UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escola Tècnica Superior d'Enginyeria Informàtica

etsinf

ANGEL KANCHEV UNIVERSITY OF RUSE

Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

Faculty of Electrical Engineering, Electronics and Automation

"Angel Kanchev" University of Ruse

Web application for monitoring automated flowerpots

Diploma Project

Degree in Computer Engineering

Autor: Viktor Toshkov Yordanov

Tutor: Poza Luján, José Luis
Posadas Yagüe, Juan Luis

2016 / 2017

Abstract

Nowadays, caring for your plants (that grow in gardens or flowerpots) has become more difficult and this can lead to slowing the growing process or even death of the plant. In order to help for the prevention of this problem so far has been created several devices which offer full control and automatization and can take care of your plants for you. Those systems however need to be managed easily as well as remotely.

The usefulness of the mobile phones is well proven over the years and is one of the best solutions for the problem with the management of the automated systems. Developing a mobile application and making it user-friendly is a long process, but surely provides a great solution to the problem. The main purpose of the current work was to create a prototype of a mobile application for remote monitoring and control of those automated systems.

In order to carry out this project, a research of similar systems have been done and from them the functionalities that are desired for the application have been extracted. All of them were described using Use-Case diagrams and tables. For the design of the application it was chosen to use the Three-Layer architecture, because it is a part of another bigger project done by other students and this way it would provide bigger flexibility, but later it was developed using the MVC architecture in a local environment. The main goal of this project was to offer a way to visualize the data coming from the chombos, also a way to move it. Another objective that was achieved is to make the design of the application user-friendly and lightweight as well as responsive to be able to run on different devices.

Most of the functionalities that were described are only visually implemented, but not functional, while only some of them were fully implemented to make the application feel alive.

Resumen

Actualmente, el cuidado de las plantas domésticas (que crecen en jardines o en macetas) es complicado para mucha gente, lo cual puede ocasionar un crecimiento más lento de las plantas o incluso que mueran. Para ayudar a prevenir este problema, se han creado dispositivos que ofrecen un control total y automático para el cuidado de las plantas. Sin embargo, estos sistemas necesitan ser gestionados fácilmente y de modo remoto.

En los últimos años se ha demostrado la versatilidad de los teléfonos móviles, de modo que están siendo una de las mejores soluciones para el problema de manejar sistemas automáticos. El desarrollo de una aplicación web que sea fácil de usar es un proceso complejo, pero sin duda proporciona una solución adecuada al problema planteado. El propósito principal del trabajo actual fue crear un prototipo de una aplicación móvil para el monitoreo y control remoto de esos sistemas automatizados.

Para llevar a cabo este proyecto, se ha realizado una investigación de sistemas similares y de ellos se han extraído las funcionalidades que se desean para la aplicación. Todos ellos se describieron utilizando diagramas de casos de uso y tablas. Para el diseño de la aplicación se optó por utilizar la arquitectura Tres Capas, ya que es parte de otro proyecto más grande realizado por otros estudiantes y de esta manera se obtendría una mayor flexibilidad, pero más tarde se desarrolló utilizando la arquitectura MVC en un entorno local. El objetivo principal de este proyecto era ofrecer una forma de visualizar los datos procedentes de los chombos, también una forma de moverlo. Otro objetivo que se logró es hacer que el diseño de la aplicación sea fácil de usar y ligero, además de responder a la posibilidad de ejecutar en diferentes dispositivos.

La mayoría de las funcionalidades que se describieron sólo se implementan visualmente, pero no son funcionales, mientras que sólo algunas de ellas se implementaron completamente para que la aplicación se sintiera viva.

Table Of Content

Illustrations

Tables

1.Introduction

1.1 Environment and Motivation

The current technologies continuously grow and more electronics are starting to be used in many aspects of our everyday activities. As the work grows, the time to care for our garden plants is lesser and the need of automation in this regard grows. For this reason, are developed different devices with sensors, which measure different environmental variables (humidity, light level, pollution, etc.) and actuators, which perform actins like controlling the watering or light level, for example. This opens another need for easier management of those devices.
The solution of the problem above is really easy by using a mobile application, because nowadays almost everyone has a smartphone. That is why this project is for making a mobile application to manage such automated flowerpot systems called “chombos”. Other project related to this one is for making a website to commerce those chombos.

1.2 Project Objectives

Firstly, the project consists of creating a prototype of a Mobile application for management of chombos, which includes the following functionalities: monitoring the data coming from the sensors, controlling the actuators and controlling the movement of the flowerpot, and also management of friends. The option to move the chombo is one of its unique features, which is still uncommon for these types of devices. This prototype must behave as a real mobile application, using fictional data and local database.

Secondly, because we live in an era where more and more information is shared between people, the chombos will also follow this trend. For this reason, the mobile application must have a way to provide the information to other users, but not all the information must be accessible by every user. The owner of the chombo must have a way to set the permissions of the others.

Moreover, the application should be designed like it is using the three-layer architecture, which means that the tree layers must be defined and replicated as close as possible.

Lastly, the mobile application is important to be simple and easy to use, and as lightweight as possible to be used by anyone without problems. This goes not only for the applications, but for all the software in general. Also, it should be responsive to be used by different mobile devices.

1.3 Structure of the document

This document is composed of 6 sections, which are described below:

The first section introduces the content of the project, its environment in which it is carried out and the objectives that must be accomplished.

In the second section, a study of the market is performed in order to research the similar mobile applications and later to select the functionalities that are going to be implemented in this project.

The third section defines the overall description and all the aspects of the application in technical and more detailed way.

The fourth section presents the design of the system, the distribution of all data and principal description of the interaction chosen.

The fifth section details how the project is implemented.

The sixth section contains the conclusions of the project and also the difficulties that have been encountered as well as their solutions, and some ideas for future implementations.

2. Environment

2.1 Introduction

Nowadays, caring for your plants (that grow in gardens or flowerpots) has become more difficult and this can lead to slowing the growing process or even death of the plant. In order to help for the prevention of this problem so far has been created several devices which offer full control and automatization and can take care of your plants for you. Those systems however need to be managed and the best way is by using mobile applications.
In this section, a marketing research of the similar systems has been performed and all the details of the applications that have been found are described. An analysis of their characteristics is also included and from that the functionalities for the project are selected according to the project's objectives and trends. Also, the technologies that are going to be used for the realization of the project are included here.

2.2 System Similarities

According to the committed research there are several systems that exist with functionalities - similar to those that are desired for the project and also help the application stand out among the others. Those mobile applications are listed below.

2.2.1 Edyn

Edyn is a smart garden system that monitors and tracks environmental conditions. Inserted in the soil, the Edyn Garden Sensor gathers and analyzes data about changing weather and soil conditions and sends this data via Wi-Fi.

The Edyn App displays this data as a real-time snapshot of your garden, and pushes alerts and suggestions to maximize plant health. It also provides a historical data charts to better understand and track the environmental changes like how much light or what temperatures has your plant received in the past. You can track the growth of your plants and set notifications to remind you for important events. Lastly, it has a rich database with information on more than 5000 plants.

Illustration 1: Interface of the EDYN application

2.2.2 MiniGrow

MiniGrow is a unique small grow box, optimized to grow and flourish one single plant in a perfect environment. It uses a hypermodern light- and computer technology. Monitor the conditions of your favorite plant 24/7 on your mobile phone or tablet with the MiniGrow App.

The App allows you to monitor not only the environmental data like light and humidity, but also the status of some of the devices like LED lights and fans. It also provides chronological charts with the environmental data and lets you set notifications.

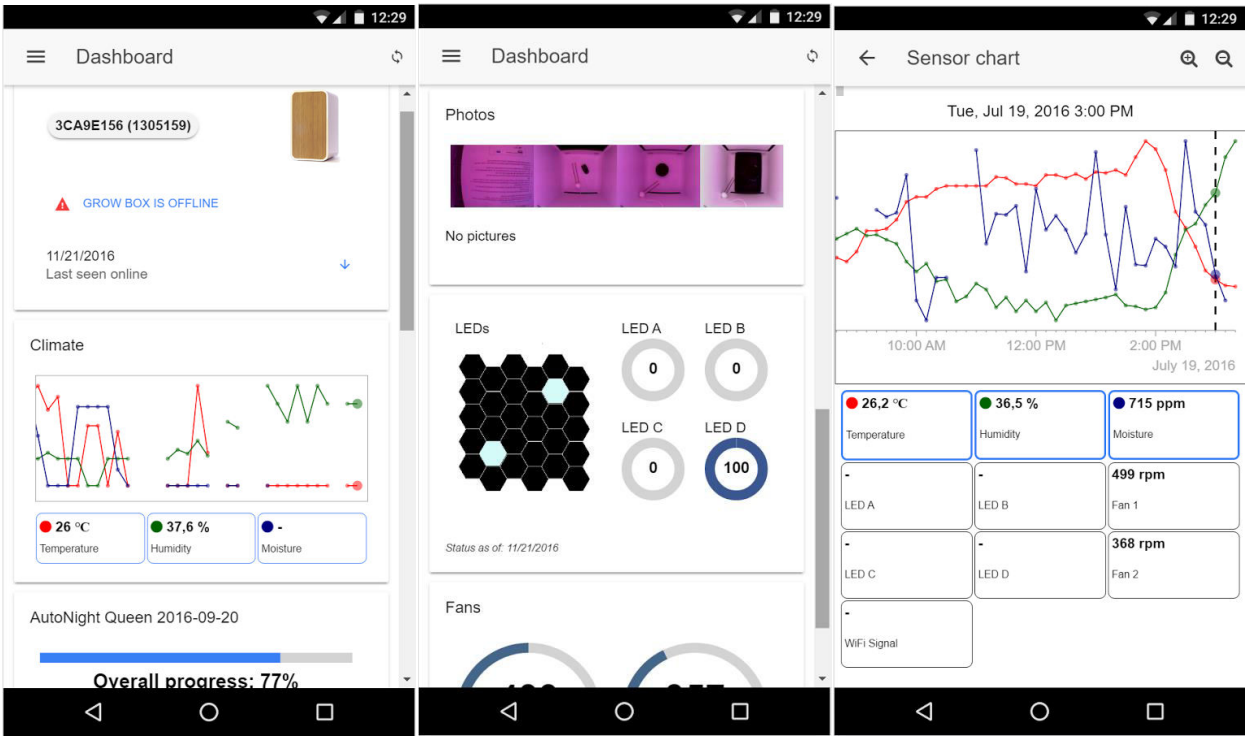


Illustration 2: Interface of the MiniGrow application

2.2.3 GreenIQ

This is the companion app to the GreenIQ Smart Garden Hub. The Hub controls irrigation scheduling based on current and forecasted weather, and saves up to 50% on your outdoor water consumption. This app allows you to control your GreenIQ Smart Garden Hub from anywhere, at any time, and connect it wirelessly to a wide variety of smart devices and sensors such as Flower Power and Koubachi, Netatmo and more.

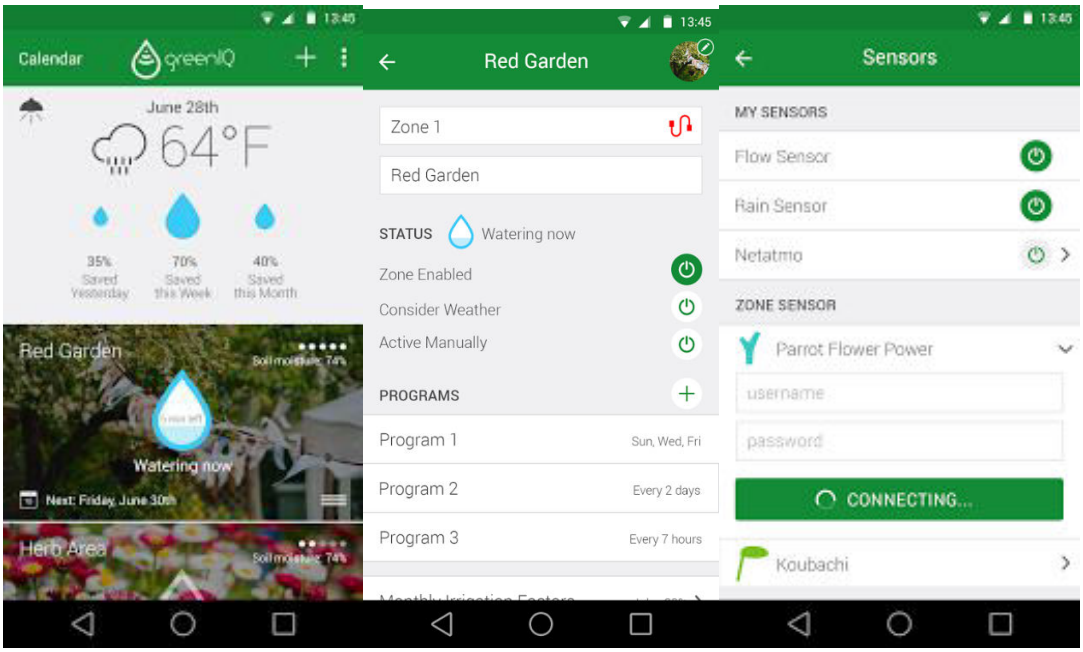


Illustration 3: Interface of the GreenIQ application

2.2.4 Parrot Flower Power

Parrot Pot is a connected pot that helps you keep your plants healthy. It incorporates an intelligent watering system and four sensors that continuously analyze your plant's needs. The data collected by your Parrot connected object (Pot or Flower Power) are sent to your smartphone or tablet thanks to Bluetooth Low Energy wireless technology (Bluetooth v4.0). With the Parrot Flower Power application, you can monitor your plants' development every day and control the watering. Additionally, you can access a vast database packed with information on your plant's maintenance and characteristics.

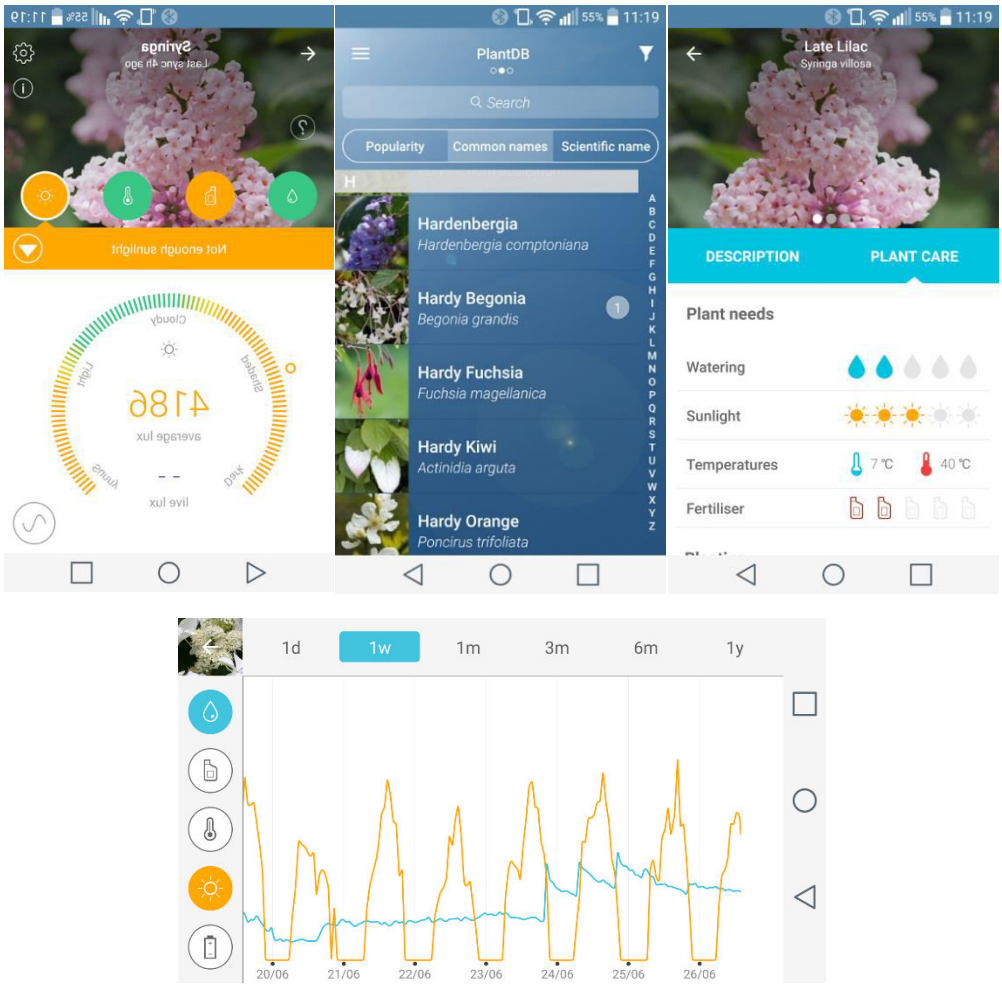


Illustration 4: Interface of the Parrot Flower Power application

2.2.5 Koubachi

Koubachi helps you care for your precious plants. Koubachi notifies you when to water your plant, give fertilizer, mist the leaves or it's too hot / cold, or too sunny / shady. Your plants information is stored on our servers where we run the Koubachi Plant Care Engine. The system takes into account the plants' species, the current season, your geographic location, and most importantly the water cycle duration from the calibration.

With the App, you can: calibrate your plants for even more accuracy; Get push notification and choose when to be notified; View plant library containing most common species; Receive over-the-air library updates; Read adaptive care plans from our users; Send feedback if plant species are missing and more.

