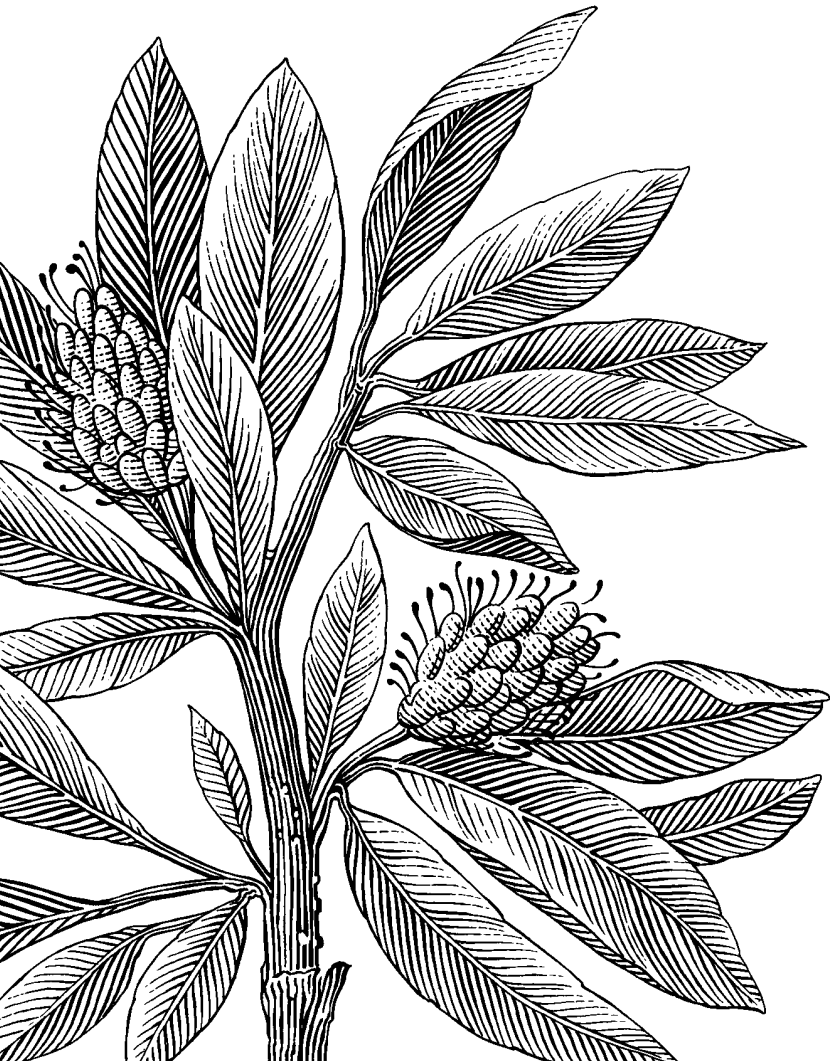
2DV609

WeCare
Requirements Specification



Author1: Mohammadali Rashidfarokhi
Author2: Khalil Mardini
Author3: Nima Safavi
Author4: Amirhossein Soltaninejad
Author5: Dominik Pappe

Contents

1. Introduction

To provide a more accurate and rapid diagnosis of Haemophilus paresis, a polymerase chain reaction (PCR) test was developed [1]. It is becoming increasingly difficult for laboratories to keep up with the fast pace of demand for the PCR test (e.g., Covid 19) because of the increase in demand.
Moving on, most laboratories are not using state-of-the-art technology for documenting the results. That is, prior to COVID-19, laboratories used papers and local archives, which meant patients preferred to take their tests at other clinics to be able to complete the tests faster and to be able to do them online. As a result, the demand for an application that assists patients and the laboratories with PCR and COVID-19 diagnosis is on the rise.

As a result, this project will result in the development of an application called "WeCare" and the purpose of developing this application is to help laboratories and individuals perform COVID-19 tests. Although the focus of the application is not only on Covid, other types of tests have also been enabled in the "Wecare" application. Those tests include "HIV", "pregnancy", and "Chlamydia".

Lastly, the document will cover system-wide requirements, System Interfaces, business rules, system constraints, and use-cases in detail.

2. System-wide Requirements

- Patients:

Patients are our main target group on the customer side for using our application. Because the pandemic affects everybody, we have a user base with different experiences and mixed ages.

- Doctor:

In addition to the patients, the doctors at the laboratory will also use the WeCare app, for inserting the test result of the patients into the application system.

- Laboratory Administrator:

Medical administration supervises the activities of doctors and patients. It develops and maintains policies and assists in the development of protocols for medical treatments, assessment of quality, and care for patients.

- Software Developer:

Developing software that fulfills the user's needs requires a developer to examine what they want, then design, test, and manufacture it.

- Database Developer:

The database developer is responsible for creating the database, as well as modifying existing databases to accommodate changes in the platform or the needs of the customers.

- Government:

The government is interested in ensuring that only certified companies issue official tests.

- Public & media

The press likes to talk about negative incidents. Especially in the area of health, government regulations should therefore be observed.

- Shareholder

Shareholders have an interest in ensuring that the application is functional and serves its purpose by freeing up capacity through minimizing bureaucratic administration and also reaching more customers by allowing them to order their test kit online.

Functional requirements:

SEC: Security
PAT: Patient
DOC: Doctor
ADM: Admin

- **SEC-01:** Users (patients) must be able to create an account for themselves so that they can reuse their account for each test. The reason for stating must here is that the user does not have the accessibility to register for the test since it is a requirement to be registered in the system to use the system's function.

- **SEC-02:** Patients must be able to authenticate themselves so that they can log in to their user account to see the next steps until they get their test results.

