab menu:

### Library

This page shows all books inserted in Oleggo independently from their status.   
For each book we can see a cover image (if it is present on internet when user adds the book or if user adds after a personal image for the book), the title of the book and the name of the author.

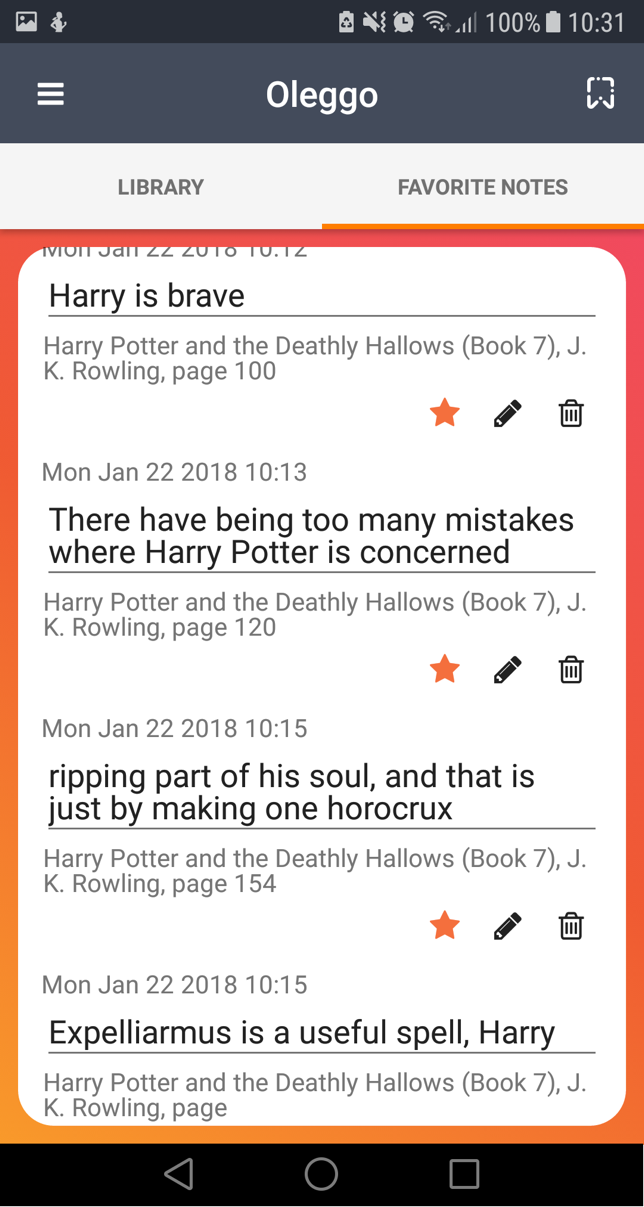
All books are ordered by title.

### *Favorite Notes*

In this page we can see all favorite notes of all books order from the newest to the oldest.   
For each note we have the date when we inserted it, then the title, the author and, optionally, the page of the book which the note is connected.

Below the note we have three buttons:

* Star – to remove, or add again, the note to the favorite. When we remove the note from the favorites note, it doesn’t disappear immediately but only when we recharge the “favorite notes” sub-page, in this way if we press the star by mistake we can press again to add the note to the favorite and not need to go to the book page.
* Pencil – to save the modifications of the note if we want to change something. However the date of insertion doesn’t change.
* Trash bin – this button simply remove the note.



## Book

This is simply the personal page of a book.

It is divided in three tabs and a full screen dialog:

### *Information*

When we select a book in the home page or in the reading book page this is the first sub-pages that appear. It shows general information of the book:

* Title.
* Author.
* How many words and notes are linked to this book.
* Status of the book (we will see it in details below) .
* How many percentage of book we already read of this book.

In this page we can some information about the book:

* Pressing in the central image we can select a new picture for the cover of the book.
* Pressing the left button it appears the update progress screen (we will see it in details below).
* Pressing the central button we can change the status of the button.
* Pressing the right button we can change the status of the button in main active.