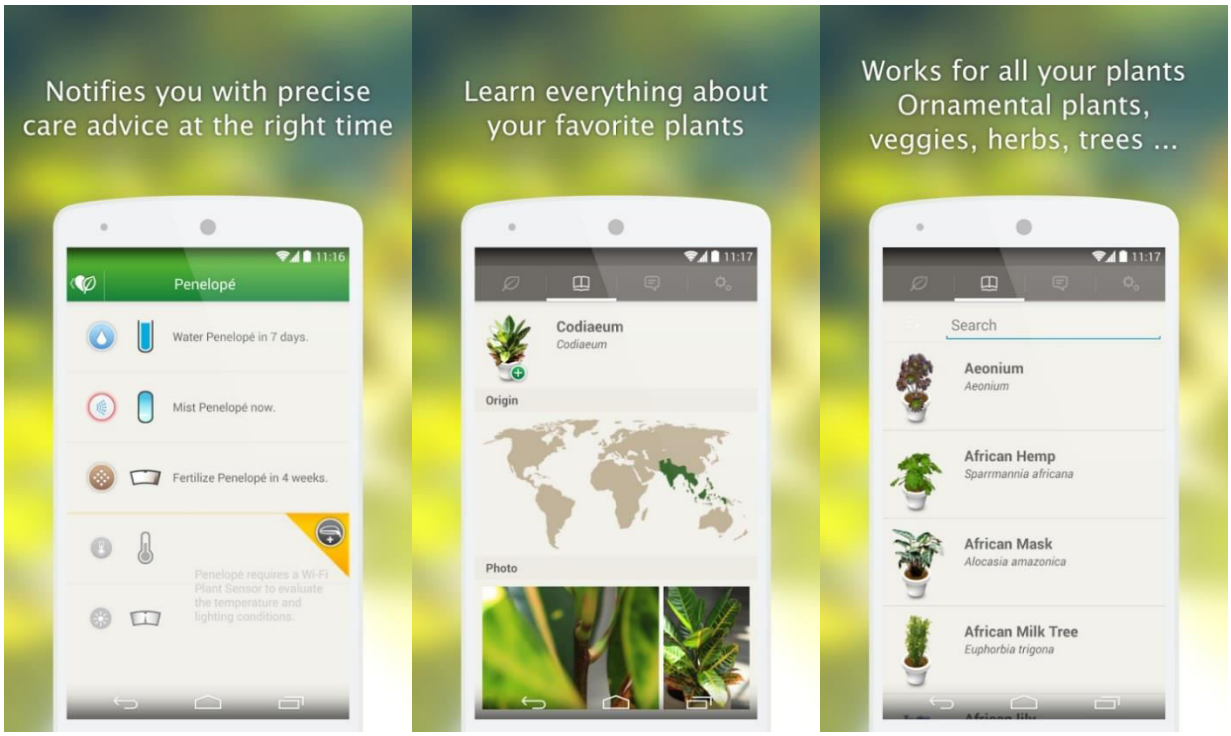


Illustration 5: Interface of the Kubachi application

2.2.6 GROVE

Grove is a mobile app for IOS devices which is designed to support the Grove growing system, the Grove App requires the Grove Garden to access most features: Customize light color spectrum and intensity, set water flow to optimize for plant growth, set fans to Low, Medium or High to strengthen your plants and more. It also provides all the information gathered from the sensors and guides on how to grow your plants, and you can also receive notifications/alarms when your plant(s) need anything.

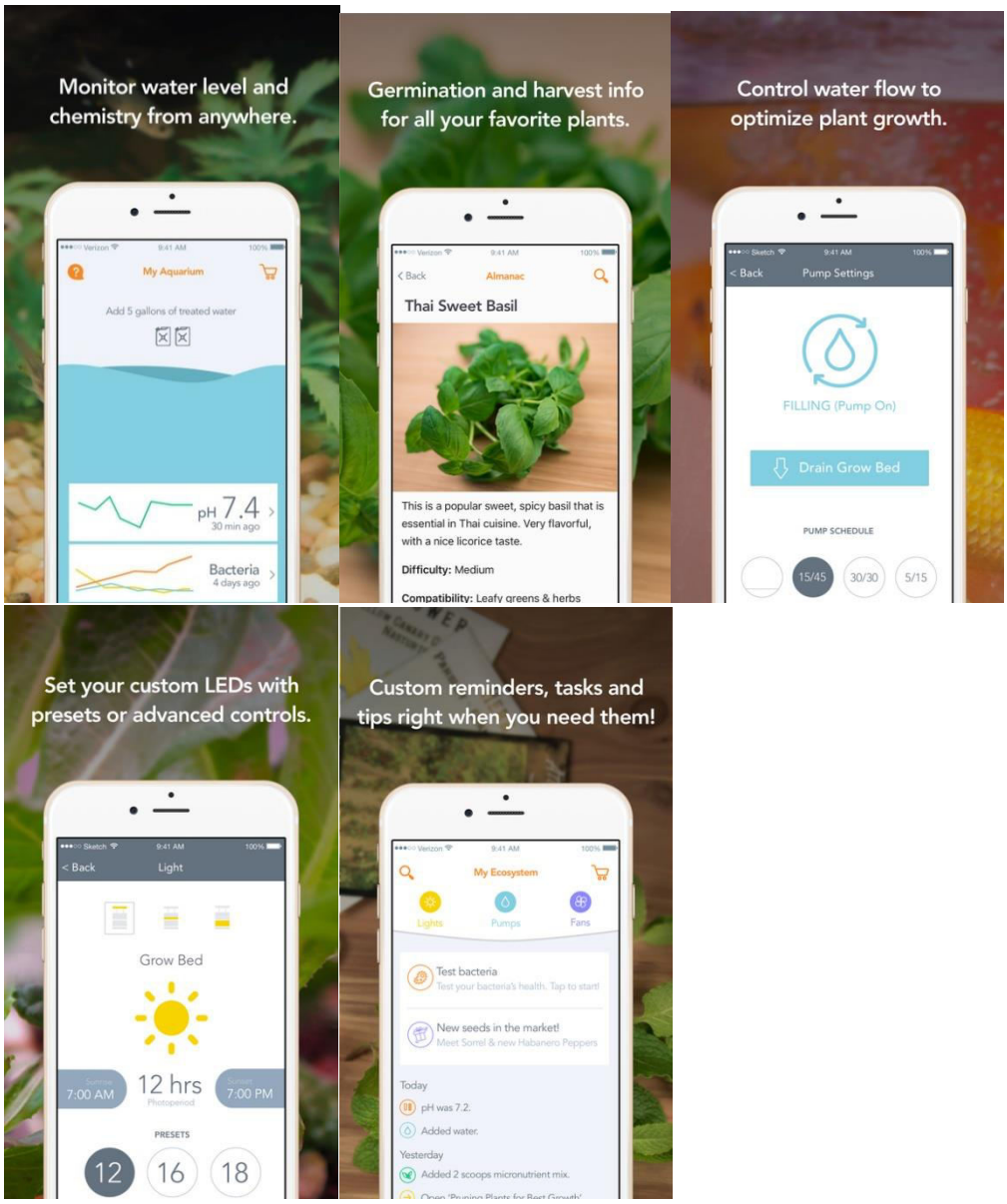


Illustration 6: Interface of the GROVE application

2.2.7 My PlantLink

By combining soil moisture sensors with an internet-connected wireless infrastructure, PlantLink generates automatic watering schedules using plant-specific algorithms. My PlantLink is a mobile app for ios devices that support the PlantLink devices that lets you monitor the soil moisture history, view watering schedule, receive notifications, manage your alerts (when you are alerted and for what). It can support up to 64 devices.

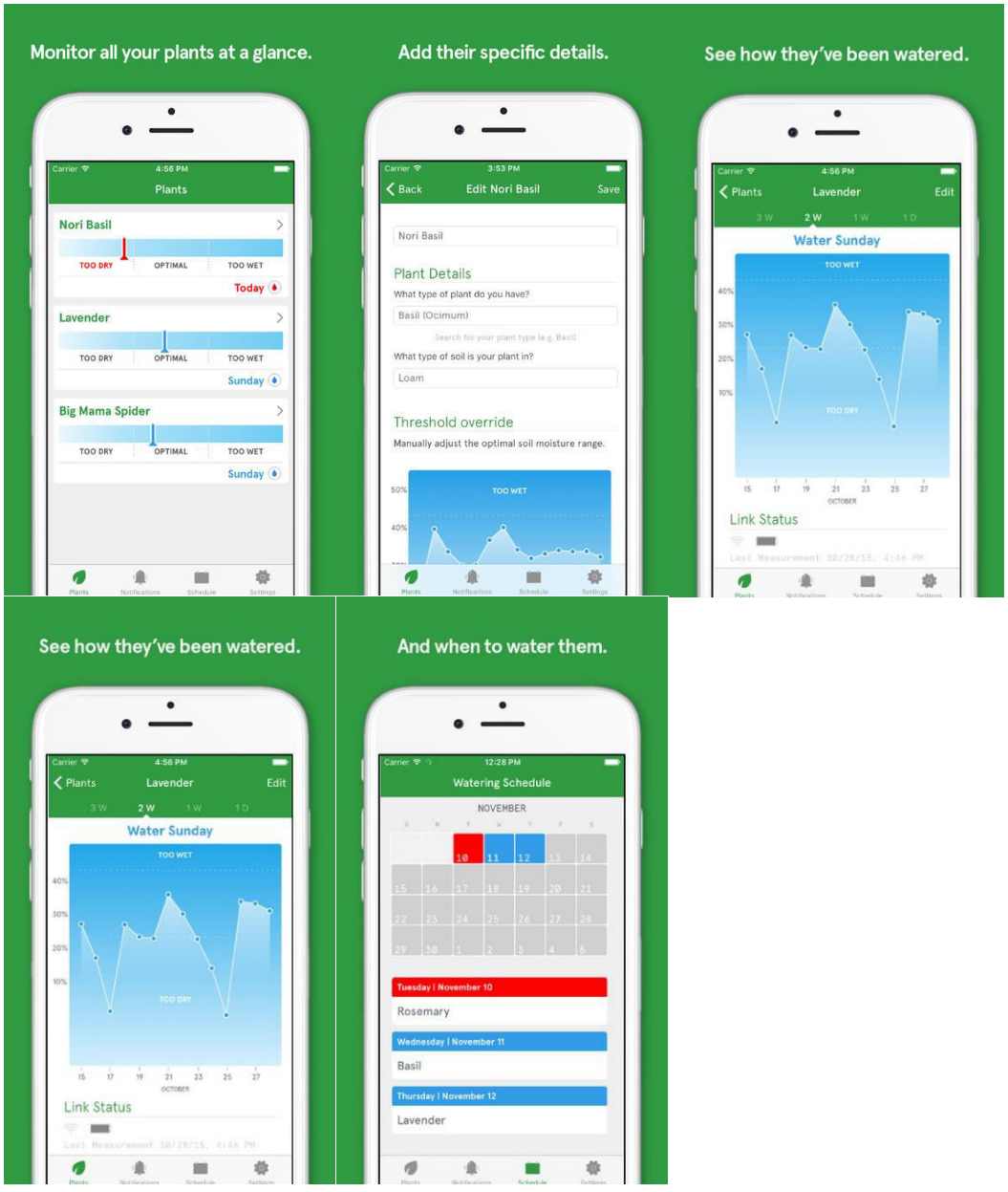


Illustration 7: Interface of the My PlantLink application

2.2.8 Green House

Android application which will closely monitor and control the micro climatic parameters of a greenhouse on a regular basis round the clock for cultivation of crops or specific plant species which could maximize their production over the whole crop growth season and to eliminate the difficulties involved in the system by reducing human intervention to the best possible extent. The system also employs a display for continuously alerting the user about the condition inside the greenhouse.

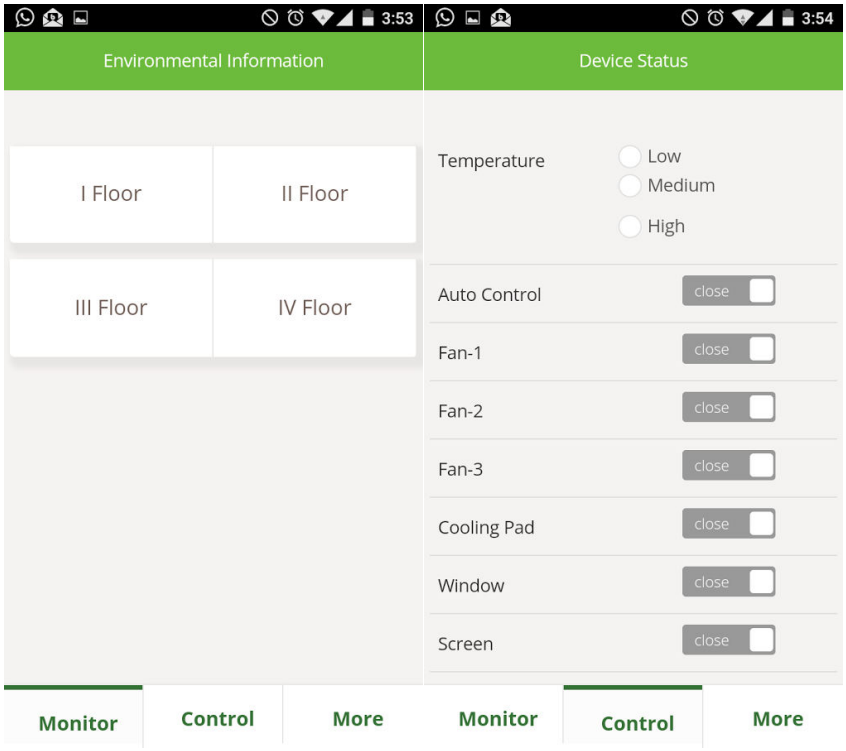


Illustration 8: Interface of the Green House application

2.2.9 Gardening Manager

Gardening Manager is the social garden diary usefull for an incredible edible initiative or your own garden. It allows you to make a “to do” list, take photo and archive them in your plant diary, keep weather history and moon phase, supports multiple gardens, plants recognition from a picture.

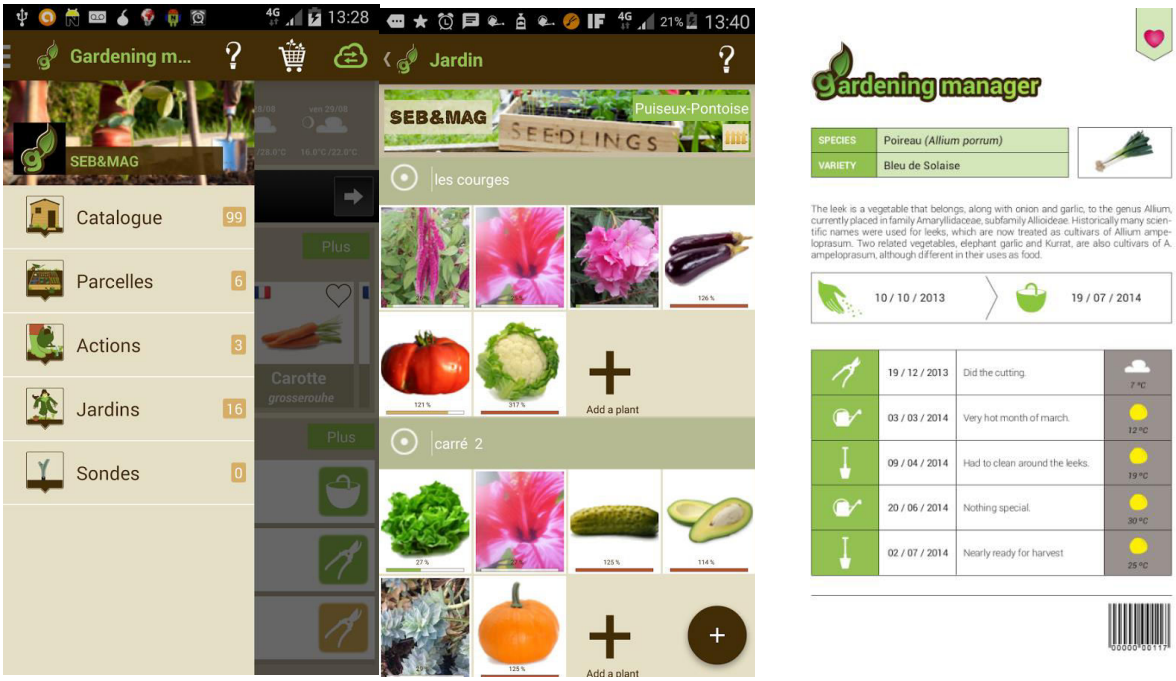


Illustration 9: Interface of the Gardening Manager application

It also has an alternative version “Garden Manager: Plant Alarm” that allows you to set alarms to remind you to take care of your plants and also some special features like:

- Sharing - You can share your logs to your friends via facebook, twitter and so on, or ask them why your plants are withering.
- Finding nearby florists - You can find florists near you.

2.2.10 GRO. Real-Time Gardening

GRO is a helper app for gardening, that can provide you with step-by-step guide on how to grow your plants. First, you pick your preferences on the type of gardening you want to do. Then it pulls in info about your local conditions – location, temperature and season to give you the best ideas for plants and projects that you can successfully grow. Also, you can set alarms to remind you when your plants need anything.

2.3 Analysis

In this subsection, there is a comparison of various features of the researched applications which will help for deciding which functionalities will be included in the project. Below is the comparative table:

Real Time – Allows you to manage your plants in real-time.
Monit. – Shows data gathered from sensors.
Control – Allows you to control some aspects of the devices like watering, light, air.
DB w/ species – Has access to a database with information for a variety of plants.
Historical Data – Keeps and displays historical data gathered by sensors. Environmental and Technical.
Data Charts – Displays data in form of charts.
Notify – Sends notifications and/or alarms to notify you for some events.
SBS Tuts – Have Step-by-Step tutorials on grow your plants.
GEO info – Displays data and/or helps you make a decision based on your location.
OS – The platforms of the application.
Sharing – Allows you to share information about your plants with other people.

Apps	Real Time	Monit.	Control	DB w/ species	Historical Data	Data Charts	Notify	SBS Tuts.	GEO info.	OS	Sharing
Edyn	Yes	Yes	No	Yes	Yes	Yes	Yes	No	No	Android iOS	No
MiniGrow	No	Yes	Yes	No	Yes	Yes	Yes	No	No	Android	No
GreenIQ	Yes	Yes	Partially	No	No	No	Yes	No	Yes	Android iOS	No
Parrot FP	No	Yes	Partially	Yes	Yes	No	Yes	Yes	No	Android iOS	No
Kubachi	No	Yes	No	Yes	No	No	Yes	No	Yes	Android iOS	Yes
GROVE	Yes	Yes	Partially	No	Yes	Yes	Yes	No	No	iOS	No
My PlantLink	No	Yes	No	No	Yes	Yes	Yes	No	No	iOS	No
Gard. Manag.	No	No	No	Yes	Yes	No	Yes	No	Yes	Android	Yes
GRO	No	No	No	Yes	No	No	Yes	Yes	Yes	Android iOS	No
Green House	Yes	Yes	Yes	No	No	No	Yes	No	No	Android	No

Table 1: Features Comparison Table

2.4 Synthesis

Completing the analysis of the various existing applications, we can now proceed to summarize and extract the characteristics that are desired to be included in this project. They are ordered by priority below, starting from the highest to the lowest priority.

1. **Real Time** – By "real time" it is meant that the application will receive information about the pot(s) every 5 seconds (for monitoring) and will be able to queue up commands for it(them) to execute (for management)
2. **Monitoring** – The application will be able to show the Real-Time data from the pot(s). For every metric (like humidity, light…) there will be a separate view.
3. **Control** – The application will be able to control the pot(s) by sending various commands for the watering, light and even for moving the pot(s).
4. **Historical data** – It will be able to show summarized data for the past few days up to a whole month.
5. **Data charts** – The historical data can be shown in the form of data charts which will help to better understand the changes for all the metrics. In the chart can be shown information only for one metric or for multiple at once.
6. **Notify** – It will have the ability to send you notifications to notify you if something important has or it is going to happen. For example, when the batteries of your pot are running low.

