**Universidad de las Fuerzas Armadas**

**ESPE**

**Specification of Software**

**requirements**

**Project:** SimulatorHealth Cody + (2021)

**Members:**

Kenneth Andrade, Mateo Ávila,

Alisson Barriga, Rafael Buse.

**Revision history**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Review** | **Description** | **Author** |
| 20/01/2021 | 0.1 | Software requirements specification | Software Developer |
|  |  |  |  |
|  |  |  |  |

Document validated by the parties as of: [--/--/----]

|  |  |
| --- | --- |
| For the client | By the supplying company |
|  |  |
| Fdo. D./ | Fdo. D./ |

CONTENT

[**1 Introduction**](#_fyk4i4uoa8w7) **5**

[1.1 Purpose](#_9y2c6ke74q0a) 5

[1.2 Scope](#_55aex0jg1m5d) 5

[1.3 Personnel involved](#_erfq65njjkbi) 5

[1.4 Definitions, acronyms and abbreviations](#_ohvo1qnmqtk5) 6

[1.5 References](#_gl7nzr48d65p) 7

[1.6 Summary](#_adqimniu9n1y) 7

[**2. General description**](#_5kid5cdfykh7) **7**

[2.1 Product perspective](#_se2oulgsv8fp) 7

[2.2 Product functionality](#_nzr45t80rvqd) 8

[2.3 User characteristics](#_cmfnlhuo7p04) 8

[2.4 Restrictions](#_dpjzmkfuwk1h) 9

[2.5 Assumptions and dependencies](#_x5393c3kev1d) 9

[2.6 Predictable evolution of the system](#_e5pqafcx81r9) 9

[**3 Specific requirements**](#_ee79z6v63zhl) **9**

[3.1 Common interface requirements](#_tph73anhhj6l) 14

[3.1.1 Software interfaces](#_iudu7unmcymo) 14

[3.1.2 Communication interfaces](#_r0ch23cgfvsv) 14

[3.2 Functional requirements](#_6e9rvx6kxjz6) 14

[3.2.1 Functional requirement 1](#_o5nd8cku90xb) 14

[3.2.2 Functional requirement 2](#_ogybqcymggsj) 15

[3.2.3 Functional requirement 3](#_qdgxo6z6cmv4) 15

[3.2.4 Functional requirement 4](#_vy7tfbv0b3g9) 15

[3.2.5 Functional requirement 5](#_ufoobgy3ybck) 15

[3.2.6 Functional requirement 6](#_sye5cgf6w9ud) 16

[3.2.7 Functional requirement 7](#_zgxsw7wtapcd) 16

[3.2.8 Functional requirement 8](#_k802r2vr8fvp) 16

[3.2.9 Functional requirement 9](#_tjty1ril1dga) 16

[3.3 Non- Functional Requirements](#_8favcjnz7s36) 16

[3.3.1 Performance Requirements](#_m8h5kcx91a6y) 18

[3.3.2 Reliability](#_eweoa3bqgu25) 19

[3.3.3 Availability](#_hptyx4nkvdnj) 19

[3.3.4 Maintainability](#_1xqbgqrbx4ey) 19

[3.3.5 Portability](#_pjizpiucmmqn) 19

[3.4 Other requirements](#_1iyfe74uy6yf) 19

[**4 Use case diagram.**](#_fgmwq53zv9wp) **20**

[**5 Use case specification**](#_n3cref6sqoax) **20**

[5.1 Specification of use cases.](#_gxvtxrn0tngi) 20

[**6 Class diagram**](#_6x0hh6jh3bcq) **26**

[**7 Class diagram specification**](#_unizq7pnk419) **26**

[7.1 Specifying class attributes](#_gw85gwfyflb9) 26

[**8 User interfaces**](#_xw71sqw8g7o9) **29**

[8.1 Software Interfaces](#_z6bdygluo8r4) 29

[8.2 Hardware Interfaces](#_nt49wcew8mj5) 31

[**9 Appendices**](#_4rxjfp9rp6v) **32**

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# **1 Introduction**

System that connects people with their favorite testing laboratory, streamlining processes and avoiding possible infections.