A book can have three status:

* Passive – it is the default state when book is added. With this state the book is showed only in the library list in the main page.
* Active – when we put active one book it is showed also in reading books page. We put active the book we are reading.
* Main Active – the main active book is the book which are linked all the notes and words that will be added by us. It is showed in reading books page like the active.
* Obviously we can have more passive or active books but only one main active book.

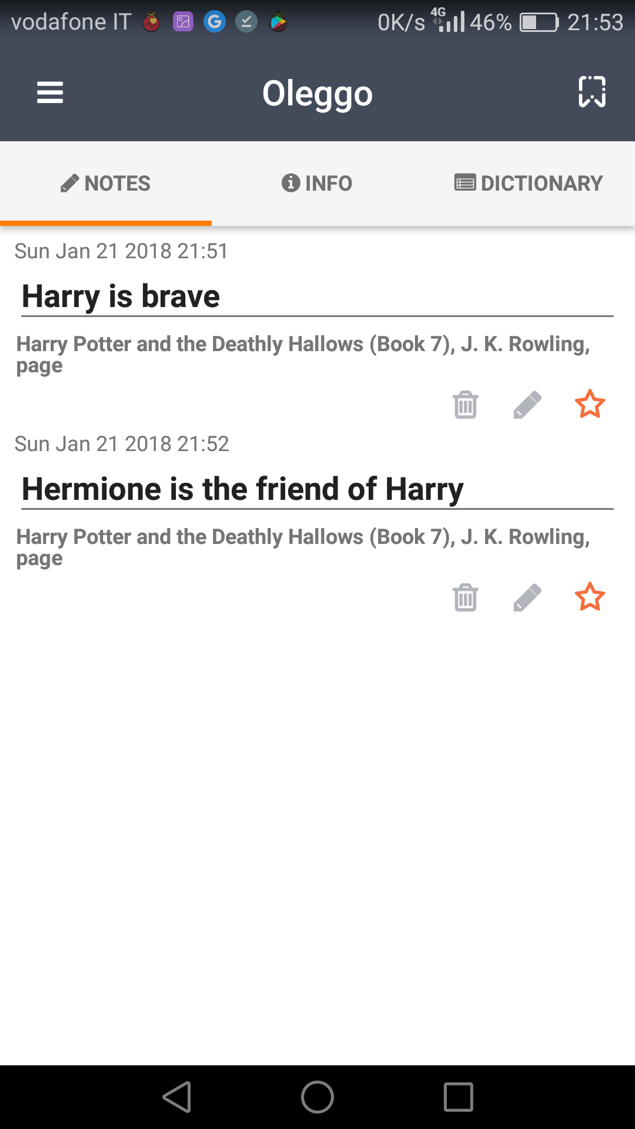


### *Notes*

If we press notes in the tab menu or if we shift the finger from left to the right we arrive in the notes sub-page.

Here we can see all notes linked to this book. Like in the home page here we can modify (and save) and remove any note.

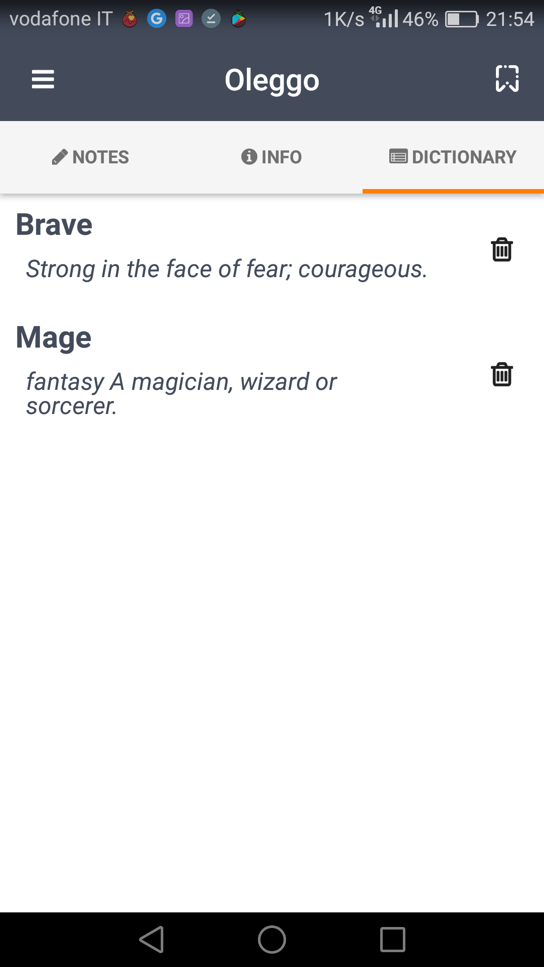
We can also add or remove a note from the favorite. If we add it obviously we will find it in the home page the next time we open it.