The following functionalities are optional, because they are not essential for our application, but can provide some nice features to it:

1. **SBS Tuts** – It can include step-by-step instructions on how to use the pot, application and/or even on how to grow your plants properly.
2. **Sharing** – You can share information for your plants to social networks or to other users of this application.
3. **DB with species** – It may include a vast database with information about many species that you can grow in our pot, which include name, description and how to take care of the plant.
4. **GEO location** – This will allow you to see the location of the chombos on the map public.

The next step is to decide on which platform the project will be developed, and Table 1 gives an idea of what may be the most appropriate option. Although some of the applications are available for more than one platform, in most of the cases it uses the Android platform. For this reason, also for reasons of availability of resources, this platform has been chosen.
For the development of Android applications it is quite common to use the SDK and in addition there are different tools to program. In this case, you have opted for Android Studio, an already known environment that provides good features and is very useful for certain aspects of programming.

2.5. Conclusion

This section has reviewed the applications for management of flowerpot/gardening systems that exist in the market. After the comparative analysis of those applications, the characteristics that would be suitable for the system have been extracted. In general, we want the application to be able to display all the information in a clear and orderly way, besides allowing us to control some aspects such as irrigation and movement, which are the most important objective of the project. The other optional functionalities are good to be added as well, but that will most likely lead to slowing the speed and making the application bigger, which is against one of the project's objectives.
Finally, for the platform it is chosen the Android, along with the Android Studio program, but only to make the integration of the application easier. The prototype of the application will be fully made like a normal responsive website and by using the "webview" feature of the Android Studio, it will be integrated and behave like a normal mobile application. This way it will be really lightweight on any mobile phone, but may cause slower loading of some pages, which can be ignored.

3. Requirement Specification

3.1 Introduction

In this section, the project is described in more details. The main functionalities are described by means of diagrams and tables in order to get a clearer view of how the application will work. One project must be defined as detailed as possible, so for that reason this part of the document is one of the most important. The use-case diagrams are ordered in depth, not in priority, which means that every next diagram is a child node from a previous with very few exceptions. The functionality tables below the diagrams help define the priority of these functionalities, so that it makes it easier to determine which of them must be implemented first and which can wait until later.
One important thing to note is that from the perspective of the application, there is only one type of user and that is the registered user, but if looked from the perspective of one chombo, there are several types of users. This is shown and explain in detail bellow.

3.2 Use Cases

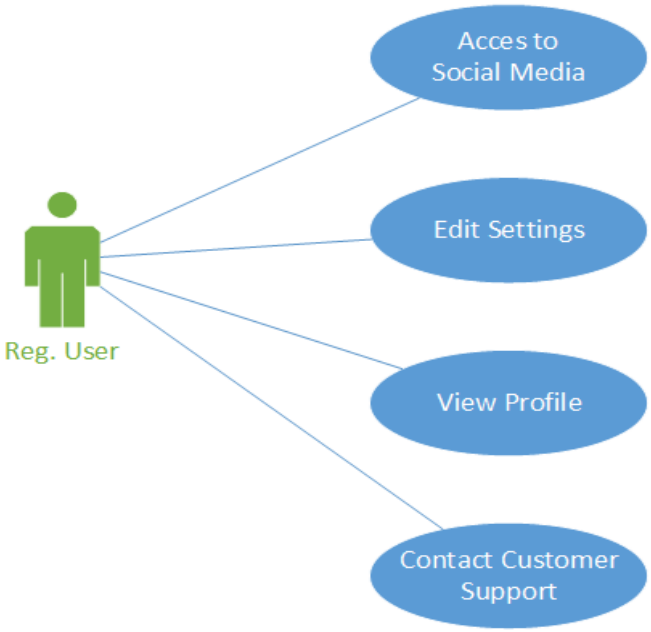


Illustration 10: UC - Main

UC_00: Main. This diagram represents the type of user in perspective of the mobile application. In order to access it he must have an account. His main actions are accessing the social media pages of the chombo community, edit the application's settings, contact customer support and of course – access his profile page, which opens more options. This is described in other diagram below.

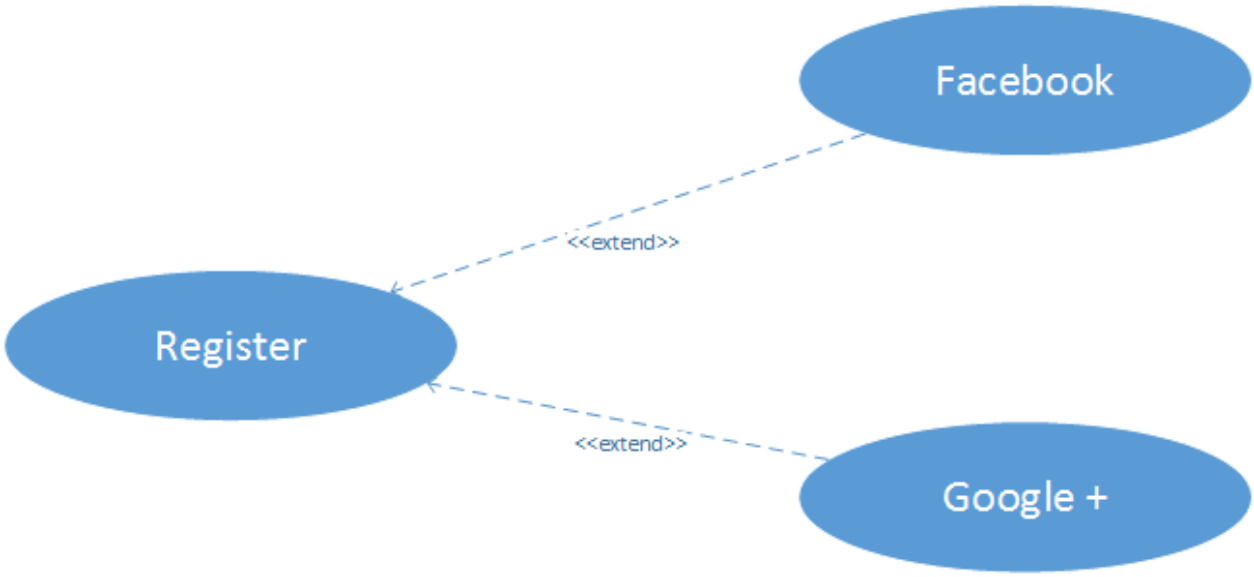


Illustration 11: UC - Register

UC_01: Register. From the main diagram, it is clear that in order to use the application, the user must have an account. He can make one from the mobile application by filling the registration form or use one of the options bellow, which are to create an account using their Facebook or Google+ account.

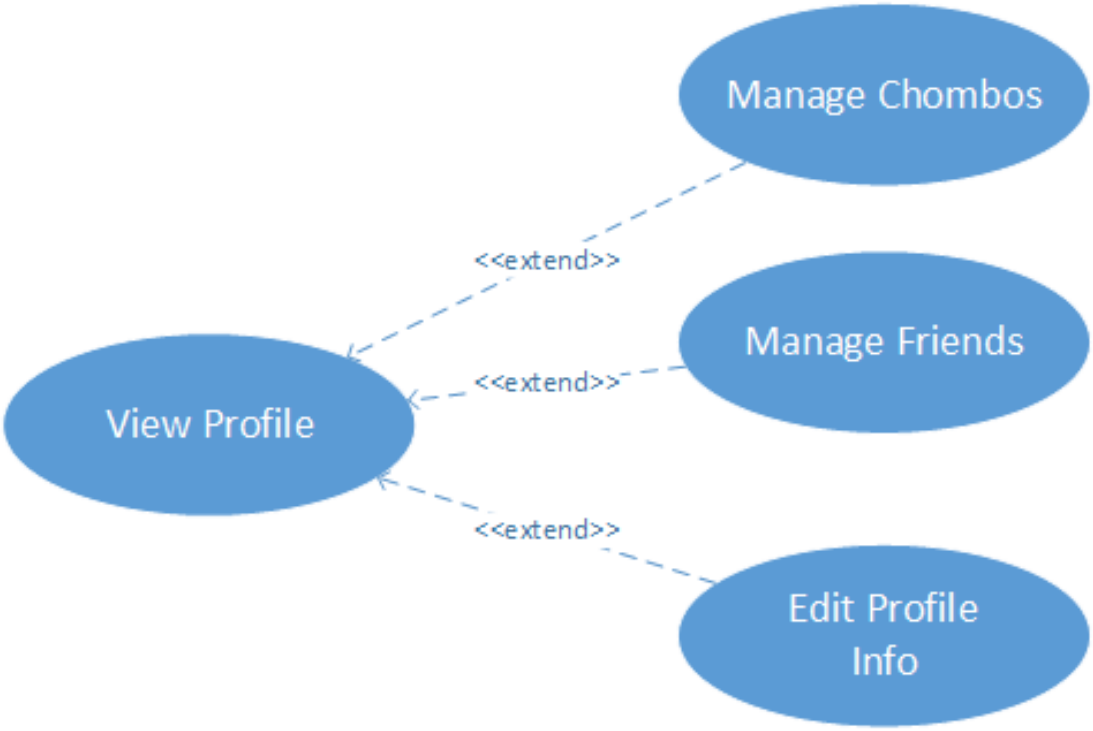


Illustration 12: UC – View Profile

UC_02: View Profile. This diagram is referring to all the options available when the user visits his profile page. This is the main page of the application. Here he can check his profile info and edit it anytime he wants, he can manage his friends and it's possible to manage the chombos that the user has or have access to, but don't belong to him. The Manage Chombos and Manage Friends nodes are described in the diagrams below.

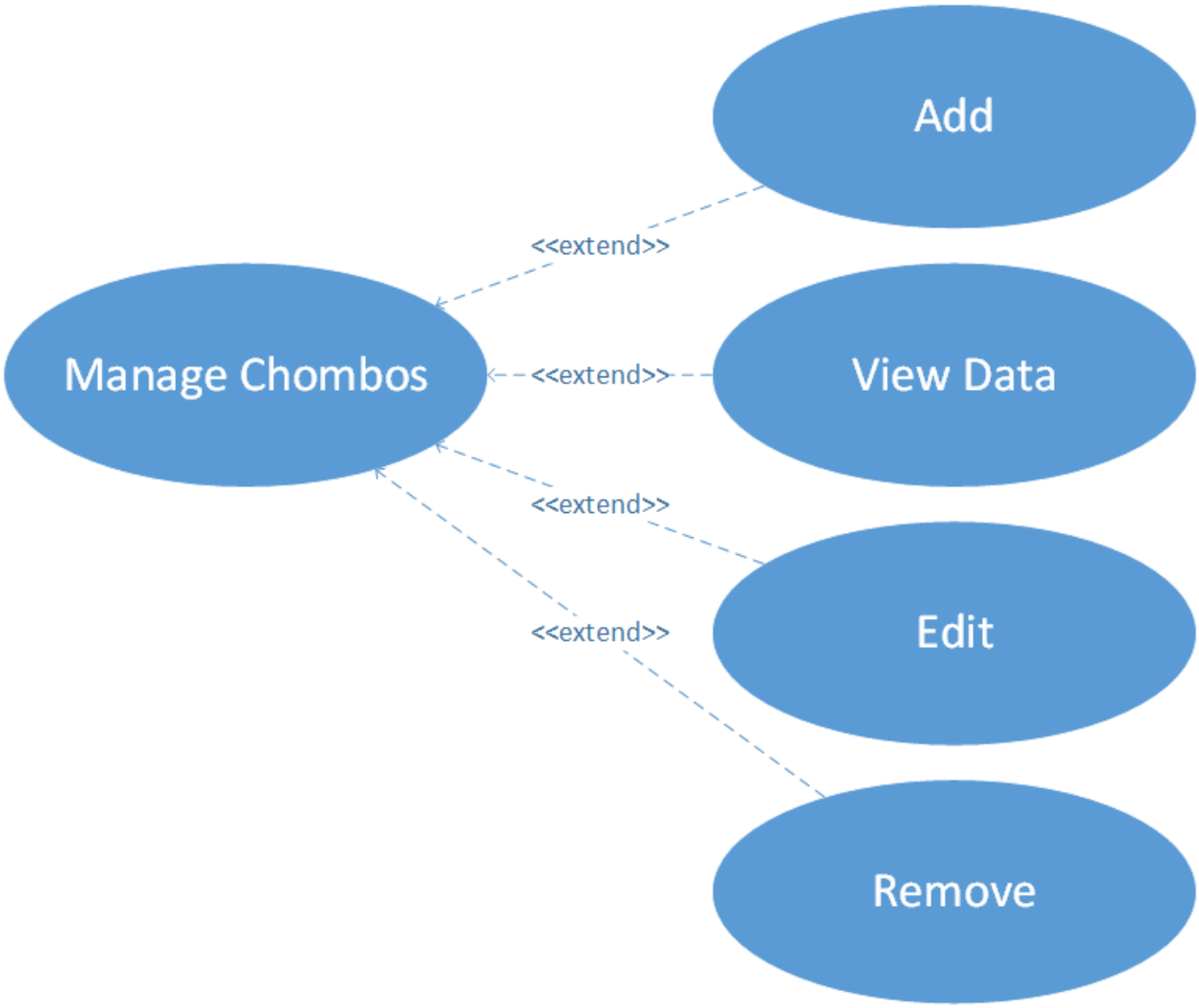


Illustration 13: UC – Manage Chombos

UC_03: Manage Chombos. This diagram describes the options available for managing one chombo from the user. The user has the option to easy add a new product by adding the product ID received from the purchase of new chombo. He can edit only his chombos, but he can view the chombo's data depending on his permissions. As shown above, the user has the option to delete the chombo from his list. The View Product Data and Edit Product nodes are described in the diagrams below.

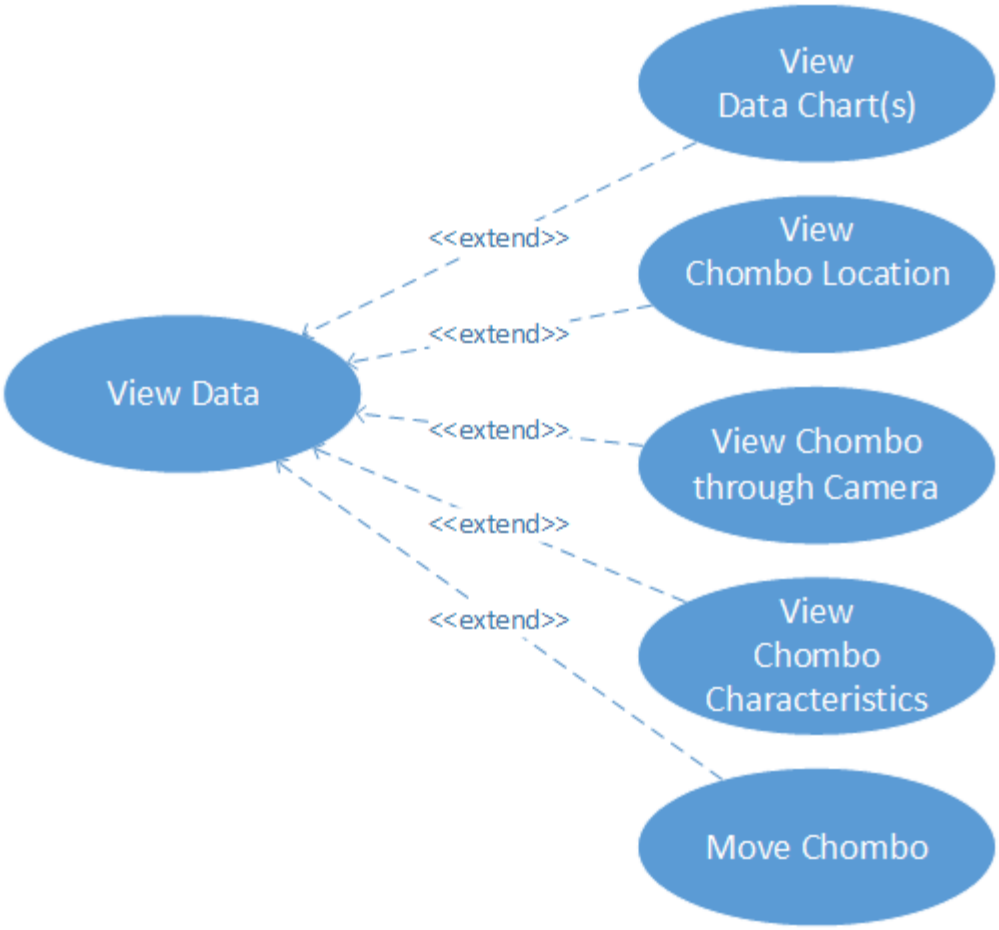
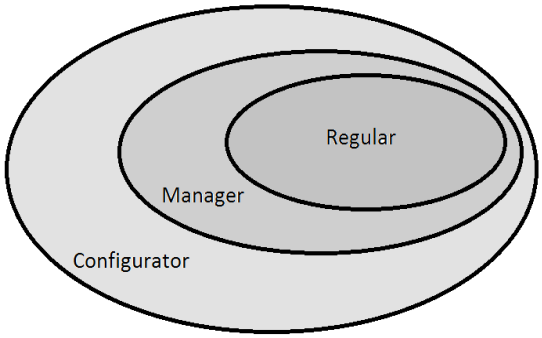


Illustration 14: UC – View Product Data

UC_04: View Product Data. This diagram is referring to all the options available to the user when he views the chombo's data. Every option is separated in a different tab. The charts tab shows the historical data charts of the data from the sensors that the chombo has. On the second tab, the user can track the chombo location using the Google Map. On the third tab, the user can view the camera if it has one, which can help to easy track the plant growth process or make a quck view of its surroundings. The fourth tab contains the characteristics for the chombo. And on the last, fifth, tab the user can move the chombo. All of the tabs above are available ONLY if the user has the permissions for them. There is also an option to set the chombo to be publicly visible and set what actions can be performed by everyone.

In the diagrams below are shown the different types of users in the perspective of the chombo and the default actions that they could perform unless there are some permission changes done by the owner



This image shows that every outer type of user has the same abilities as their inner plus some additional. The owner has full rights, but it's not included in the illustration.

More specifically, the Regular user is only allowed to monitor the Chombos; the Manager can monitor and control; the Configurator can also configure some aspects of the chombo's behavior.