## **1.1 Purpose**

Help people to have a prediction of a possible contagion, so that later they can connect with the laboratory of their choice to schedule a rapid test or pcr, to avoid the crowding of people in the same place. In addition, the administrator will have control of inventory, itinerary and results

## **1.2 Scope**

Our project is called "Health Cody +", a system that links the user with the test laboratory.

The user interface allows you to obtain a prediction of a possible contagion, schedule appointments, and obtain results after taking the test in person.

While the administrator (laboratory) can manage an inventory, control the schedule and manage patient results.

## **1.3 Personnel involved**

|  |  |
| --- | --- |
| **Name** | Kenneth Andrade |
| **Role** | Developer and reviewer |
| **Professional category** | Software engineering student |
| **Responsibilities** | Code review and help in documentation tasks |
| **Contact information** | koandrade@espe.edu.ec |
| **Approval** | Yes |

|  |  |
| --- | --- |
| **Name** | Mateo Ávila |
| **Role** | Developer and reviewer |
| **Professional category** | Software engineering student |
| **Responsibilities** | Code review and help in documentation tasks |
| **Contact information** | mdavila5@espe.edu.ec |
| **Approval** | Yes |

|  |  |
| --- | --- |
| **Name** | Alisson Barriga |
| **Role** | Developer and reviewer |
| **Professional category** | Software engineering student |
| **Responsibilities** | Code review and help in documentation tasks |
| **Contact information** | albarriga1@espe.edu.ec |
| **Approval** | Yes |

|  |  |
| --- | --- |
| **Name** | Rafael Buse |
| **Role** | Developer and reviewer |
| **Professional category** | Software engineering student |
| **Responsibilities** | Code review and help in documentation tasks |
| **Contact information** | rebuse@espe.edu.ec |
| **Approval** | Yes |

## 

## **1.4 Definitions, acronyms and abbreviations**

**User:** is a person who uses a computer or a network service.

**Administrator:** is the person who is in charge of carrying out the administrative task through planning, organization, direction and control

**Laboratory:** are those fixed or mobile places that have the technical, material and human capacity to carry out measurements, analysis or determine the characteristics of materials, products or equipment according to established specifications.

**PCR test:** is a diagnostic test that allows to detect a fragment of the genetic material of a pathogen.

**Quick test:** identify the presence of antibodies or antigens and show the result qualitatively, positive or negative.

## **1.5 References**

World Health Organization. Laboratory Testing Strategy Recommendations for COVID-19.; 2020

Abbott. Abbott launches molecular point-of-care test to detect novel coronavirus in as little as five minutes. Vol 564.2020. https://abbott.mediaroom.com/2020-03-27-Abbott-Launches-Molecular-Point-of-Care-Test-to-Detect-Novel-Coronavirus-in-as-Little-as-Five-Minutes.

Carbone M, Green JB, Bucci EM, Lednicky JA. Coronaviruses: Facts , Myths , andHypotheses. J Thorac Oncol. 2020.

## **1.6 Summary**

System called "Health Cody Plus" that connects people with their favorite test laboratory, where the user provides prediction of COVID-19 contagion, scheduling of appointments and visualization of results, while the Administrator allows to manage the inventory, control the itinerary and manage PCR and rapid test results.

# **2. General description**

The system mainly seeks to help people to obtain an accurate prognosis of a possible contagion of covid 19 and later to book an appointment in the laboratory to do the pcr or rapid test. The system will carry out activities such as taking surveys and giving its results obtaining a diagnosis for each patient and also the possibility of scheduling appointments for the laboratory of your choice.