### Dictionary

If we press dictionary in the tab menu or if we shift the finger from right to the left we arrive in the notes sub-page.

Here we can see all words linked to this books with their definitions in alphabetic order.  
We can’t search a specific word like in the dictionary page and if we remove a word here we just remove the word linked to this book. It’s mean that if the same word is linked to different book if I remove it from here I will continue to see it in the dictionary page or in the personal book page of all books that are linked with this word but not anymore in this book.



### Update progress screen

In this screen we can update the bookmark of the book inserting the page where we are arrived to read and pressing the update progress button.

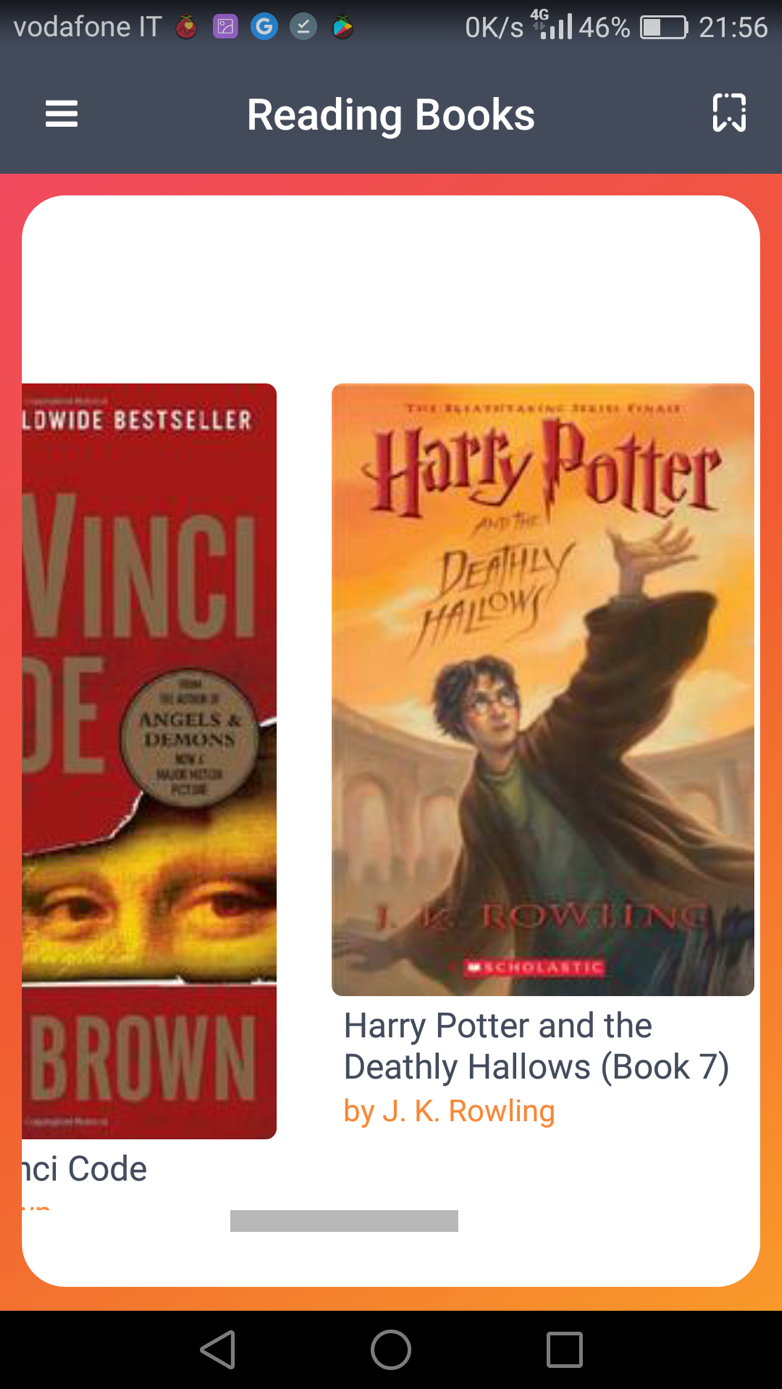
If we finish the book we can directly press “I have finished the book” button.