Illustration 15: the layers of access levels

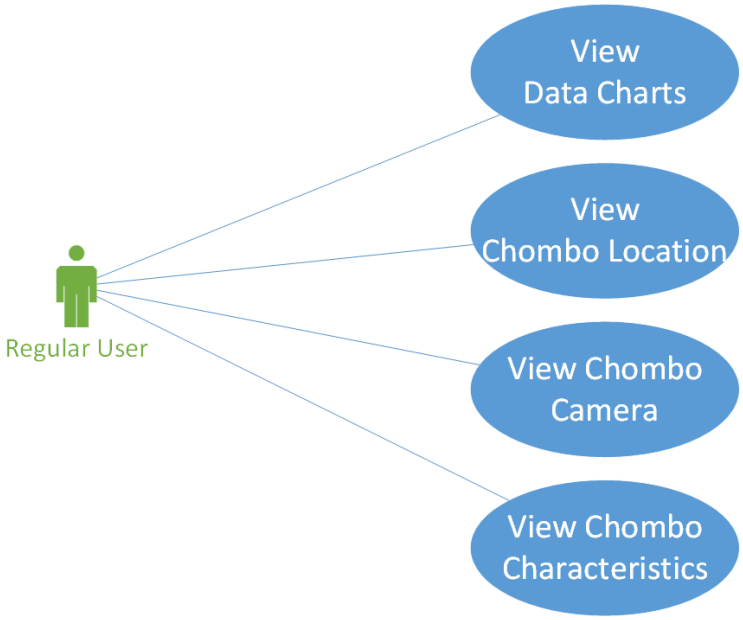


Illustration 16: UC – Regular user

UC_04-1: Regular user. This user can view all the information from the sensors and also see the chombo's characteristics by default. This is the most basic type of user after the public one if the chombo is set to be publicly accessible.

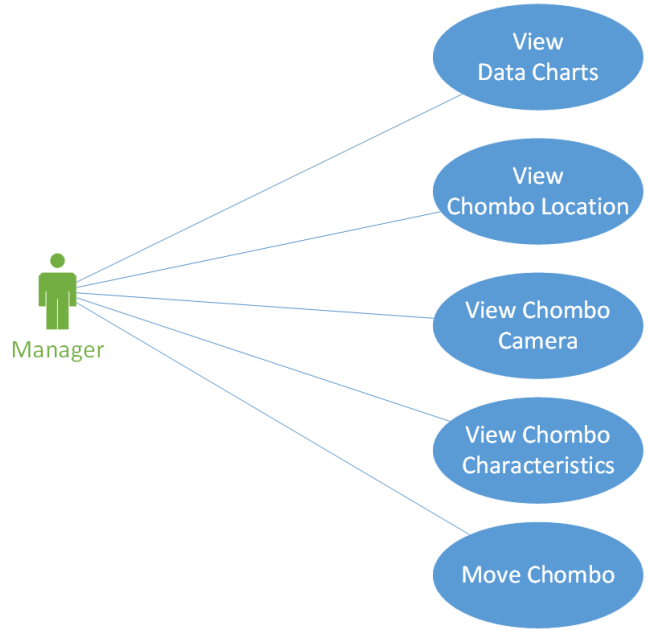


Illustration 17: UC – Manager user

UC_04-2: Manager user. By default, this type of user for the current chombo, can perform all the actions available to the Regular users. He is allowed to control the chombos actuators, which for now refers mostly to moving the chombo.

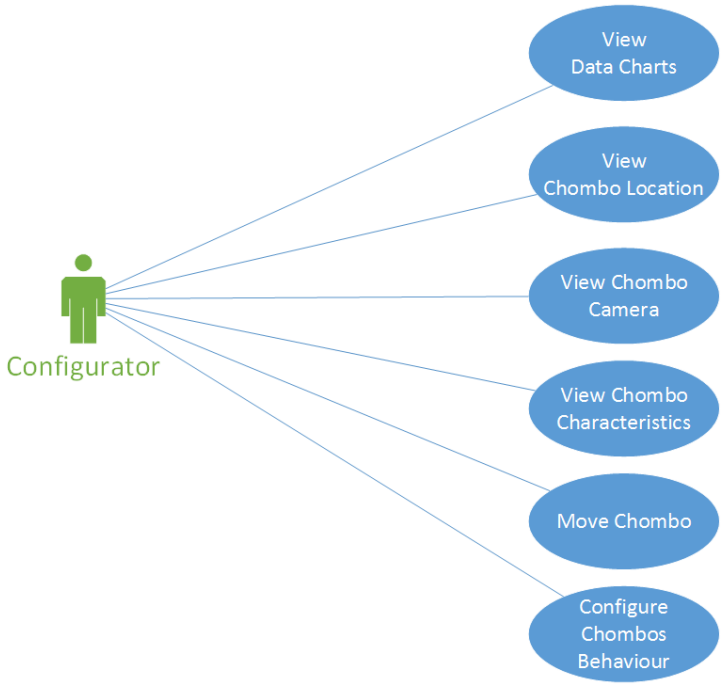


Illustration 18: UC – Configurator user

UC_04-3: Edit Product. The Configurator is the last type of user with the most permissions after the owner. He can perform all the actions from the previous two(three, with the public) users and also configure some aspects of the chombo's behavior.

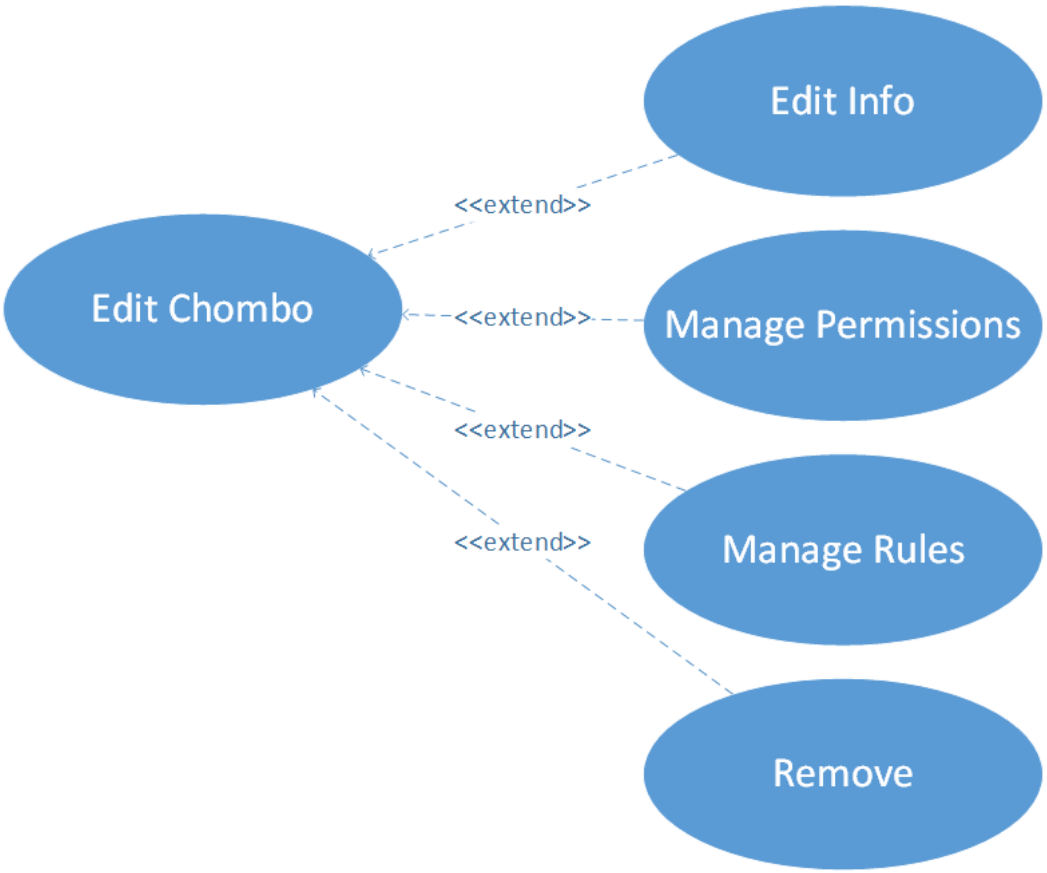


Illustration 19: UC – Edit Chombo

UC_05: Edit Chombo. The user can edit the general information of the chombo and enable/disable it. He can also edit the permissions of a certain chombo for a public or a specific user from his friend list. To give permissions to a friend, the owner should add access level to that friend, which refers to one of the types of users described in the sub-diagrams (UC_04-1 to UC_04-3) above. It is possible to change some of the permissions that are set for the particular access level only for the current friend. For example: the owner can disable the view of the location of the chombo for the friend, while it is still enabled for this type of access level (user). Also, the owner can manage the chombo's rules, which consist of actions that are performed by the applications when a certain event has occurred from the chombo. For example: The application can send an email if the water level in the tank of the chombo Is below 20%.

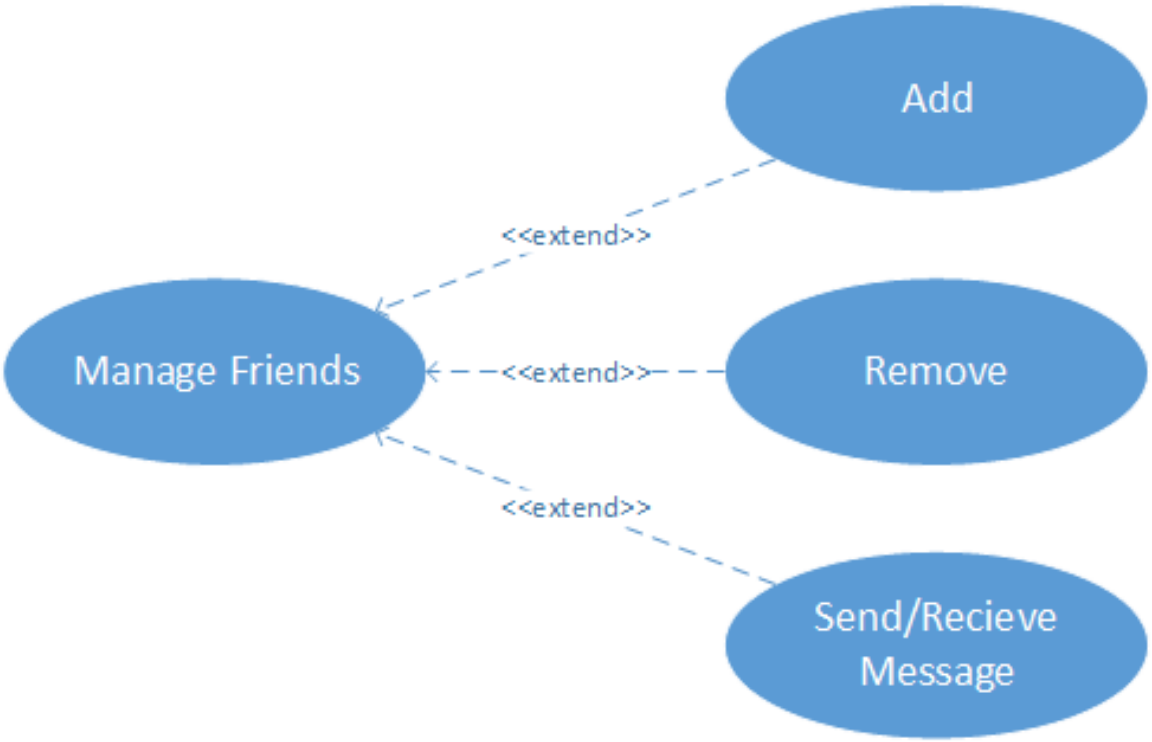


Illustration 20: UC – Manage Friends

UC_06: Manage Friends. This diagram describes the actions that the user can perform when managing his friends. The Add, Edit and Remove of a friend can only be done from the profile page in the Friends tab. The send/Receive Message can be done in any page by using the right menu. This will be shown in the next section of the document.

3.3 Tables of Functionalities

After describing and detailing the use cases, it is necessary to describe which functionalities should be implemented and which can wait, because are not with high priority. The following tables describe what each functionality consists of and shows its priority:

Number of Functionality	UC_01
Name	Register
Description	The user can register through the mobile application by filling the registration form or using his Facebook or Google+ account
Priority	☐High/Essential ☒Medium/Desired ☐Low/Optional

Table 2: Functionality - Register

Number of Functionality	UC_02
Name	View Profile
Description	The user can view his or his friend's profile page. On his profile page, he can manage his chombos and friends and also can edit his profile info
Priority	☒High/Essential ☐Medium/Desired ☐Low/Optional

Table 3: Functionality – View Profile

Number of Functionality	UC_03
Name	Manage Chombos
Description	The user must have the option to manage chombos, considering add new product, view product data, edit the product or remove it.
Priority	☒High/Essential ☐Medium/Desired ☐Low/Optional

Table 4: Functionality – Manage Chombos

Number of Functionality	UC_04
Name	View Product Data
Description	The user can view the data of a selected chombo from his list. His permissions depend on that if he is the owner or if not, then what access level and custom permissions are set for him.
Priority	☒High/Essential ☐Medium/Desired ☐Low/Optional

Table 5: Functionality – View Product Data

Number of Functionality	UC_05
Name	Edit Product
Description	The owner of the chombo can edit it by changing its general information. Also he has the oprtions to manage its permissions and set some rules.
Priority	☒High/Essential ☐Medium/Desired ☐Low/Optional

Table 6: Functionality – Edit Product

Number of Functionality	UC_06
Name	Manager Friends
Description	The user can manage his friends in his profile page by adding new friends or removing an existing one. Also, at any time he has access to the message menu where the user can send/receive messages to/from friends.
Priority	☐High/Essential ☒Medium/Desired ☐Low/Optional

Table 7: Functionality – Manage Friends

3.4 Conclusions

This section contains details about how the different functionalities in the project should work and their overall description, and priority. The main use-cases are shown separately, but can also be viewed in one big diagram. With their help, we can view the structure of the project in depth, so that we know how to build it later.
Another thing that was made clear from the diagrams is that we can view the user from two perspectives – from the application and from the chombo. From the application's point of view the users are all the same, but from the chombo's there are several types of users depending on their access level that is set from its owner.
The tables of functionalities easily summarize those functionalities and show their priority. As shown, they include all the functionalities that are described in the project's objective plus some additions that are essential for a mobile application, like the edit of the settings. There is no functionality with low priority, but some of them are Middle and the rest of them are with High priority. The registration and the management of friends is middle priority, because it is not essential to the main purpose of the application and can be left until those with high priority are fully implemented. Of course, more functionalities can be added to the project, but the time is not enough for all of them, so they can be left as future implementations or maybe implementations for the real application.

4. System Design

4.1 Introduction

Now that we know what the mobile application is for and how it should work from the previous sections of this document, the next step is to design the structure of the project in both visual and technical aspect. Designing the structure and the business logic of the application is easy, because we already know what are we going to be using and also know the functionalities that we want to achieve. However, it is more important to have a good visual design, because this is the part of the application that is closest to the users and if it is not appealing and easy to use, then it can lead to dissatisfaction.
In the conceptual specification is defined the whole structure of the project that this application is into and the part that the current project represents.
At the formal specification is defined the design of this project and how it should be developed.

4.2 Conceptual Specification

The whole system is revolved around deliberative control that makes the communications between the chombos and the servers as well as between the chombos. The purpose of this project is to contribute in creation of an intelligent control layer (supervision by the user) so that this system can be managed easier.

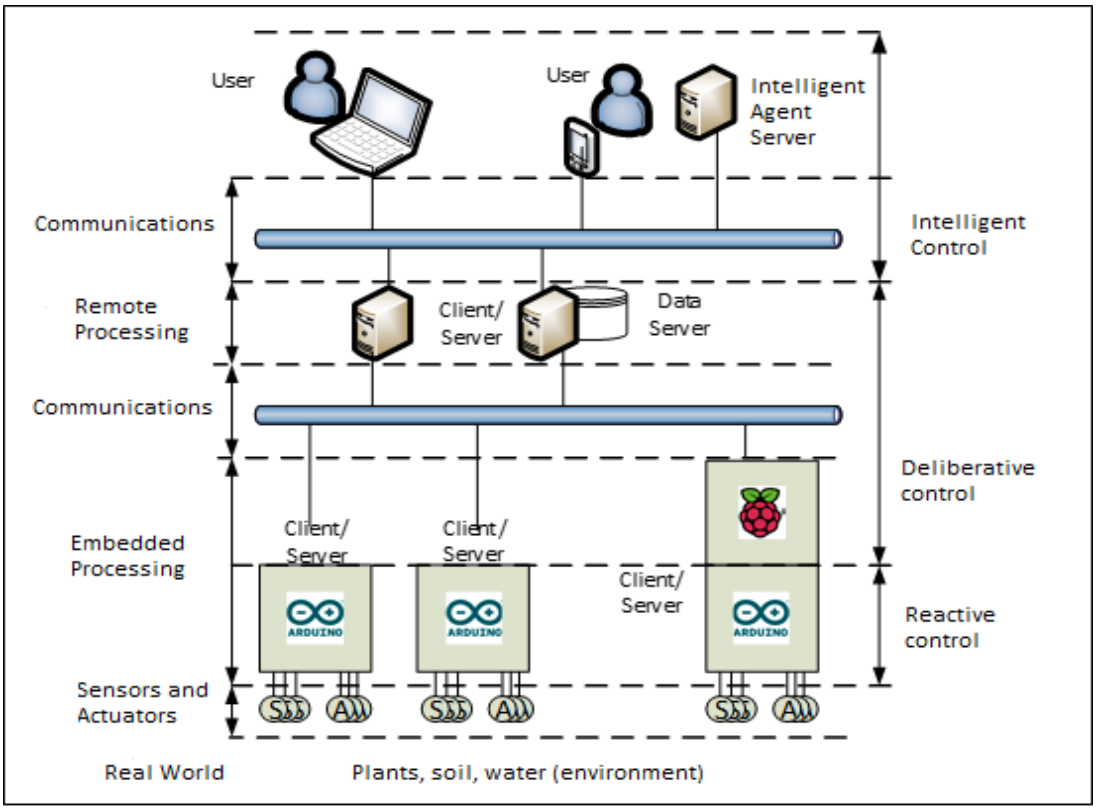


Illustration 21: Architecture of the system

The illustration above shows the structure of the whole system that this project takes part of. At the bottom are illustrated the chombos with their sensors that record data from the environment and the actuators. They are programed with Arduino and by using deliberative control, they communicate with the remote processing units (the servers) at the middle of the illustration. At the top are the clients (users). This project takes part in developing a mobile application that makes the communication between the servers and the end user (using smartphone).