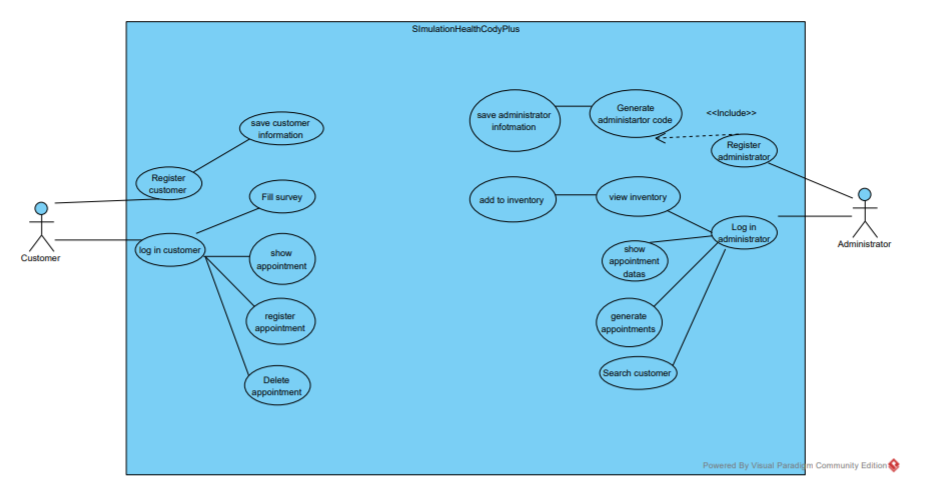
## **2.1 Product perspective**

The system is being developed by the entire work team, it is a totally independent system, that is to say, it does not depend on other systems, much less larger systems. This system seeks that users can feel good through its interface, conducting easy and understandable surveys for the user and giving an accurate diagnosis to the patient.

## **2.2 Product functionality**

The system will have a simple console interface to be used and, in conclusion, it can be understood at a glance by all users who will use it in the future.

**USE CASE DIAGRAM**



## **2.3 User characteristics**

|  |  |
| --- | --- |
| **Type of user** | Customer |
| **Training** | Basic education |
| **Abilities** | Ease of handling Windows Operating System. |
| **Activities** | Register in the system, Login, Fill out the survey, Schedule appointment. |

|  |  |
| --- | --- |
| **Type of user** | Administrator |
| **Training** | Higher education |
| **Abilities** | Ease of handling Windows Operating System. |
| **Activities** | Register in the system, Login with a code, Administrate the inventory, Administrate the appointments. |

## 

## **2.4 Restrictions**

• The system must be developed in Java programming language.

• Information about users must be stored.

• The interface should be user friendly and easy to use.

• The system must be suitable for users over 16 years old.

• The system can only be used with a Windows Operating System

## **2.5 Assumptions and dependencies**

• The program that will be delivered will mainly work in the console and then it will be transferred to the graphical interface.

• As soon as you choose to use a database, you must modify the system connection.

## **2.6 Predictable evolution of the system**

After a while, the system will be modified in graphical mode for the greater benefit of the users who use it, as well as transferring their data to a more sophisticated database.

# **3 Specific requirements**

RF 1 Register users

RF 1.1 Register (customer)  
The system will allow the user (customer) to register requesting personal data

|  |  |
| --- | --- |
| Requirement number | 1.1 |
| Requirement name | Register customer |
| Type | Requirement Restriction |
| Requirement source | User (customer) |
| Priority of requirement | High/Essential Medium/Desired Low/Optional |

RF 1.2 Register (Administrator)  
The administrator must have a special code that identifies him as a system  
administrator.

|  |  |
| --- | --- |
| Requirement number | 1.2 |
| Requirement name | Register administrator |
| Type | Requirement Restriction |
| Requirement source | User (administrator) |
| Priority of requirement | High/Essential Medium/Desired Low/Optional |

RF 2 log in users

RF 2.1 log in (customer)  
Once registered, the user will be able to log in and access the other features of the system

|  |  |
| --- | --- |
| Requirement number | 2.1 |
| Requirement name | log in (customer) |
| Type | Requirement Restriction |
| Requirement source | User (customer) |
| Priority of requirement | High/Essential Medium/Desired Low/Optional |

RF 2.2 log in (administrator)   
Once registered, the administrator will be able to log in and access the other system functions as an administrator.

|  |  |
| --- | --- |
| Requirement number | 2.2 |
| Requirement name | log in (Administrator) |
| Type | Requirement Restriction |
| Requirement source | User (administrator) |
| Priority of requirement | High/Essential Medium/Desired Low/Optional |

RF 3 Fill survey (customer)

The client must fill out a survey with questions related to covid-19

|  |  |
| --- | --- |
| Requirement number | 3 |
| Requirement name | Fill survey |
| Type | Requirement Restriction |
| Requirement source | User (customer) |
| Priority of requirement | High/Essential Medium/Desired Low/Optional |