### Reading Book

This page shows all books that have the “reading” status. It’s useful for the us because here we can see only the book that we are reading or studying in this moment and find quickly the book we are looking for.

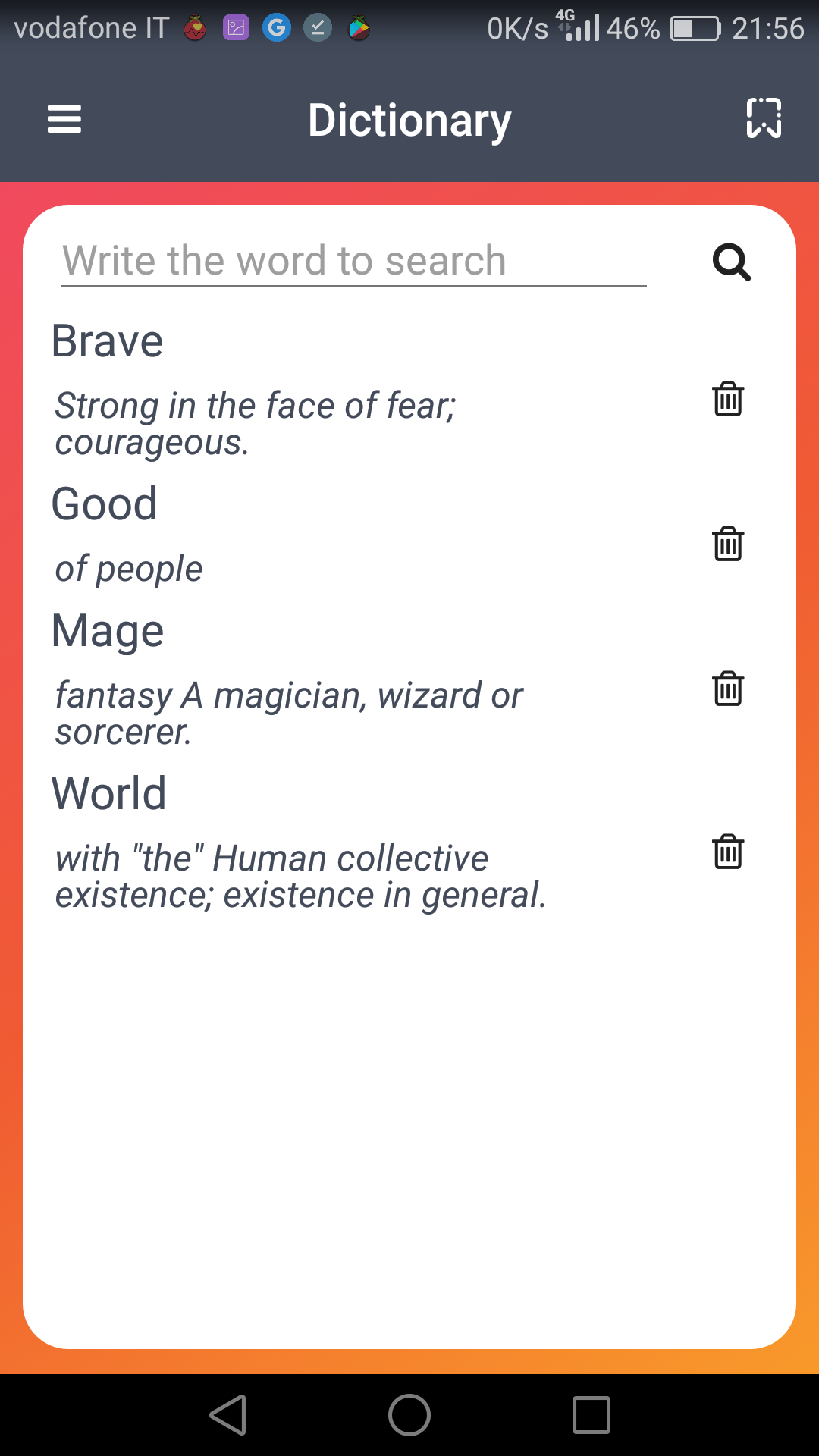
If we press in a book, we go to the personal page of that book.