This project is actually part of a bigger one, that is dedicated to creating the whole "Chombo" system and the other parts of it are also developed by other students and is supervised by teachers from the UPV in Valencia.

4.3 Formal Specification

For the design of the architecture it has been has chosen the three layers architecture: Data, Business and Presentation. This is an architectural deployment style that describe the separation of functionality into layers with each segment being a tier that can be located on a physically separate computer. Each tier is developed and maintained as an independent tier.

To make a good design, one must carefully define each of the layers. In this case the most important layer is the presentation layer, because it is the layer closer to the user and must be visually appealing and easy to understand, and navigate. Each layer of the architecture is detailed below.

4.3.1 Data Layer

For the saving of the data it is going to make use of the MySQL database configured on the server. For the communication between the server and the database are used PHP model classes. In some case parts of the information are stored in the form of cookies or cache. It is important to note that most of the data is stored on the server and only some of the information is stored on the user's device.

4.3.2 Business / Logical Layer

The most important thing in the mobile application is to offer the user the opportunity to manage his chombo by editing its details or change its permissions. The user can also view the chombo's data like sensor data, location, camera or characteristics. Also, he can control its movement.

The following sequence diagrams represent the workflow of some of the main functionalities of the mobile application.

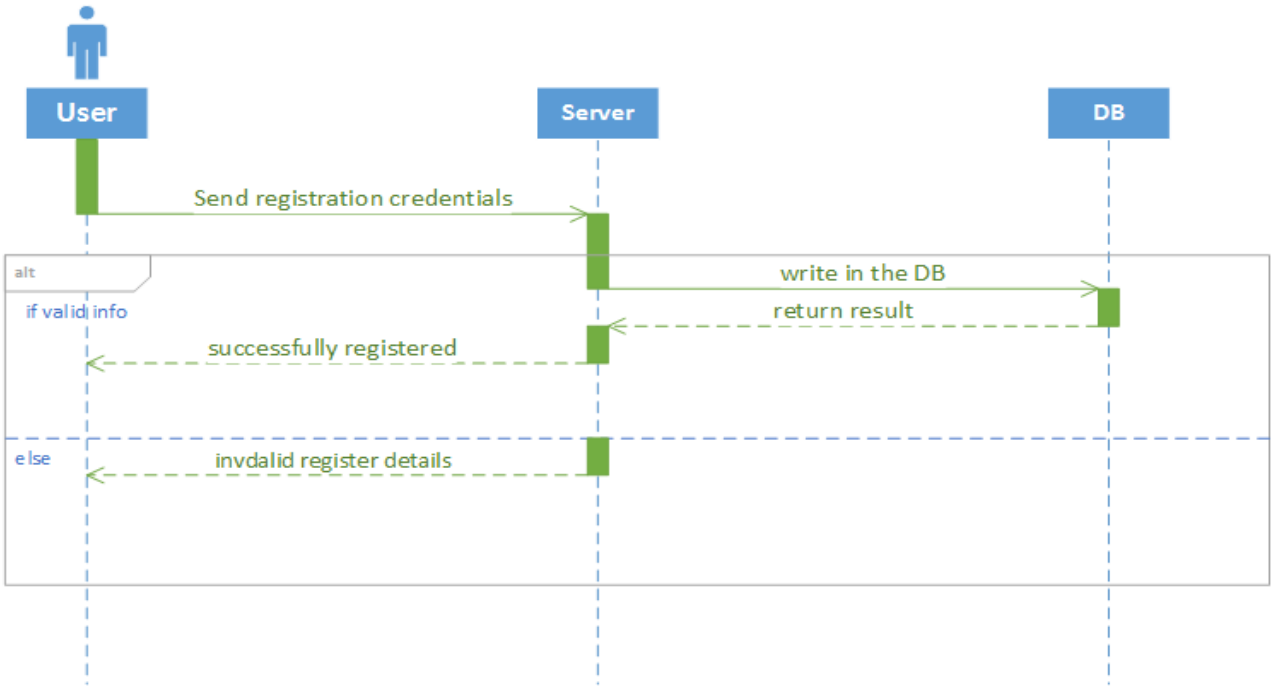


Illustration 22: Sequence - Register

This first diagram reviews the register function. There are two cases. For the first case if the information is valid - every time when the user fills the register form and submit it he sends the credentials to the server. Then the server sends the data in the database. Then it sends back a result to the database and the server returns a message to the user that he is successfully registered. In the other case if the user input invalid information that is required or the username already exist the server sends back message to the user that the details are invalid and he's not able to register.

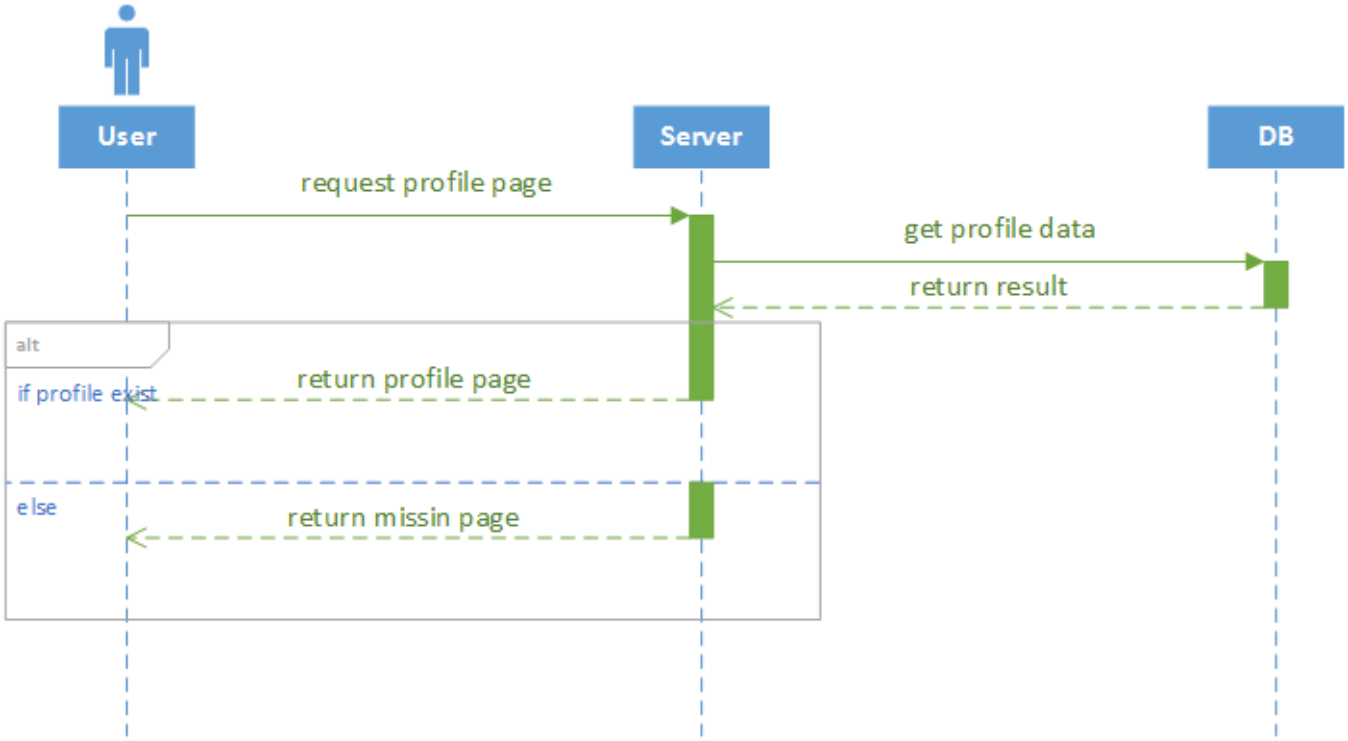


Illustration 23: Sequence – View Profile

This diagram reviews the view profile function. When the user enters in a friend's profile he sends a request to the server. Then the server makes a request to the database which checks for the profile data. Then the database returns result. After this there are two cases. If the profile exist the server returns the profile page to the user. If not, then it returns an error that say that the page is missing.

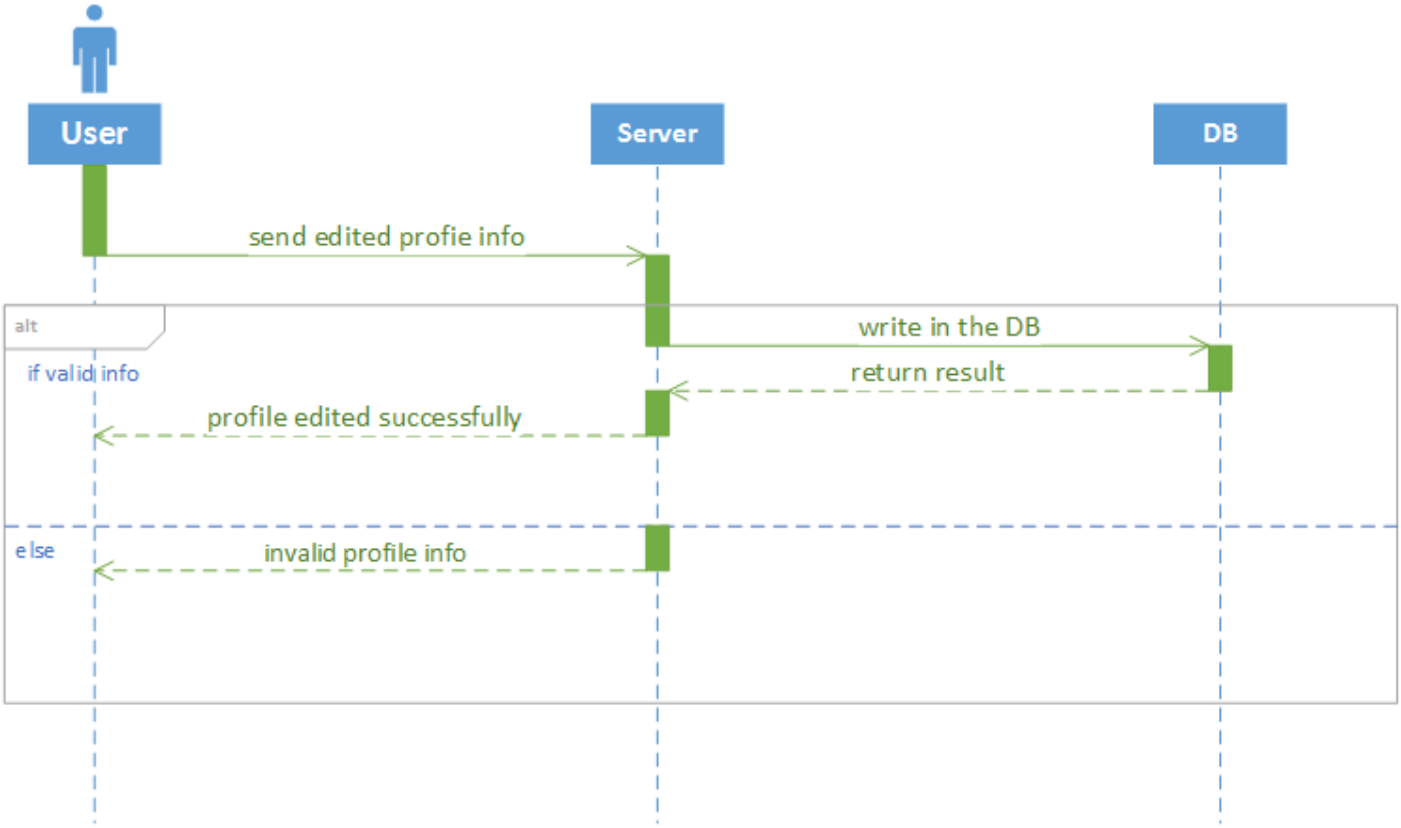


Illustration 24: Sequence – Edit Profile

This diagram reviews the view profile edit function. When the user edits his profile info he sends a request to the server. Then the server sends a request to write the changes in the database. Then the database returns the result. After this there are two cases. If the profile info is valid the server sends a message to the user that profile is edited successfully. If not, the server sends a message to the user that says the profile info is not valid.

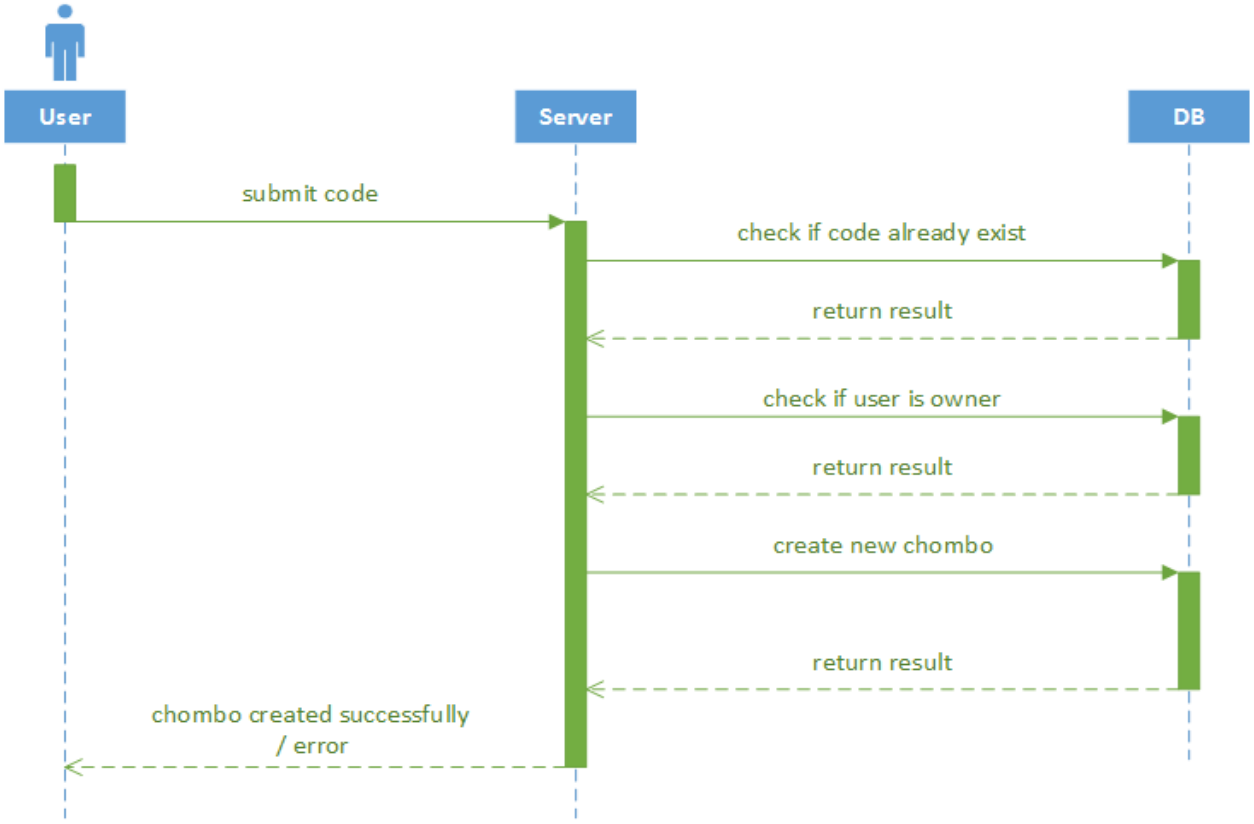


Illustration 25: Sequence – Add Product

This diagram is associated with the managing of a products or more precisely add a product. After the user has bought the chombo and he got the key, he adds it to his profile. When he submits it, he sends request to the server. Then the server makes a request to the database which checks if this code already exists. If the code didn't exist then the user is redirected back to his profile page. If exist then send a result to the server. Then the server makes a new request which checks if the user is the owner of this code. If owner, then return result to the server. Then the server sends a new and final request to create new chombo. The database return

result to the server and the server send a message to the user that the chombo is added successfully.

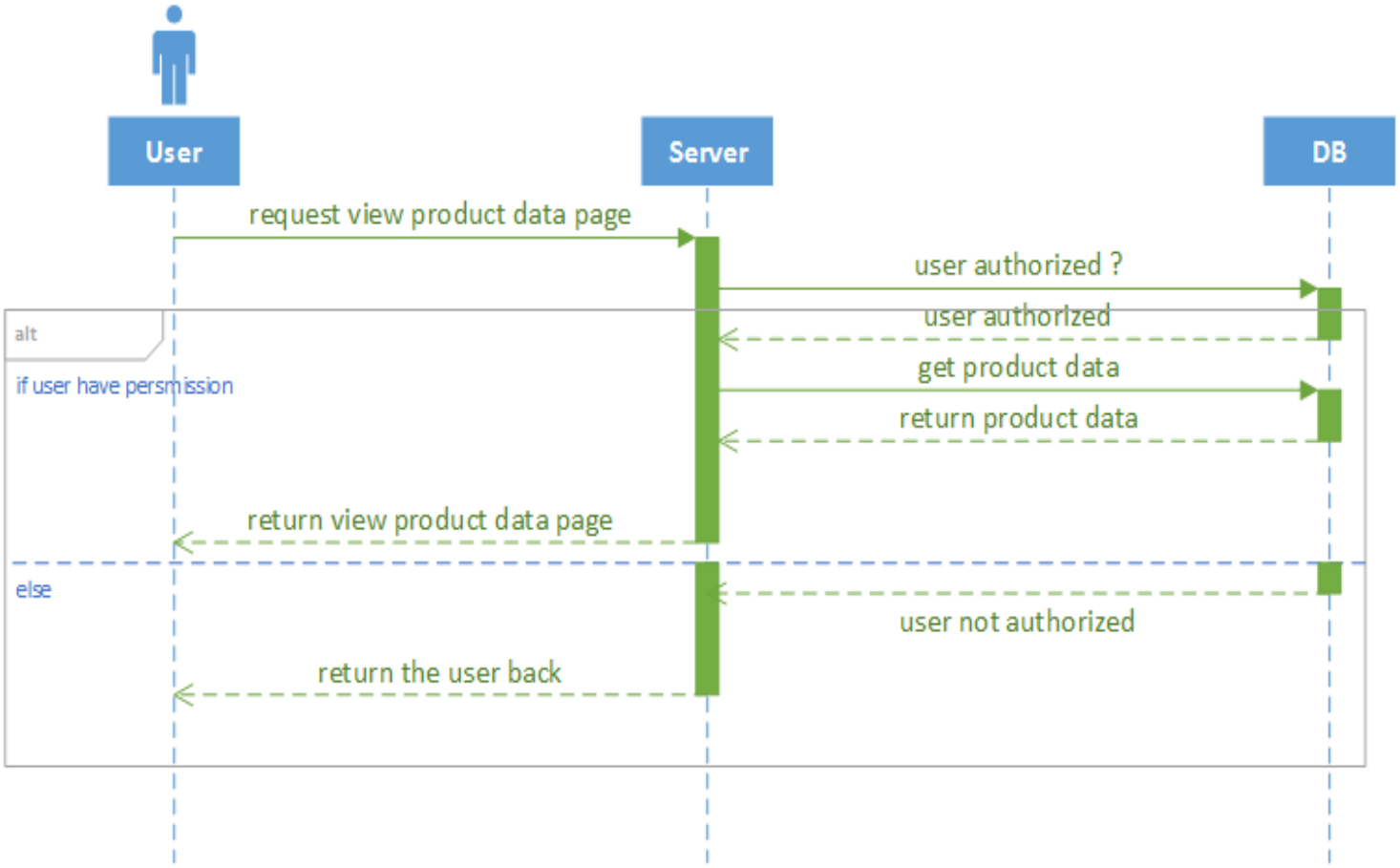


Illustration 26: Sequence – View Product Data

This diagram is associated with the managing of a products or more precisely view product data for a product. When the user enters the product data page he sends a request to the server. Then the server makes request to the database which checks if the user is authorized to view the data. This check is general which means that it doesn't matter if the user is owner or not an owner. There are two cases. If the user is authorized the database returns a result to the server. Then the server makes a new request to the get the product data for the chombo. The database

returns result to the server and the server sends back the page of the product. If the user is not authorized, he is returned to the previous page.