RF 4 Schedule Appointment (customer)

RF 4.1 see available appointment times (customer)  
the user will be able to see the schedules that are available to be able to schedule their appointment

|  |  |
| --- | --- |
| Requirement number | 4.1 |
| Requirement name | See available appointment times (customer) |
| Type | Requirement Restriction |
| Requirement source | User (customer) |
| Priority of requirement | High/Essential Medium/Desired Low/Optional |

RF 4.2 view scheduled appointments (customer )  
The client will be able to see the appointments they have scheduled and decide if they can cancel them

|  |  |
| --- | --- |
| Requirement number | 4.2 |
| Requirement name | view scheduled appointments (customer ) |
| Type | Requirement Restriction |
| Requirement source | User (customer) |
| Priority of requirement | High/Essential Medium/Desired Low/Optional |

RF 5 Check result (customer)  
The results of the tests to be performed will reach the user in this section

|  |  |
| --- | --- |
| Requirement number | 5 |
| Requirement name | Check result (customer) |
| Type | Requirement Restriction |
| Requirement source | User (customer) |
| Priority of requirement | High/Essential Medium/Desired Low/Optional |

RF 6 view medical history (administrator)  
The administrator will be able to see the client's medical history

|  |  |
| --- | --- |
| Requirement number | 6 |
| Requirement name | view medical history (administrator) |
| Type | Requirement Restriction |
| Requirement source | User (administrator) |
| Priority of requirement | High/Essential Medium/Desired Low/Optional |

RF 7 View scheduled appointments (administrator)  
The administrator will be able to see the appointments scheduled by the client users

|  |  |
| --- | --- |
| Requirement number | 7 |
| Requirement name | View scheduled appointments (administrator) |
| Type | Requirement Restriction |
| Requirement source | User (administrator) |
| Priority of requirement | High/Essential Medium/Desired Low/Optional |

RF 8 Manage inventory

RF 8.1 view available inventory (administrator)  
The administrator will be able to see in the inventory of the products

|  |  |
| --- | --- |
| Requirement number | 8.1 |
| Requirement name | View inventory (administrator) |
| Type | Requirement Restriction |
| Requirement source | User (administrator) |
| Priority of requirement | High/Essential Medium/Desired Low/Optional |

RF 8.2 Manipulate inventory (administrator)  
The administrator can add delete and modify products in the inventory

|  |  |
| --- | --- |
| Requirement number | 8.2 |
| Requirement name | Manipulate inventory (administrator) |
| Type | Requirement Restriction |
| Requirement source | User (administrator) |
| Priority of requirement | High/Essential Medium/Desired Low/Optional |

RF 9 Generate reports ( administrator)  
The administrator will be able to generate inventory reports and appointments

|  |  |
| --- | --- |
| Requirement number | 9 |
| Requirement name | Generate reports ( administrator) |
| Type | Requirement Restriction |
| Requirement source | User (administrator) |
| Priority of requirement | High/Essential Medium/Desired Low/Optional |

## **3.1 Common interface requirements**

### **3.1.1 Software interfaces**

The system uses the system console, preferably windows or also other operating systems (Linux, MC OS)

### **3.1.2 Communication interfaces**

In order to use the program, the user must have the JAR and the JRE installed

## **3.2 Functional requirements**

### **3.2.1 Functional requirement 1**

* Register users: The system will allow the user to register (administrator, customer)
* Register administrator: The system will allow you to register as an administrator with an identification code provided by the company and also personal data.
* Register client: The system will allow registering client requesting their personal data

### **3.2.2 Functional requirement 2**

* Log in users : Users must identify themselves in order to access the other functions
* Log in Administrator: Administrators must identify themselves in the administrator section to access the functions that correspond to them
* Log in customers: Customers must login with a username and password to access the other functions

### 

### **3.2.3 Functional requirement 3**

* Fill survey : Customers must fill out a survey with questions related to covid 19 and according to that a report will be generated with their health status

### **3.2.4 Functional requirement 4**

* Schedule Appointment :Clients will be able to schedule an appointment for a test
* See available appointment times: Users will have available the hours in which the tests will be carried out and thus be able to reserve a shift
* view scheduled appointments: Customers when scheduling an appointment will be able to see the detailed information of the appointment and decide to cancel it