### Dictionary

This page shows all words which we searched the definitions in alphabetic order.  
If we searched the same words for different books in this page the word appear only one time but in the database it is saved more times, one for each book.  
we can also search a specific word inserting it, or just a part of it, in the text box on top on the page and pressing the lens. The page will show all words that contain the text that user insert in the text box.

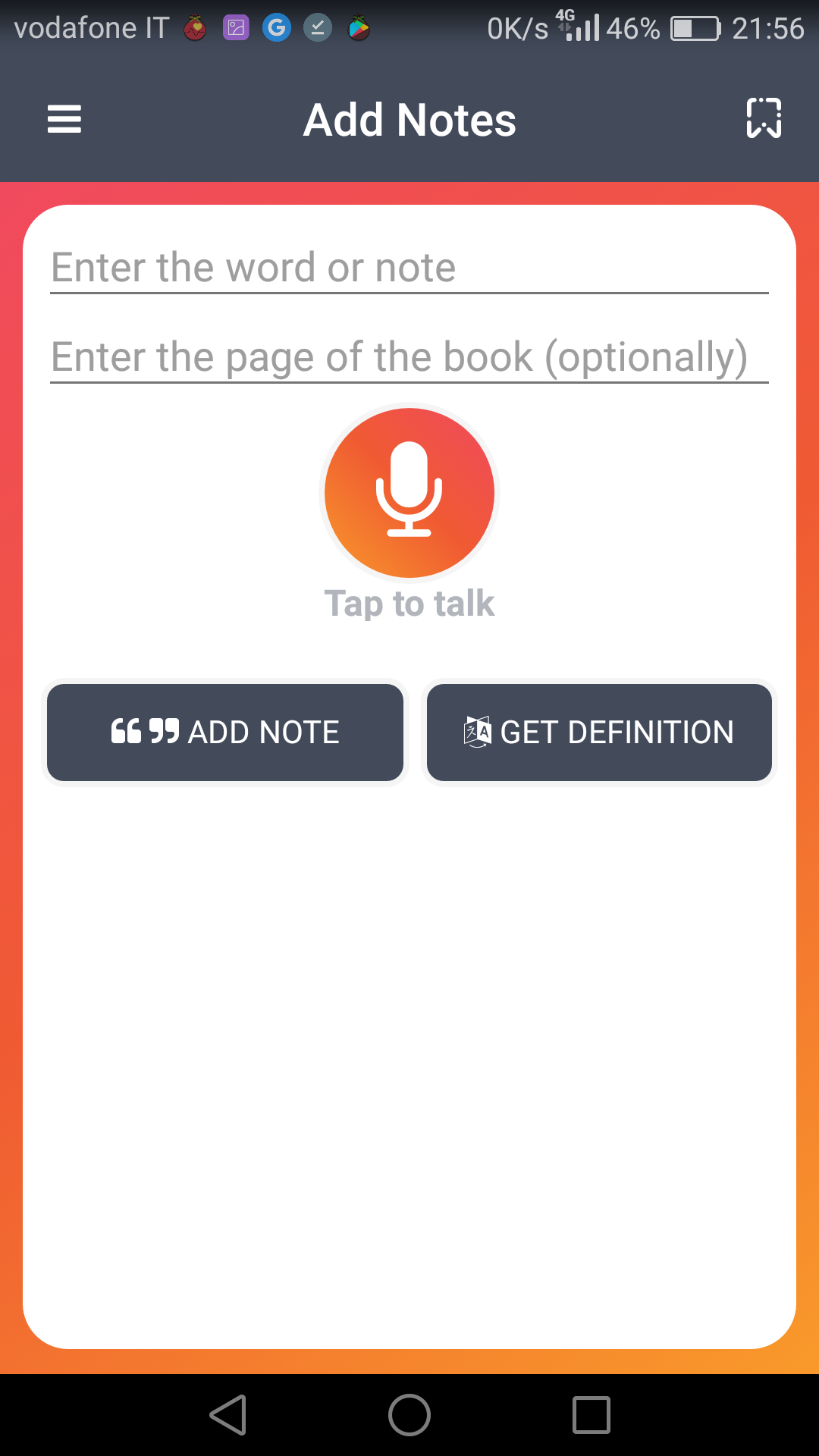
Finally, we can remove a word pressing the trash bin next to the word. If we remove a word here it will be removed all the occurrence of this word present in the database.