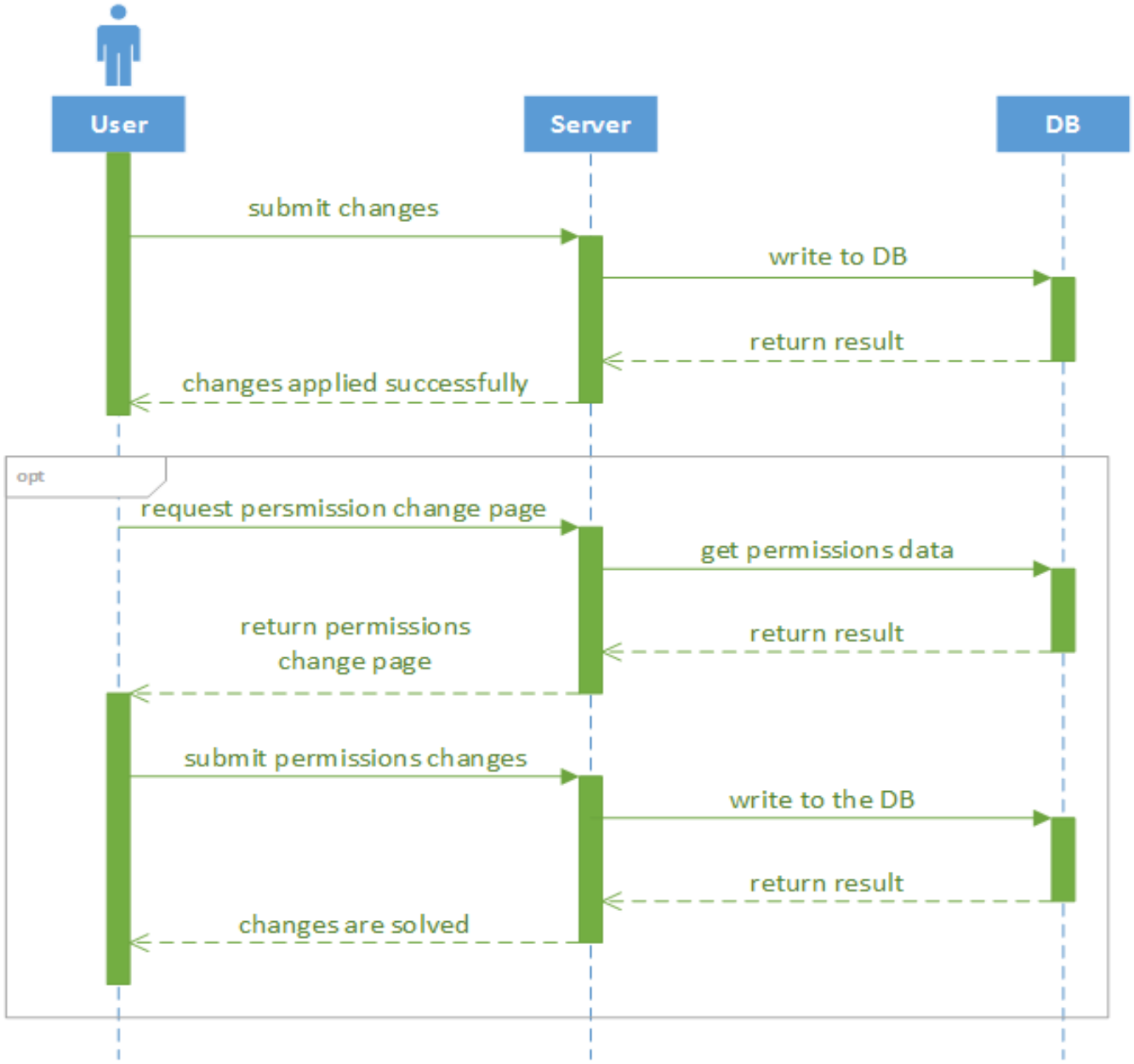


Illustration 27: Sequence – Edit Chombo

This diagram is associated with the managing of a products or more precisely edit a product. When the user edits some information of the chombo he sends a request to the server. Then the server sends request to write the changes in the database and then the result is returned. The server sends a message to the user that the changes are successfully applied. An optional

case is if the user wants to change the permission of the chombo when he edits it. A request is again sent to the server. Then the server makes request to get the permissions data. The database sends results back and the server send the permission change page. If the user submits the permission changes they process the same way as the edit process.

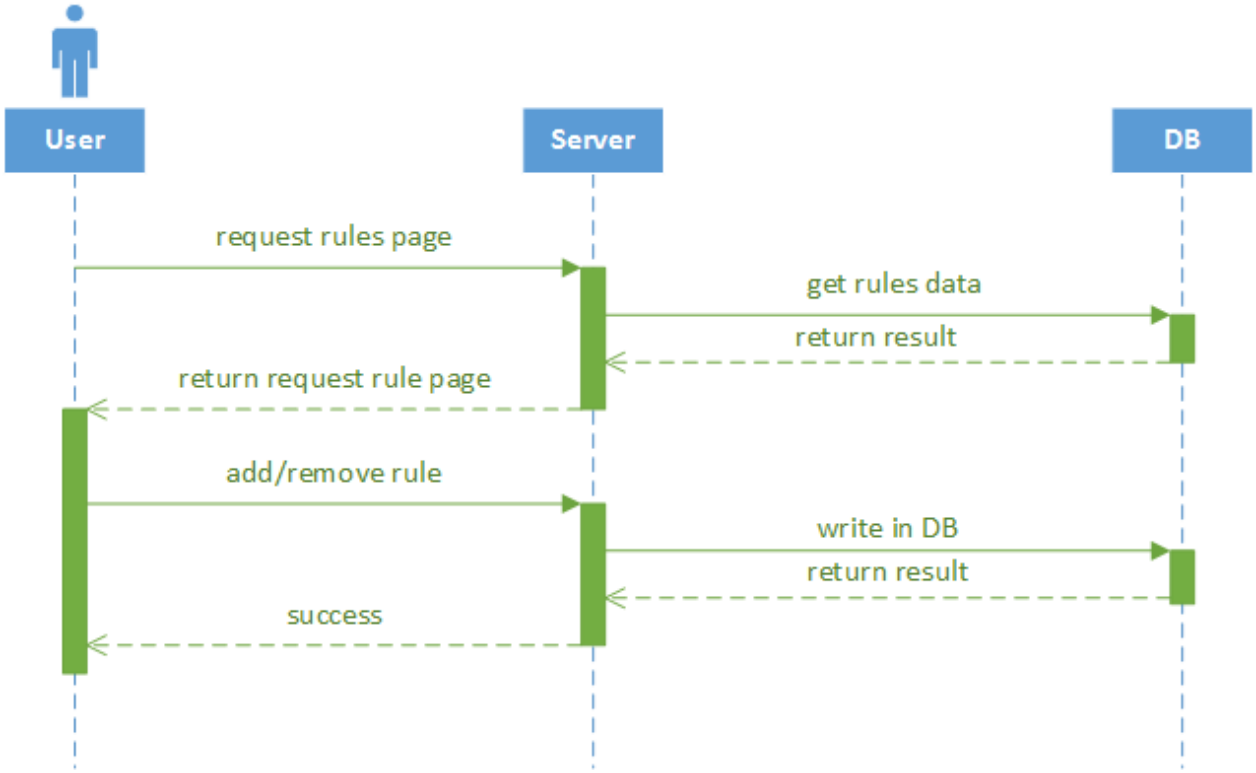


Illustration 28: Sequence – Add/Remove Rules

This diagram is associated with the add or remove rules. When the user wants to enter the rules page he sends a request to the server. Then the server sends a request to the database to get rules data. The database sends back result and then the server return the page to the user. After this the user send a new request to add/ remove rule to the server. The server makes request to the database to add or remove a rule. Then the database returns result and the server sends a message to user that the add or remove is successful.

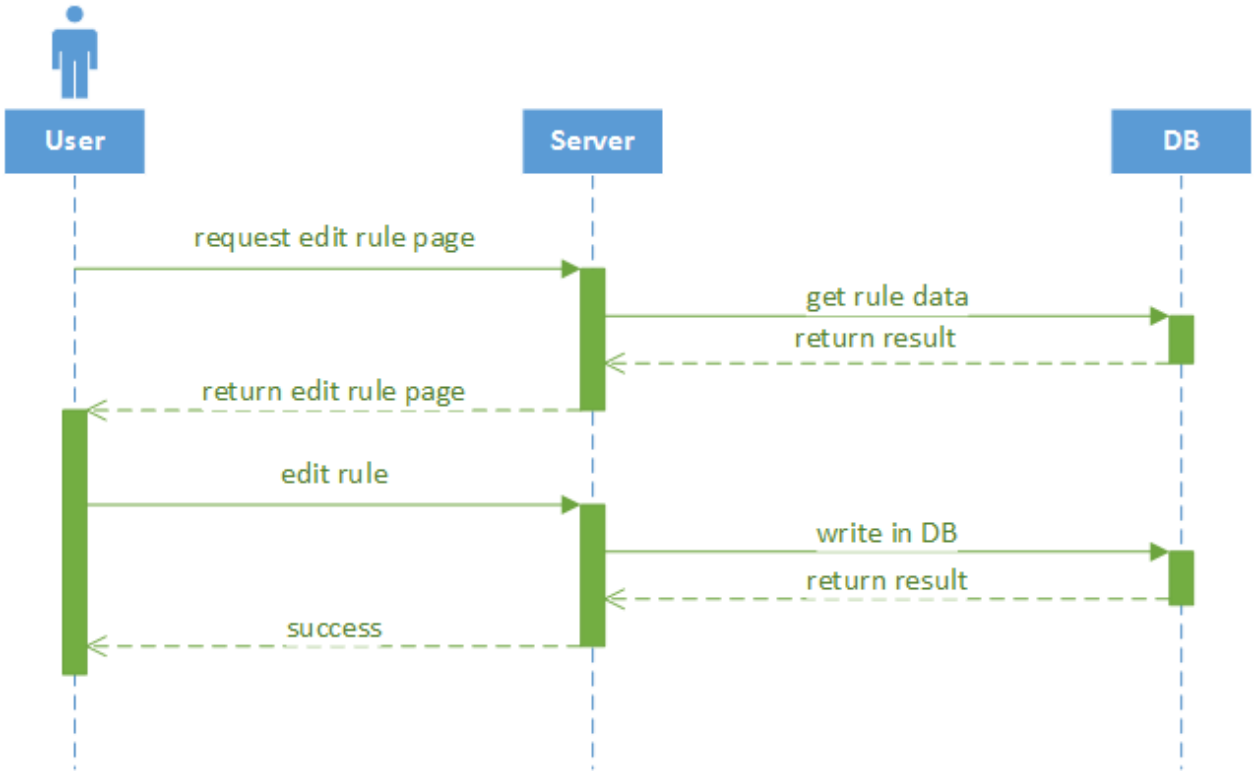


Illustration 29: Sequence – Edit Rules

This diagram is associated with the edit rule. When the user wants to enter the edit rules page he sends a request to the server. Then the server sends a request to the database to get rules data. The database sends back result and then the server return the page to the user. After this the user send a new request to edit rule to the server. The server makes request to the database to edit a rule. Then the database returns result and the server sends a message to user that the edit is successful.

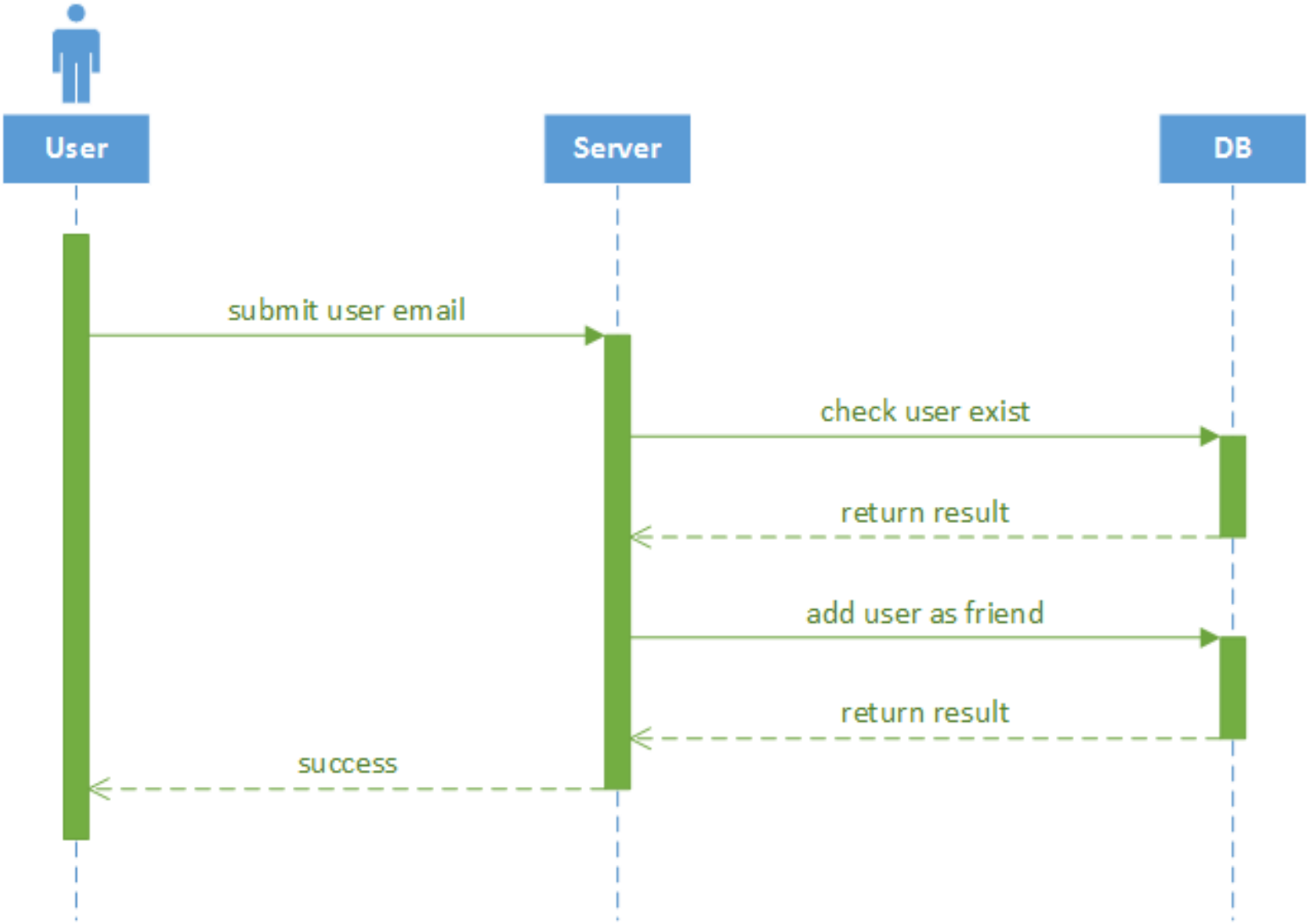


Illustration 30: Sequence – Add Friend

This diagram is associated with the friend managing or more precisely adding a friend. When the user wants to add a friend via email he sends a request to server. Then the server a request that

checks if the user exist. If yes, then send back a result to the server. Then the server makes a new request to the database to add the founded user as friend. The database again sends back a result and the server send message to the user that the user is successfully added as friend.