### **3.2.5 Functional requirement 5**

* Check result :In this section the client will be able to see the results of the tests that have been carried out

### **3.2.6 Functional requirement 6**

* view medical history :The administrator can search the medical history of a client

### **3.2.7 Functional requirement 7**

* View scheduled appointments: The administrator will be able to see the appointments that are scheduled in a day or week

### **3.2.8 Functional requirement 8**

* Manage inventory :The administrator will be in charge of managing the inventory
* view available inventory: The Administrator will be able to see the stock of the products
* Manipulate inventory: The Administrator will be able to perform the actions such as add, delete and modify the inventory

### **3.2.9 Functional requirement 9**

* Generate reports :The administrator will be able to generate medical inventory reports and appointments

## **3.3 Non- Functional Requirements**

RNF 01 Execution\_Time

The execution\_time is necessary to know if the software is running good.

|  |  |
| --- | --- |
| Identification of the requirement. | RNF 01 |
| Requirement Name. | Execution time |
| Features | The user executes the application. |
| Requirement Description | After the user executes the application, the time that takes the application to execute. |
| Priority requirement | High |

RNF 02 Survey\_Percentege (Customer)

The percentage of the survey is necessary to know if the user needs or not a scheduled appointment.

|  |  |
| --- | --- |
| Identification of the requirement. | RNF 02 |
| Requirement Name. | Survey percentage |
| Features | The user obtains a percentage after that he fills the survey. |
| Requirement Description | The percentage of the survey is necessary to know if the user needs a schedule appointment. |
| Priority requirement | High |

RNF 03 Exceptions

The exception is an important part of the software, because if the exception doesn't work correctly, the software falls down.

|  |  |
| --- | --- |
| Identification of the requirement. | RNF 03 |
| Requirement Name. | Exceptions |
| Features | The user such as the administrator or the patient does something incorrectly in the software. |
| Requirement Description | The software needs to make an exception and continue with his flow. |
| Priority requirement | High |

RNF 04 User\_documentation

The software needs to contain user documentation because if the user doesn't know what they need to do, the user documentation is to be a help for this.

|  |  |
| --- | --- |
| Identification of the requirement. | RNF 04 |
| Requirement Name. | User documentation |
| Features | The user needs documentation to use the application. |
| Requirement Description | The application is going to do everything that is written in the user documentation. |
| Priority requirement | High |

RNF 05 Save information

The software needs to save information such as inventory and users.

|  |  |
| --- | --- |
| Identification of the requirement. | RNF 05 |
| Requirement Name. | Save information |
| Features | The software needs to save information in files. |
| Requirement Description | The application is going to save information about inventory, users, appointments. |
| Priority requirement | High |

### **3.3.1 Performance Requirements**

The system will be able to save user data, inventory and medical appointments in files, therefore no information will be lost when running the application or when it is closed.

### **3.3.2 Reliability**

The system will have an exhaustive validation, where in case the user makes mistakes the system will allow him to continue after giving a notification message to the user.

### **3.3.3 Availability**

The availability will be around 800% because it requiere many requirements, and it’s structure is very complex, so it must always be available to the user.

### **3.3.4 Maintainability**

The preventive maintenance will be carried every two weeks for a period of three months, the developers will carry out a check of the application, verifying the correct use of the exceptions, and the normal operation of the system, they will also add functionalities that help the system to the function better and to better manipulate it by the user.