## New Entry

First of all, we must know that we can reach quickly this page from any page using the icon on the top right page. In this way we don’t lost time if we must write a note and we have a few time.

In this page we can insert a new note or search the definition about a word that will be added to the dictionary.



* *Note*

We can write the note directly in the text box on top of the page or we can press the microphone and speech. Our voice will be translated in text and printed in the same text box, so we can modify it. We can also insert a page that will be showed together the note, but it is optionally.

Then we must press “add note” and note will be added and connected to the main active book.

* *Word*

Like to the note we can insert the word we want to know the definition writing it or using the speech recognition.

Then we must press “get definition” to look for the word online, in Wiktionary, and add it the dictionary.  
If the application doesn’t found the word in Wiktionary it adds the words to the dictionary without definition.

If we doesn’t set a main active book we can’t add word or note and if we try to add one of then an alert advice us to choose a main active book before.

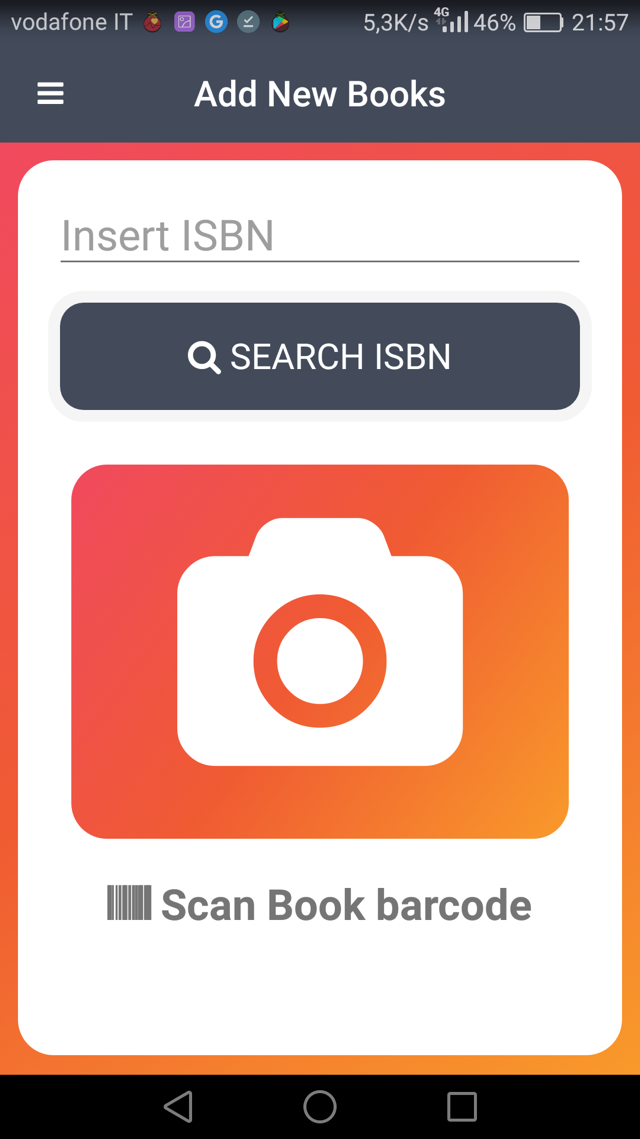
## New Book

In this page we can add a book inserting its ISBN. It can be inserted using the text box or reading the bar-code of the book. To use the bar-code we must press the image of photo camera and point the smartphone on the bar-code of the book checking to center it.