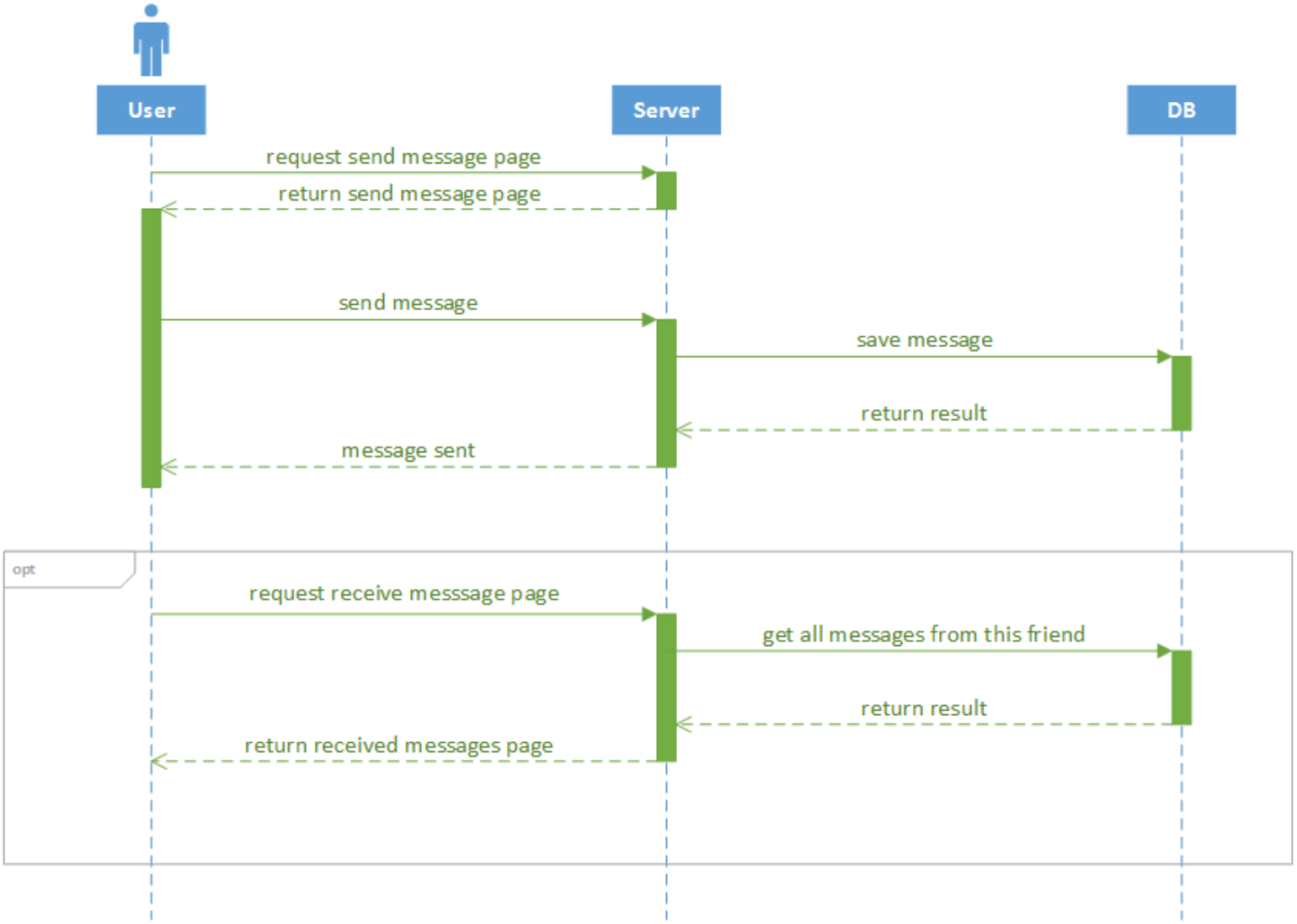


Illustration 31: Sequence – Send Message to Friend

This diagram is associated with the friend managing or more precisely sending or receiving message to/from user. When the user wants to enter the send message page he sends a request to the server and the server returns the page to the user. Then when the user wants to send a message he sends a request to the server. After that the server send request to the

database to save the message. The database returns the result and the server send back a message to the user that the message has been send successfully. An option case is when the user wants to enter the page for the received messages. The process is the same as the previous, but this time the server sends to the user the received messages page.

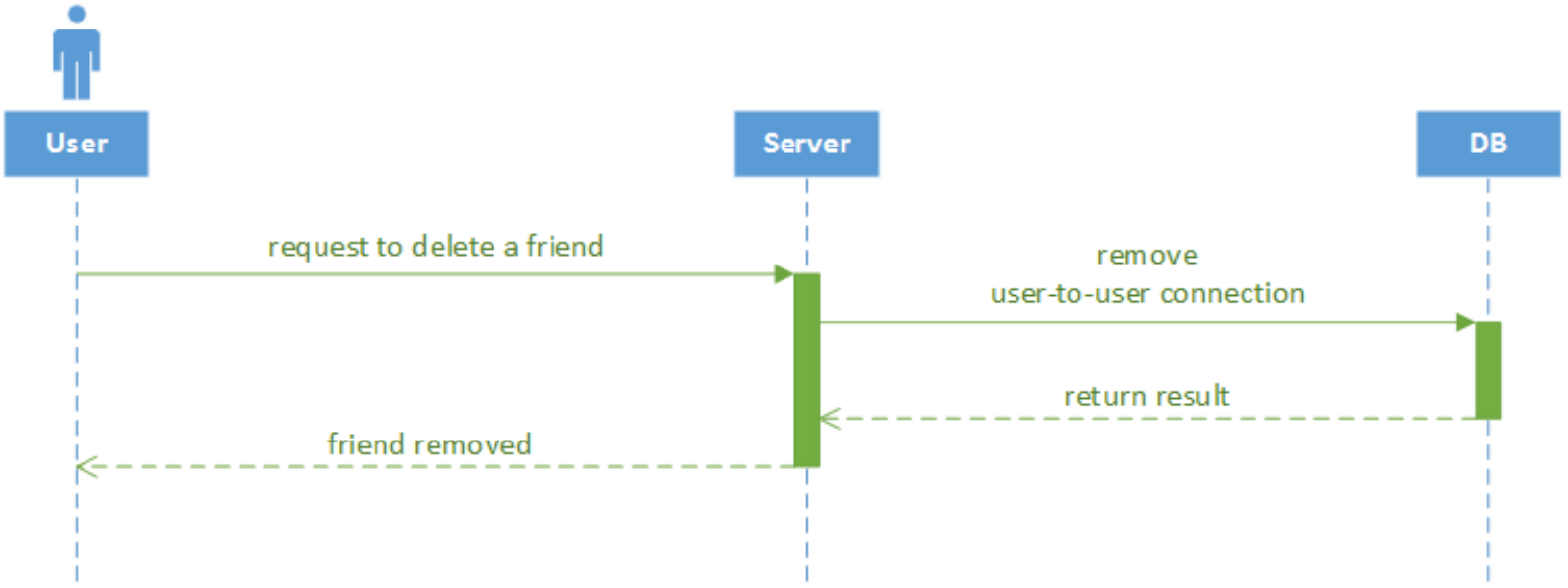


Illustration 32: Sequence – Remove Friend

This diagram is associated with the friend managing or more precisely removing friend from friend list. When the user wants to remove a friend, he sends a request to the server. Then the server send request to the database to remove the user-to-user connection. After this the database sends result and the server sends a message to the user that the friend is removed.

All the functions derived from the sequence diagrams described above are detailed below.

Name	Register		Number	01
Input	Process	Output	Actors	
Firstname Lastname Username Email Confirm email Password Confirm password	Test if the user can register in the system.	YES	Unregistered User	

Table 8: Function - Register

Name	View Profile		Number	02
Input	Process	Output	Actors	
User_id	Test if the user can view profile info	YES	Owner	
		YES	Registered User	
		NO	Unregistered User	
Firstname Lastname Email Confirm email Password Confirm password	Test if the user can edit his profile info	NO	Registered User	

Table 9: Function – View Profile

Name	Manage Products		Number	03
Input	Process	Output	Actors	
Product_key	Test if the user can add product	YES	Owner	
		NO (the key belongs to the owner)	User that is not Owner	

Product_id *Title* *active* *description*	*Test if the user can edit a product*	*YES*	*Owner*

Table 10: Function – Manage Products

Name	Manage Rules		Number	04
Input	Process	Output	Actors	
Chombo_id rule	Test if the user can add rule	NO	Owner	
Rule_id rule	Test if the user can edit rule	NO	Owner	
Rule_id	Test if the user can remove rule	NO	Owner	

Table 11: Function – Manage Rules

Name	View Product Data		Number	05
Input	Process	Output	Actors	
Product_id	Test if the user can see the product data.	YES	Owner	
		YES	Authorized User	
		NO	Unauthorized User	

Table 12: Function – View Product Data

Name	Manage Friends		Number	06
Input	Process	Output	Actors	
email	Test if the user can add a friend	NO	User	
user_id message	Test if the user can send or receive a message	NO	User	
User_id	Test if the user can remove a friend	NO	User	

Table 13: Function – Manage Friends

4.3.3 Presentation Layer

This layer includes the preliminary designs of the application. It is important to make a good design that meets certain requirements. The application should be easy to use for the user. It should also display all the necessary information for the user. Therefore, an application with a simple and ordered structure must be designed.
Below are the designs of the interfaces that a user can find in the application with descriptions, which represent some of the functionalities.

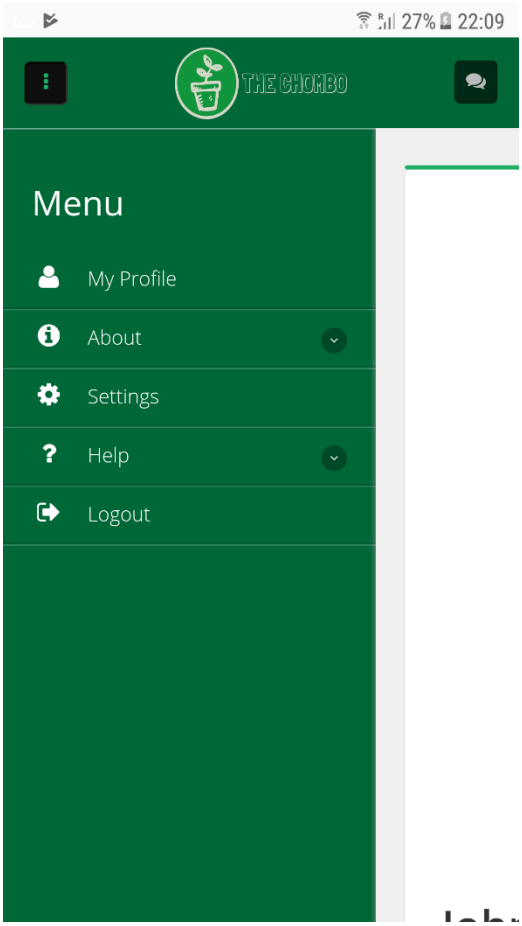


Illustration 33: Interface of the Application - Menu

This illustration shows the main menu of the application. It represents the UC_00 with the addition of the about pages and the logout option. As shown, the user has the option to enter his profile page, to enter the settings page and to access customer support or social networks from the help dropdown.

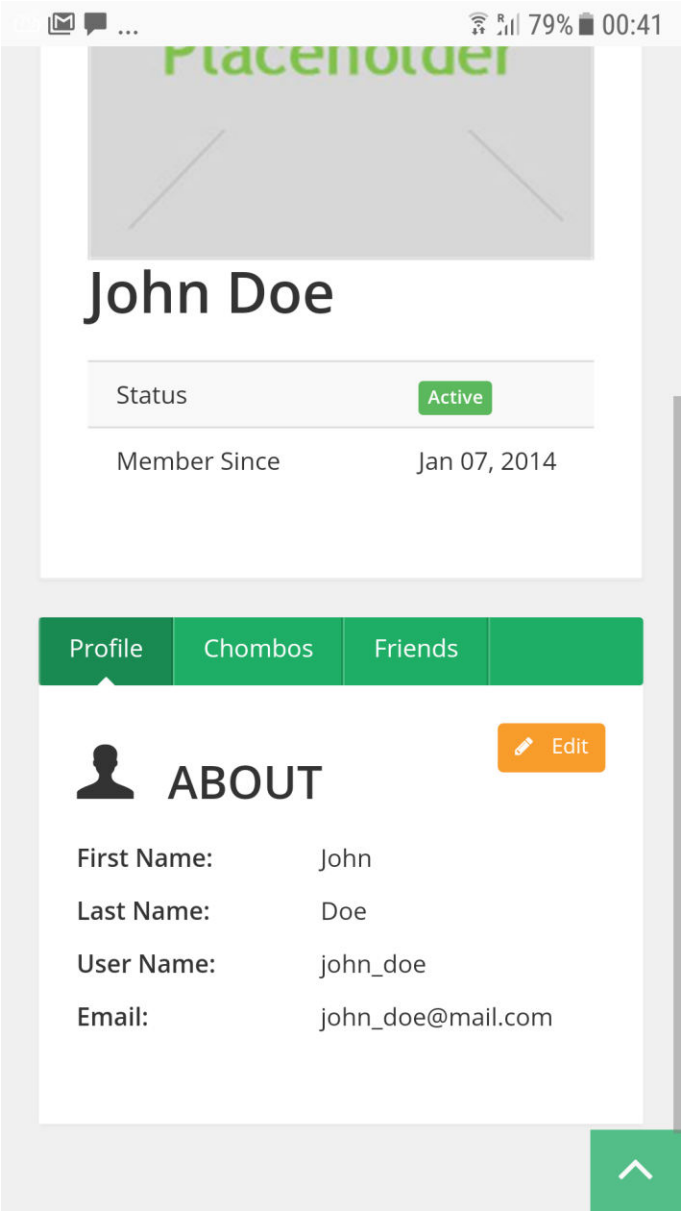


Illustration 34: Interface of the Application – Profile Page

This illustration represents the profile page when the user enters it. It shows his image at top with his name and some additional details and at the bottom mart there are these three tabs: Profile, Chombos and Friends. The profile tab shows the users personal information and also offers a button to go to the edit page. The other two tabs are described bellow

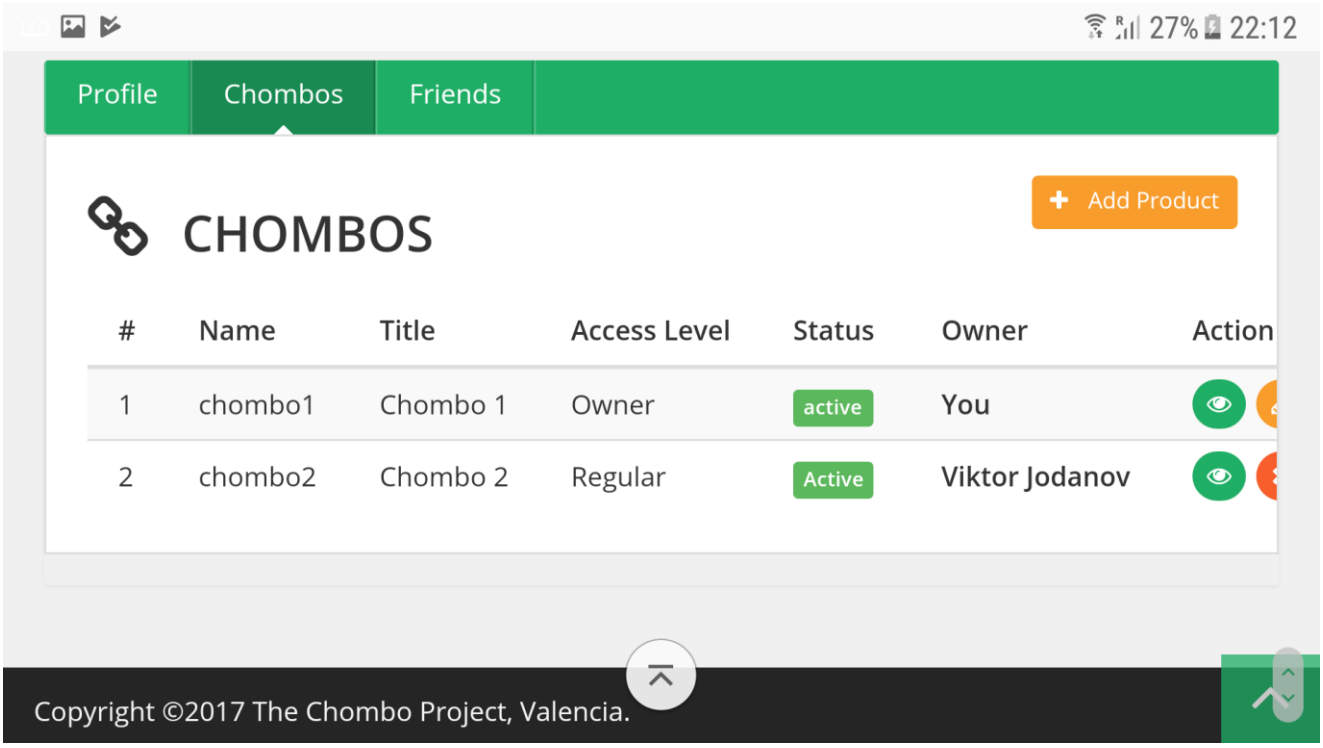


Illustration 35: Interface of the Application – Profile Page – Chombos tab

This illustration represents the chombos tab in the user's profile page. It consists of a table with all the chombos that he has access to. In the table for every chombo it is shown who is the owner (it is showing "You" if the current user is the owner), and also the access level of this user. The actions that the user can perform depend on the user's permissions and were described in the use-case diagrams above. Also, you can access the add product page in order to register a new chombo using the code provided after the purchase.

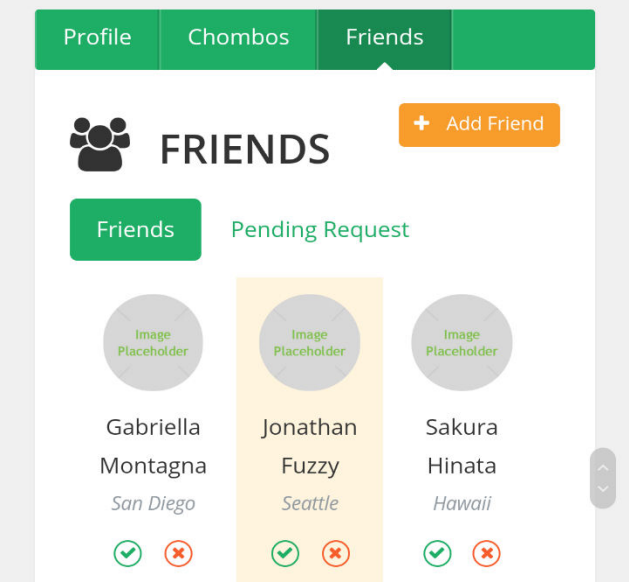


Illustration 36: Interface of the Application – Profile Page – Friends tab

This illustration represents the friends tab in the user's profile page. It shows you all your friends and allows you to accept new ones. You can also add new friend by entering the add friend page using the button above.

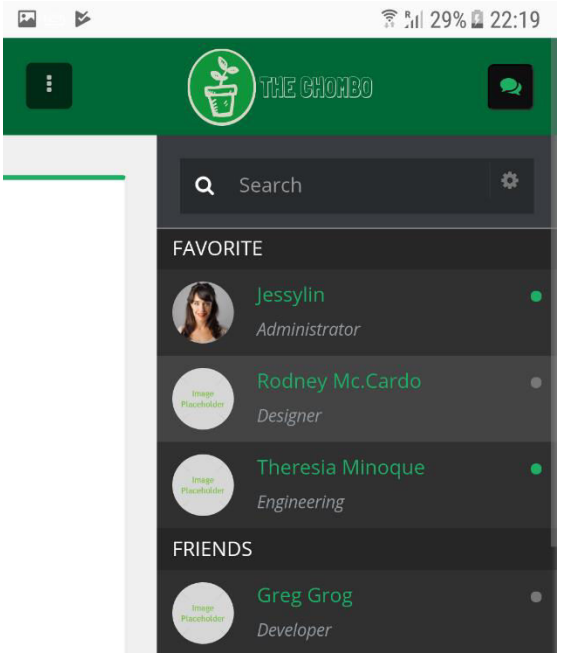


Illustration 37: Interface of the Application – Friends Sidebar

This illustration represents the message feature of the application. The button on the top-right corner of the header section opens a side menu with all your friends and their status. You can also use the search t sy access.