### **3.3.5 Portability**

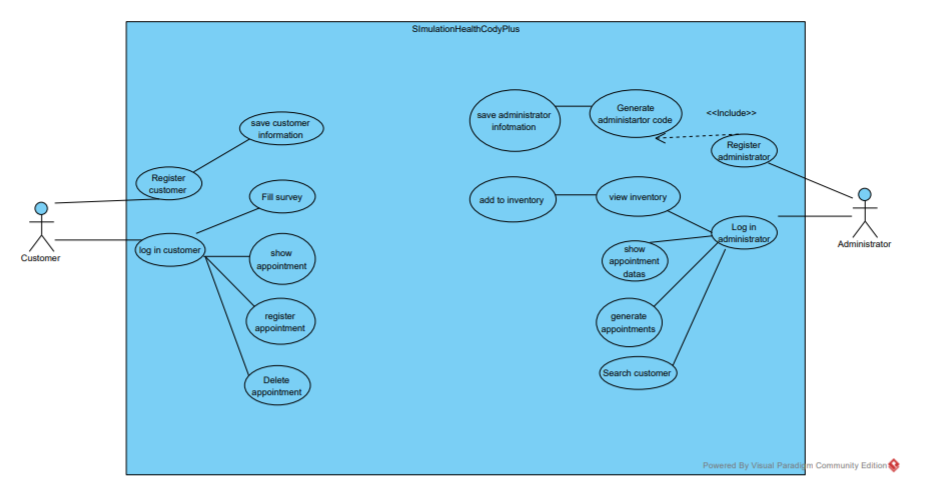
The application will be available for the operating system of Windows or with other operating system such as Linux, Mac that have the JRE(Java Runtime Environment)

this has to be installed so that the developed software can work.

## **3.4 Other requirements**

The changes that need to be done in the project are going to be in different versions understood in different classes.

# **4 Use case diagram**.



# 5 **Use case specification**

## 5.1 Specification of use cases.

|  |  |
| --- | --- |
| Identifier | Use Case 1 |
| Name | Register administrator |
| Description | Register administrators. |
| Actors | Administrator |
| Preconditions | Nothing. |
| Postconditions | The system will save the administrators information. |

|  |
| --- |
| Normal flow of events |
| The system can register an administrator in a json File. |
| Alternative flow of events |
| The system can not register an administrator and show an exception. |

|  |  |
| --- | --- |
| Identifier | Use Case 2 |
| Name | Generate administrator code |
| Description | The system creates an administrator code. |
| Actors | Administrator |
| Preconditions | The administrator needs to be registered. |
| Postconditions | The administrator can access the code. |

|  |
| --- |
| Normal flow of events |
| The system creates an administrator code. |
| Alternative flow of events |
| The system can not access it with a different code. |

|  |  |
| --- | --- |
| Identifier | Use Case 3 |
| Name | Save administrator information |
| Description | Save all the data of the administrator. |
| Actors | Administrator. |
| Preconditions | The administrator needs to be registered. |
| Postconditions | The administrator’s data appeared in a file. |

|  |
| --- |
| Normal flow of events |
| The system saves the data of the administrator in a file. |
| Alternative flow of events |
| The system can’t save the data because the administrator can’t complete the data. |

|  |  |
| --- | --- |
| Identifier | Use Case 4 |
| Name | Login administrator |
| Description | Login an administrator account. |
| Actors | Administrator |
| Preconditions | The administrator needs to be registered and have a code. |
| Postconditions | The system recovered the administrators data and comprobate this. |

|  |
| --- |
| Normal flow of events |
| The system can access the administrator registers and comprobate the administrator. |
| Alternative flow of events |
| The system cannot comprobate anything because the file is empty |

|  |  |
| --- | --- |
| Identifier | Use Case 5 |
| Name | View inventory |
| Description | The administrator can administer the inventory. |
| Actors | Administrator. |
| Preconditions | The administrator need access to the system. |
| Postconditions | The administrator can view inventory |

|  |
| --- |
| Normal flow of events |
| The system can show the inventory for the administrator. |
| Alternative flow of events |
| The system cannot show inventory because the file is not created. |

|  |  |
| --- | --- |
| Identifier | Use Case 6 |
| Name | Add to inventory |
| Description | The administrator can add things to the inventory. |
| Actors | Access to the inventory. |
| Preconditions | The administrator needs to be registered. |
| Postconditions | The administrator can |

|  |
| --- |
| Normal flow of events |
| The inventory is created by the administrator. |
| Alternative flow of events |
| The inventory can’t be registered for the file. |

|  |  |
| --- | --- |
| Identifier | Use Case 7 |
| Name | Show appointments datas |
| Description | The administrator can view the appointments datas. |
| Actors | Administrator. |
| Preconditions | The administrator needs to be registered. |
| Postconditions | The administrator can view the appointments datas. |

|  |
| --- |
| Normal flow of events |
| The administrator can view the appointments datas. |
| Alternative flow of events |
| The administrator can’t view the appointments datas because don’t exist the file. |