After inserting the ISBN we must press the search ISBN button that look for information about the book in two different online library.

Also the book is found can happen that it isn’t present in the information the author or the number of pages. In these two case application adverts us with a dialog and asking us to enter the name of the author or/and the number of the pages. In any case application asks us to confirm the number of the pages found in the information if it is.

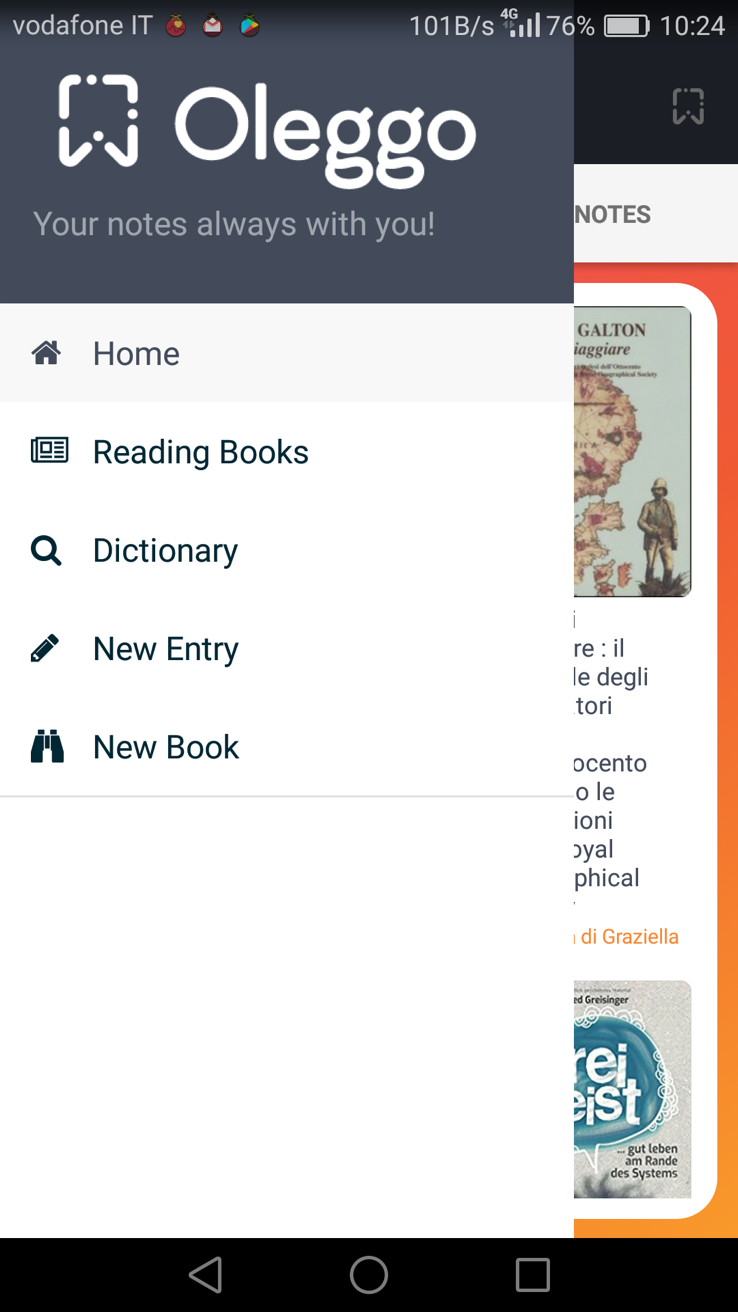
In other case, if the ISBN isn’t present in the online libraries used, application asks us to insert title, author and number of pages.