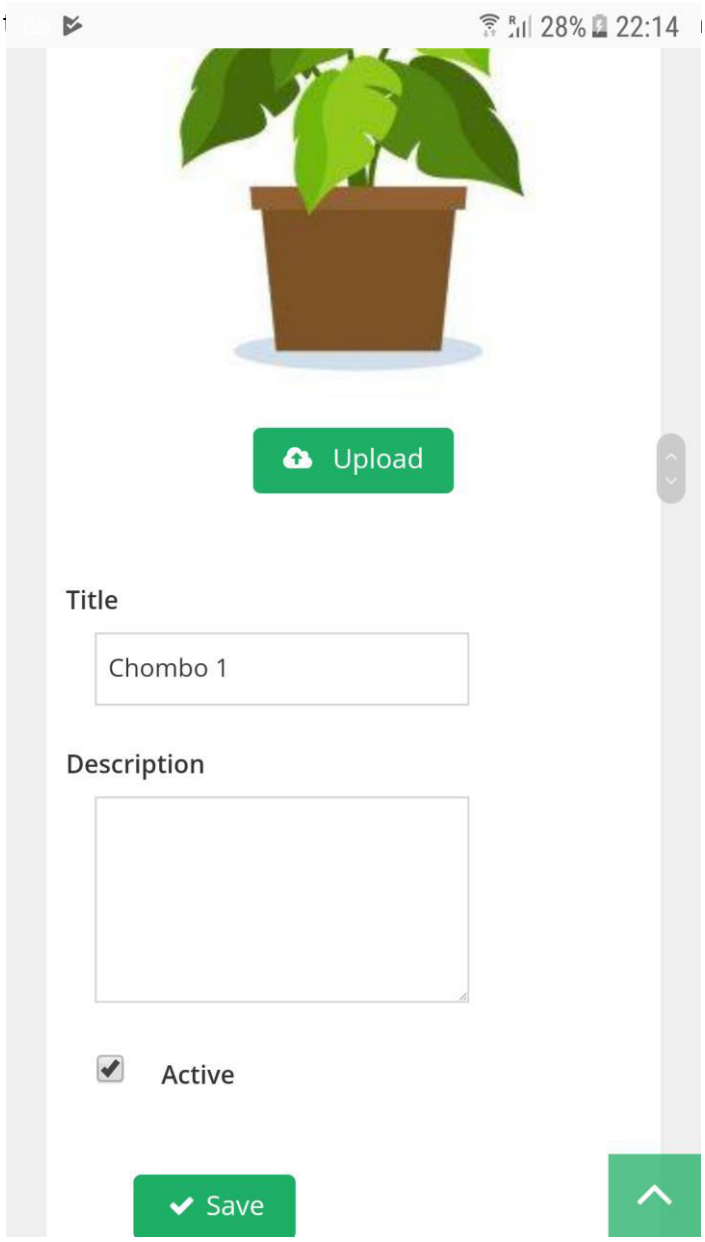


Illustration 38: Interface of the Application – Edit Chombo Page 1

This illustration represents the chombo's edit page which allows the owner to change some of it's parameters and also enable/disable it for the other users by editing the active parameter.

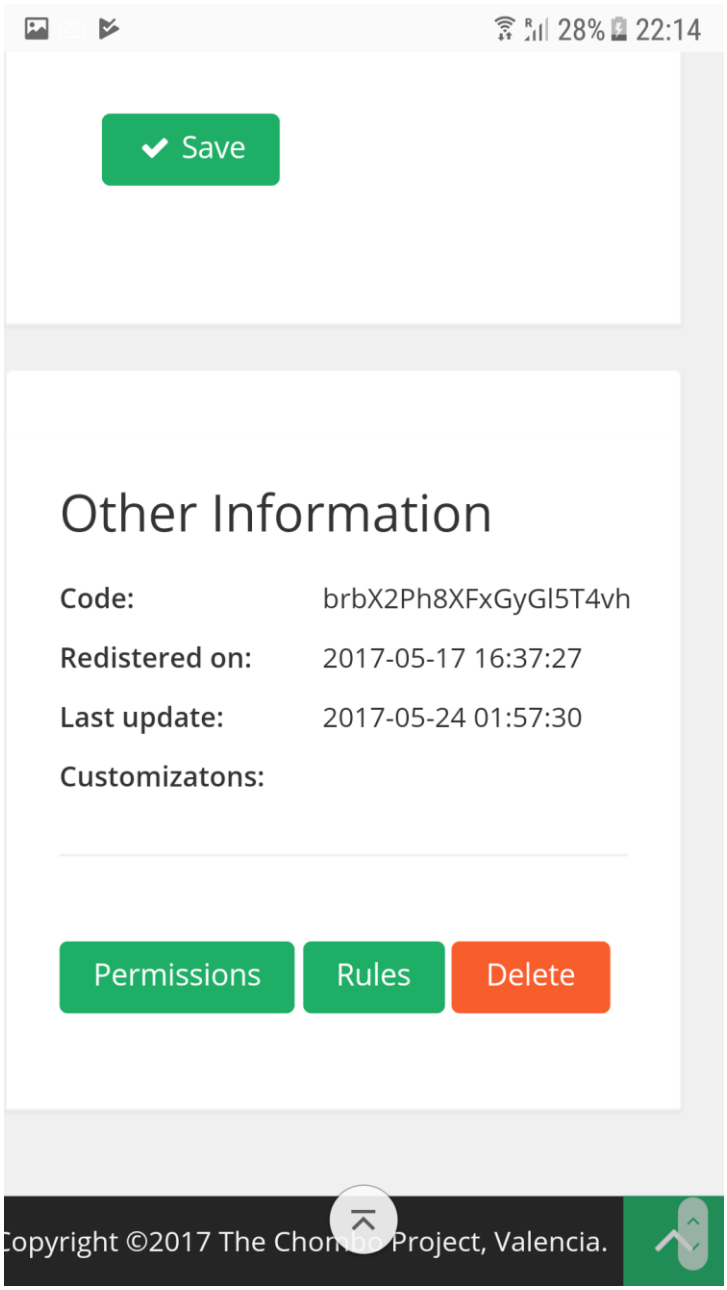


Illustration 39: Interface of the Application – Edit Chombo Page 2

This illustration represents the second part of the chombo's edit page. It contains some other personal information and also has the links to some other pages like the permissions and rules pages. They will be described in the next illustrations. There is also an option to delete the chombo from here, which will redirect you back to your profile after confirming the deletion.

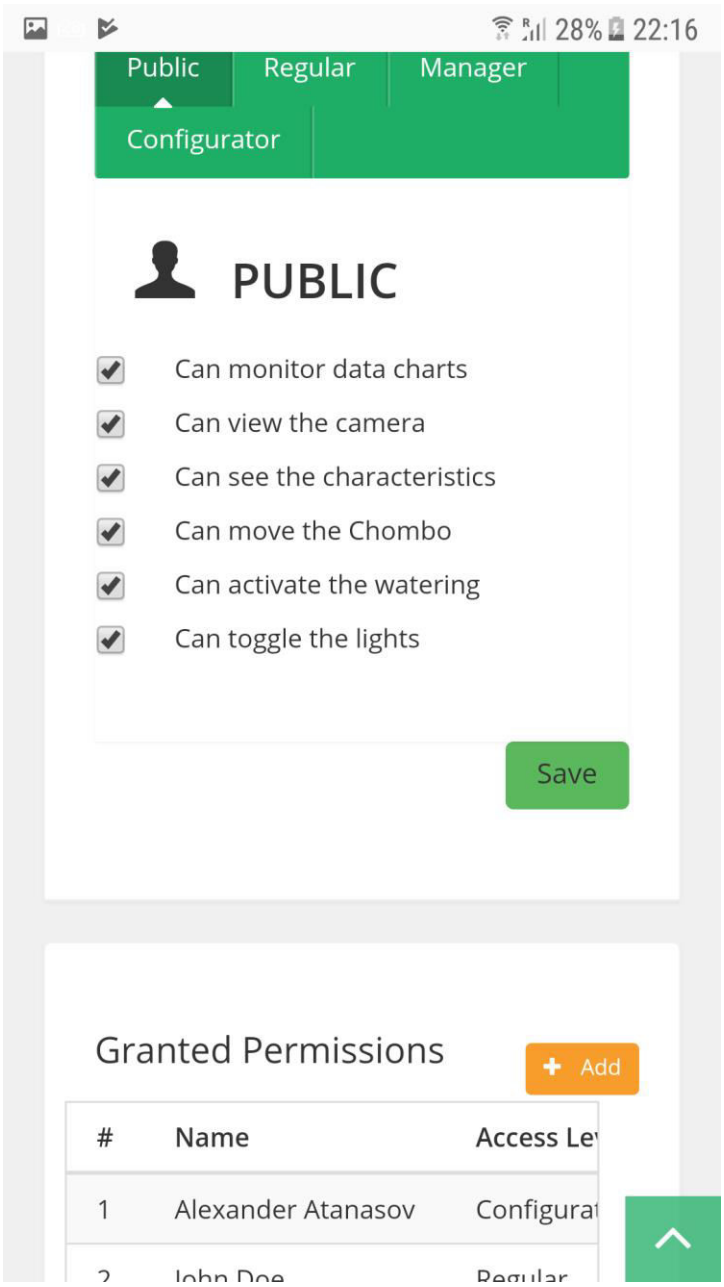


Illustration 40: Interface of the Application – Permissions Page

This illustration represents the chombo's permissions page. Here the user can manage the permissions for every type of user. This must be set for every chombo separately. The public permissions will be applied only if the chombo is set to be public, else it will ignore these permissions. The bottom part of this page contains a list of users that have special permissions applied, to them. The owner can give or take permissions to other user which does not depend on the user's access level for that chombo. This can be done in the page accessed from the "Add" button in the "Granted Permissions" section.

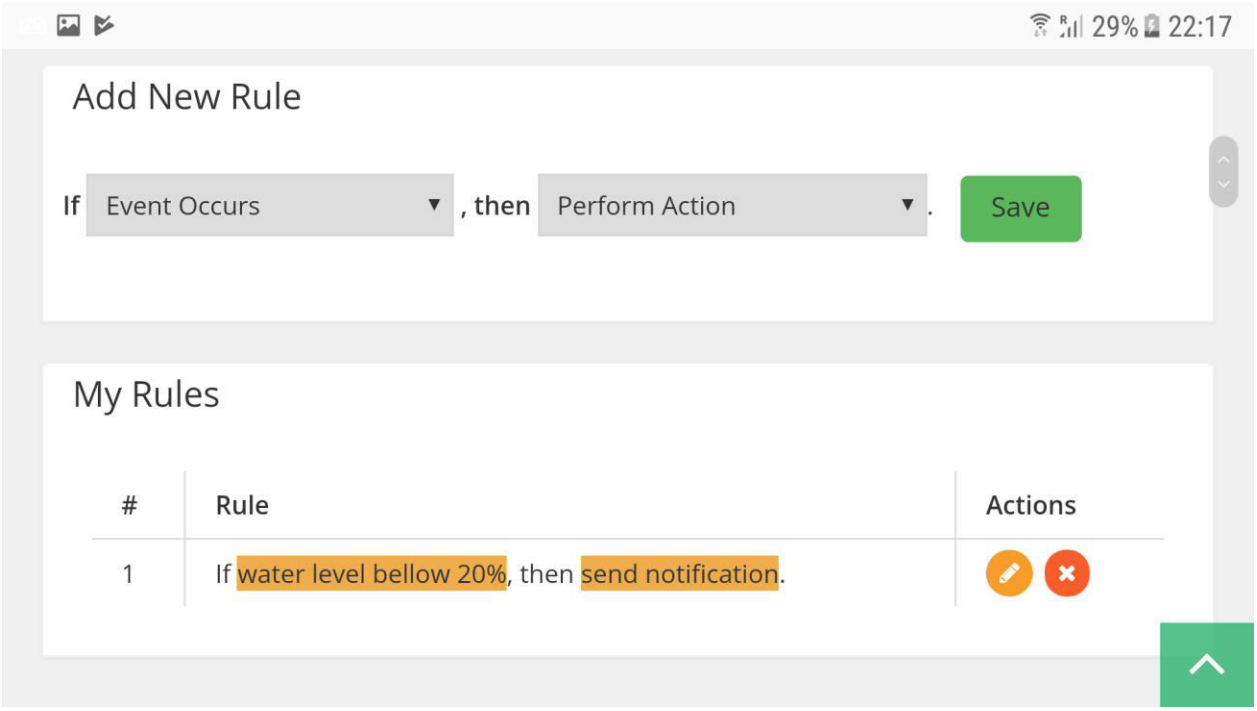


Illustration 41: Interface of the Application – Rules Page

This illustration shows the rules page of the currently edited chombo. The rules are created in the upper section of the page, by selecting an event and then selecting an action from the dropdowns, and pressing save. The rule then is shown in the list of rules for this chombo in the bottom section and can be edited or deleted.

4.4 Conclusions

In this section, it was review the whole design of the application. It has been opted for a 3-layer architecture that covers one of the project's objectives.

The overall appearance of the application has been designed. It aims to achieve an application with a user-friendly interface but complete at the same time and offering all the required functionality. It is an intuitive and functional design. With this design has managed to avoid a saturation of information and at the same time maintains an order in the distribution of information and convey to the user a sense of comfort. This partially covers the last goal of the project.

5. Implementation, implantation and evaluation

5.1 Introduction

This section will detail the way in which the functionalities described in the previous section have been implemented into the prototype. This describes the structure that has been followed to develop the entire application.

It also describes the implementation process by the user and then describes the evaluations that have been made to ensure the correct operation of all use cases.

5.2 Implementation

This section will explain the details of the implementation. It starts with the presentation layer because in this case it is the most important layer, since it is the one that is in direct contact with the user and must be done in detail.

5.2.1 Presentation Layer

In order to implement this layer, first must define which are the main pages of the mobile application and make a good design. The overall design of the pages is shown in the previous section above with detailed description for every page. The most important pages are the profile page, view chombo details and permissions pages, because the mobile application is orientated in the management of the chombos. The main purpose of the application is to give the user the opportunity to easily manage them from his phone.

Another key aspect of the overall design of the application is that is made to be responsive in order to run on different devices. This is done using the Vendroid templates and CSS with the latest Bootstrap library in addition to jQuery to make some of the components. Only some parts are yet to be developed responsively. It is important to note that as it is a prototype of a Mobile application the pages aren't fully functional.

5.2.2 Business/ Logical Layer

In this layer, the most important functionalities such as View Profile, Manage Chombos, Manage friends and others that are described above, are implemented using PHP. This will make it easier to implement this project, because of the great flexibility of the language. The application uses the MVC(Model-View-Control) architecture in a local environment.

The usage of a good IDE is also an important part of the development of this project. The Sublime Text 3 provides really good design and it's flexibility provides great personalization, so for that reason it has been chosen. Of course, there are many other IDEs that can provide the same usability and comfort, but this goes to personal preferences.

5.2.3 Data Layer

This layer is oriented to the saving of the information. In this case, most of the information is contained in the database on the server and retrieved from there. However, there is an essential element in the application that guarantees us to collect the correct information at all times.

Because this prototype is based on the MVC architecture, the interaction with the database is made with model classes in PHP, which partially resemble the way the Data layer of the Three-layer architecture is defined.

5.3 Integration

The process of integration of the application is very simple. You just need to install the apk file of the application and you are all done. In order to work with the application however, you must first make sure the web application is integrated on a server and it is fully functional.

5.4 Evaluation

Once the implementation is complete, and even during the implementation, all necessary evaluations are made on the application to ensure that it is working correctly.

Tests	Coverage	Result
Test1		If the user enters valid register details then he can register in the system.
Test2		If the user is logged he can view

		his or any profile page.
Test3		If the user is authorized he can view the product data.

Table 14: Tests

6.Conclusions

Finally, this section is aimed at showing the level of completion of the initial project, what difficulties have been encountered throughout the development and what decisions have been taken to resolve them. There is also a small introduction to possible improvements or additions that may appear in later years, since other students of the university have shown interest in improving and / or expand each of the parts that make up the system.

6.1 Level of completion

The main goal of the project is to create a prototype of a Mobile application for remote management of "chombos". This is fulfilled using PHP and templates for the views to create the overall application, and after that, using the Android Studio's libraries it was integrated into an Android application.
The functionalities that were desired has been all implemented, but not all are fully functional. The user, can login/register and access his profile from where he can manage his chombos. The rest of the functionalities are only visually implemented, but not functional, like the management of the chombo's permissions, which is the second objective of this project.
As for the design of the project, in the System Design the architecture is defined as three-layered, which also accomplishes one of the objectives. The implementation of the prototype however, doesn't use this architecture. Instead it uses the MVC(Model-View-Control).
The templates that were used for the implementation of the project are responsive and are made to be user-friendly and as simple as possible. This fulfills the final goal of the project.

6.2 Resolved difficulties

One of the biggest difficulties for the implementation of the project was the creation of a prototype of a mobile application. It was a difficult, because of the lack in knowledge in application development and in programming on different programming languages other than PHP. To solve this problem, I have

researched and found a way to integrate a website into an Android application, using the webview library of Android Studio. This way it was only needed to make the “website” behave like mobile application.

Other difficulty was making the views (templates) responsive, because of the short time that this project had to be developed. The solution was to find a premade responsive template and use it as base for the templates of the project. The templates used are from Vendroid, which provide all the needed components to carry out all the required functionalities like data chars, map, menus, tables and so on.

6.3 Future Implementations

After describing the totality of the project, future implementations are presented that can be carried out by other students interested in its continuity.

Firstly, the application can be made to be fully functional, by fully implementing it using Android Studio or with other tools, or programing languages.

Very important functionality that will be good to be added is the opportunity for the user to manage a group of chombos at the same time.

Other ideas would be to add a functionality that allows comparing the data charts of two chombos, or making a statistical analysis of several combos. Also creating custom access levels so that the owner can add it to multiple people without having to set the permissions for everyone separately.

7. References

VENDROID TEMPLATES

http://www.venmond.com/demo/vendroid/

CODEPEN

https://codepen.io/

CSS TRICKS

https://css-tricks.com/

PHP MANUAL

http://php.net/

CODEIGNITER FRAMEWORK

https://codeigniter.com

STACKOVERFLOW

https://stackoverflow.com

MICROSOFT DEVELOPER NETWORK

https://msdn.microsoft.com

LOGO MAKER

https://logomakr.com/