|  |  |
| --- | --- |
| Identifier | Use Case 8 |
| Name | Generate appointments |
| Description | The administrator can create appointments for the week. |
| Actors | Administrator |
| Preconditions | The administrator needs to be registered. |
| Postconditions | The administrator can create appointments for the week for the customer |

|  |
| --- |
| Normal flow of events |
| The administrator can create appointments for each week and it is saved in a file. |
| Alternative flow of events |
| The administrator cannot access because the file doesn't exist. |

|  |  |
| --- | --- |
| Identifier | Use Case 9 |
| Name | Search customer |
| Description | The administrator can search a customer. |
| Actors | Administrator |
| Preconditions | The administrator needs to be registered. |
| Postconditions | The administrator can search a customer. |

|  |
| --- |
| Normal flow of events |
| The administrator can search a customer. |
| Alternative flow of events |
| The administrator can’t search a customer because the file doesn't exist. |

## 

|  |  |
| --- | --- |
| Identifier | Use Case 10 |
| Name | Register customer |
| Description | Register customers |
| Actors | Customer |
| Preconditions | Nothing |
| Postconditions | The system will save the customers information. |

|  |
| --- |
| Normal flow of events |
| The system saves the customers information in a file. |
| Alternative flow of events |
| The system can not register a customer and show an exception. |

|  |  |
| --- | --- |
| Identifier | Use Case 11 |
| Name | Save Customer Information |
| Description | The system saves the customer information in a json file. |
| Actors | Customer. |
| Preconditions | The customer needs to register an account.. |
| Postconditions | The customer is saved in a file. |

|  |
| --- |
| Normal flow of events |
| The system save the customer information in a json file. |
| Alternative flow of events |
| The system can’t create a file with the customer. |

|  |  |
| --- | --- |
| Identifier | Use Case 12 |
| Name | Login customer |
| Description | Login a customer account. |
| Actors | Customer |
| Preconditions | The customer needs a user and a password to access. |
| Postconditions | The system recovered the customers data and comprobate this. |

|  |
| --- |
| Normal flow of events |
| The system can access the customer registers and comprobate the customer. |
| Alternative flow of events |
| The system cannot comprobate anything because the file is empty. |

|  |  |
| --- | --- |
| Identifier | Use Case 13 |
| Name | Fill Survey |
| Description | The system shows a survey that needs to complete the customer. |
| Actors | Customer. |
| Preconditions | The customer needs to be registered and enter for the first time to the system. |
| Postconditions | The other occasions that the customer enter to the system, the survey don´t appeared again. |

|  |
| --- |
| Normal flow of events |
| The survey only appeared to be the first that the user accessed the system. |
| Alternative flow of events |
| The survey was not filled and the other occasion appeared again. |

|  |  |
| --- | --- |
| Identifier | Use Case 14 |
| Name | Show appointment. |
| Description | The system shows the appointments disponibles to the customer. |
| Actors | Customer. |
| Preconditions | The customer needs to be registered. |
| Postconditions | The customer shows each one of the appointments that are disponibles in the system. |

|  |
| --- |
| Normal flow of events |
| The system shows the appointments disponibles to the customer |
| Alternative flow of events |
| The system can’t show anything because every date is ocupated. |

delete appointment

|  |  |
| --- | --- |
| Identifier | Use Case 15 |
| Name | Register appointment |
| Description | The system registers the appointment of the system. |
| Actors | Customer. |
| Preconditions | The customer only can save one date in the system. |
| Postconditions | The customer registers an appointment for the week. |

|  |
| --- |
| Normal flow of events |
| The customer registers an appointment for the week. |
| Alternative flow of events |
| The customer can’t register an account because the appointments are ocupated. |

|  |  |
| --- | --- |
| Identifier | Use Case 16 |
| Name | Save appointments |
| Description | Delete appointment |
| Actors | Customer |
| Preconditions | The customer can delete appointments. |
| Postconditions | The customer can save another appointment because save the last appointment. |

|  |
| --- |
| Normal flow of events |
| The customer can delete the appointment. |
| Alternative flow of events |
| The customer can’t delete the appointment because don’t have appointments done. |

# 

# **6 Class diagram**

# 

# **7 Class diagram specification**

The diagram is based in the model system package that creates the operations that are going to do our project.