## Side Drawer menu

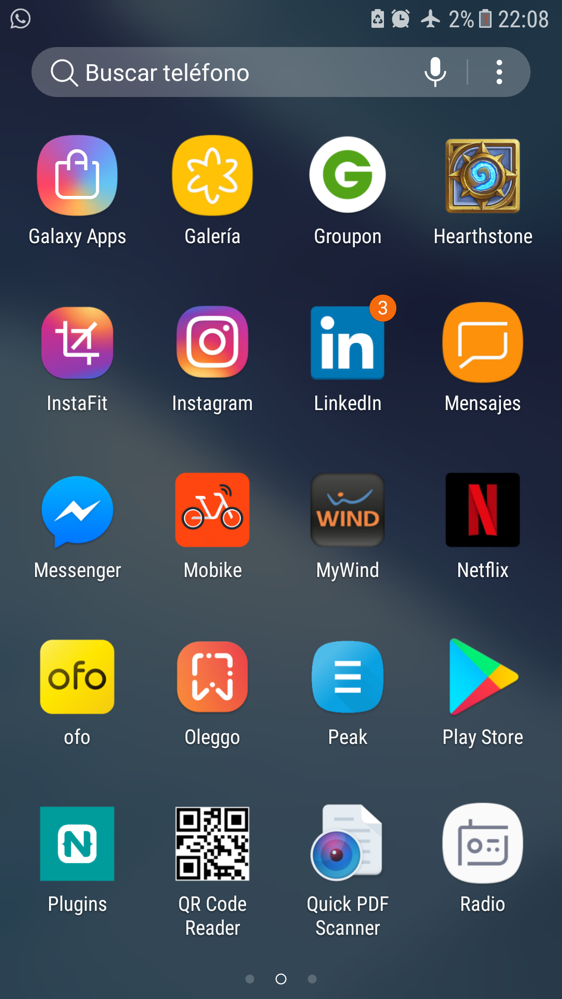
We can access to the drawer menu pressing the icon on top left of the page or shift the finger from left to the right.

Using it we can navigate between the different main pages of the application.



## Complementarty screens

icon view and splash



# Requirements Traceability

## Software System Attributes

### Reliability

The system will be developed with redundancy, periodical backups, and all the functions must pass several test to be implemented in the program, to avoid algorithm problems as possible, for that reason the information is save in the cache of the logic tier and often saved also in the database.

### Availability

My taxi service will be accessible anytime. All the service will be hosted in the cache of the logic tier primary, but is also will have the information on the database as a backup plan in case the servers goes down, the database will recover the information to start working at any time they would be need it, the system when starts send a message to the users and restart the services as soon as possible.

### Security

The app is password-protected, there is a method to authenticate the user identity and specially the driver’s identity. The system will guarantee the safety of the user's data according with the agreement accepted at the registration.

### Maintainability

The system will be very maintainable thanks to a modular structure. The application will be documented in a way that will truly help the future developers, explaining how the application works and how it has been developed, in addition, the code will be periodically examined and improved, with a proper documentation.

### Usability

The application is meant to be as easy and intuitive to use as possible. The drivers have capacitation for improve their performance with the app. All the users can provide feedbacks to improve the app.

### Portability

The software for the system will be developed using the Java language and related dependent technologies because Java is designed to have as few implementation dependencies as possible. Meaning that code that runs on one platform does not need to be changed to run on another.

# Software used

The following software were used to redact and do the content of this document:

* StarUML: to create Use Cases Diagrams, Sequence Diagrams and Class Diagrams.
* Microsoft Word: this editor was used to redact the document.
* Mockups.com: to create interface mockups.

# References

1. Design and Implementation of Mobile Applications course material [Online], <http://home.deib.polimi.it/baresi/dima.htm>
2. Official website of NativeScript with useful guides and plugins, <https://docs.nativescript.org/>
3. Repositories when we took different pieces of code for graphics and template components, <https://github.com/EddyVerbruggen> - <https://github.com/TobiasHennig>
4. API used for search the books using ISBN, <https://openlibrary.org/developers/api> - <http://xisbn.worldcat.org/xisbnadmin/doc/api.htm>
5. Plugin for search the meaning of the word, <https://www.npmjs.com/package/word-definition>
6. Icons, <http://fontawesome.io/>