## 7.1 Specifying class attributes

1. **Person**

**1.1) Attributes:**

* namePerson: String
* idPerson: String
* genderPerson: String
* agePerson: Integer

**1.2) Methods**

* Person(all)
* showDataPerson():String

1. **Administrator**

**2.1) Attributes:**

* administratorUser: String
* administratorPassword: String
* administratorCode: String
* namePerson: String
* idPerson: String
* genderPerson: String
* agePerson: Integer

**2.2) Methods**

* showDataAdministrator():String
* Administrator(all)

1. **Customer**

**3.1) Attributes:**

* customerUser: String
* customerPassword: String
* namePerson: String
* idPerson: String
* genderPerson: String
* agePerson: Integer

**3.2) Methods**

* showDataCustomer():String
* Customer(all)

1. **Registry**

**4.1) Attributes:**

* administrator: Admministrator
* customer: Customer

**4.2) Methods**

* Registry(administrator: Administrator)
* Registry(customer: Customer)
* registerAdministrator()
* registerCustomer()
* generateAdminCode()

1. **User**

**5.1) Attributes:**

* user: String
* password: String
* accessCode: String

**5.2) Methods**

* loginCustomer():boolean
* loginAdministrator():boolean
* checkFirstTime():boolean

1. **DateAppointment**

**6.1) Attributes:**

* day: Integer
* month: Integer
* year: Integer
* hour: Integer
* minutes: Integer
* seconds: Integer
* code: Integer

**6.2) Methods**

* toString():String

1. **Appointment**

**7.1) Attributes:**

* weekend: ArrayList<ArrayList<DataAppointment>>
* appointments: ArrayList<Appointments>

**7.2) Methods**

* generateAppointments()
* registerAppointments()
* showAppointment(nameFile String)
* saveCustomerAppointment(date int,user String)
* showAppointmentCustomer(user String)
* showAppointment Dates(nameFile String,dataToString String)

1. **Inventory**

**8.1) Attributes:**

* products: ArrayList<Product>

**8.2) Methods**

* saveProduct(product: Product)
* findProduct(dataToFind: String): String
* modifyProduct(dataToFind: String,dataToUpdate: String):boolean
* deleteProduct(dataToDelete: String): boolean
* showInventory():ArrayList<Product>

1. **Product**

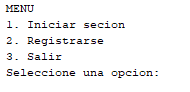
**7.1) Attributes:**

* idProduct: String
* codeProduct: String
* nameProduct: String
* quantityProduct: Integer

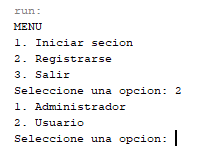
# **8 User interfaces**

## 8.1 Software Interfaces

**MAIN MENU**



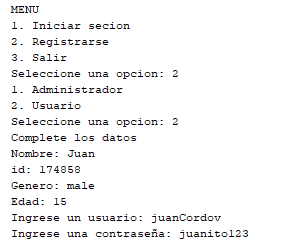
**MENU FOR REGISTER AN ACCOUNT**



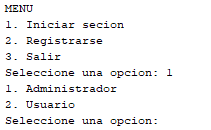
**SELECT ADMINISTRATOR**

# 

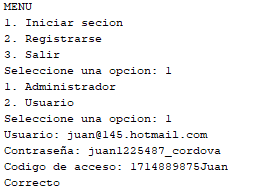
**SELECT CUSTOMER**



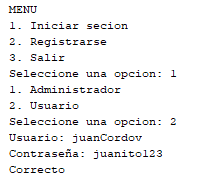
**MENU FOR THE LOGIN**

****

**MENU FOR THE LOGIN ADMINISTRATOR**

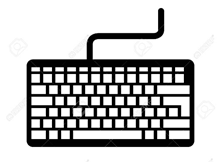
****

**MENU FOR THE LOGIN CUSTOMER**

****

## 8.2 Hardware Interfaces





The hardware interface is the most important tool for a programmer. The code is created from the display and the keyboard.

# **9 Appendices**

The project and the documentation was done by software engineering students.

As the project is in constant evolution, the documentation is going to change for every part of the project.