- **SEC-03:** Doctors must be able to authenticate themselves so that no unauthorized person gets access to the sensitive data of the patients.

- **SEC-04:** Admins must be able to authenticate themselves so that no unauthorized person gets access to privileged settings of the system.

- **PAT-01:** Patients must be able to make an appointment for their desired test type (PCR, HIV, Pregnancy, and Chlamydia) at a specific time in the preferred laboratory.

- **PAT-02:** Patients must be able to access their test results as a PDF-file with a QR-Code so that they can use that document as proof regardless of the results.

- **DOC-01:** The doctor must have access to information about the performed test such as patient ReportId number (patient case number) and the patient of the test, so he knows which case belongs to which test.

- **DOC-02:** The doctor must be able to insert the test result (positive or negative) into the system after they evaluated the test (e.g., PCR, HIV, Pregnancy and Chlamydia), so they don't need to fill out a physical document.

- **DOC-03:** Doctors should be able to have access on displaying the patient pdf test report, after inserting the result into the document.

- **ADM-01:** Admin have the accessibility to create new users within the system, because new admin and doctor accounts can't be created by the user.

- **ADM-02:** Admin have accessibility to assign roles to any registered users such as (patients and doctors).

- **ADM-03:** Admin has the accessibility to delete the user data from the system, to keep the data in the system up to date.

Non-functional requirements (NFR):

- **NFR-01 - Performance:** The home page should load in less than 3 seconds.
- **NFR-02 - Availability:** The system should work 99% of the time so the users can access it at any time.
- **NFR-03 - Usability:** The user interface should be easy for users to work with.
- **NFR-04 - Portability:** The system should be executable on any web browser.
- **NFR-05 - Security:** The system should consider the security measures by using SSL certificate.

Requirements properties for a faceted approach:

	Communication	**Database (changes)**	**System**	**User-interface**	**Security**
SEC-01	No	Yes	Yes	Yes	Yes
SEC-02	No	No	Yes	Yes	Yes
SEC-03	No	No	Yes	Yes	Yes
SEC-04	No	No	Yes	Yes	Yes
PAT-01	Yes	Yes	Yes	Yes	No
PAT-02	No	No	Yes	Yes	No
DOC-01	No	No	Yes	Yes	No
DOC-02	Yes	Yes	Yes	Yes	No
DOC-03	No	No	Yes	Yes	No
ADM-01	No	Yes	Yes	Yes	No
ADM-02	No	Yes	Yes	Yes	No
ADM-03	No	Yes	Yes	Yes	No

Requirements properties for a check-list based analysis:

A) Requirements testability

B) Premature design
C) Requirements realism
D) Requirements ambiguity
E) Combined requirements
F) Use of non-standard hardware
G) Conformance with business goals
H) Unnecessary Requirements

	A)	B)	C)	D)	E)	F)	G)	H)
SEC-01	Yes	No	Yes	No	No	No	Yes	No
SEC-02	Yes	No	Yes	No	No	No	Yes	No
SEC-03	Yes	No	Yes	No	No	No	Yes	No
SEC-04	Yes	No	Yes	No	No	No	Yes	No
PAT-01	Yes	No	Yes	No	No	No	Yes	No
PAT-02	Yes	No	Yes	No	No	No	Yes	No
DOC-01	Yes	No	Yes	No	No	No	Yes	No
DOC-02	Yes	No	Yes	No	No	No	Yes	No
DOC-03	Yes	No	Yes	No	No	No	Yes	No
ADM-01	Yes	No	Yes	No	No	No	Yes	No
ADM-02	Yes	No	Yes	No	No	No	Yes	No
ADM-03	Yes	No	Yes	No	No	No	Yes	No

Requirements properties for a check-list based risk analysis:

A) Performance risks
B) Safety and security risks
C) Process risks
D) Implementation technology risks
E) Database risks
F) Schedule risks
G) External risks
H) Stability risks

No risk = -
Low = 1
Medium = 2
High = 3

	A)	B)	C)	D)	E)	F)	G)	H)

SEC-01	1	-	1	-	-	-	-	-
SEC-02	1	B 2	1	-	-	-	-	-
SEC-03	-	2	1	-	-	-	-	-
SEC-04	-	2	1	-	-	-	-	-
PAT-01	1	-	-	-	-	1	-	-
PAT-02	-	-	-	-	2	1	-	-
DOC-01	-	-	-	-	-	-	-	-
DOC-02	-	-	-	-	-	-	-	-
DOC-03	-	-	-	-	2	1	-	-
ADM-01	-	2	-	-	-	-	-	-
ADM-02	-	2	-	-	-	-	-	-
ADM-03	-	2	-	-	-	-	-	-

Explanation of analyzed risks:

- **SEC-01**
  - A Too many people could create a new user account at the same time, which could lead to an overload of our system.
  - C Security standards should be set e.g., password requirements
- **SEC-02**
  - A Too many people could login to our system at the same time, which could lead to an overload of our system
  - B This could lead to a security gap, as login pages are often used by hackers.
  - C Security standards should be set e.g., password requirements
- **SEC-03**
  - B This could lead to a security gap, as login pages are often used by hackers. Since the role of a "Doctor" has access to personal user data, this means a greater risk.
  - C Security standards should be set e.g., password requirements
- **SEC-04**
  - B This could lead to a security gap, as login pages are often used by hackers. Since the role of "admin" has access to personal user data and can make possible changes in the system, this means a greater risk.
  - C Security standards should be set e.g., password requirements
- **PAT-01**
  - A Too many people who are making a booking at the same time could lead to an overload of our system
  - F Booking is one of the most important requirements and should therefore function smoothly. There could be delays as possibly the implementation is dragging on.
- **PAT-02**
  - E Saving the PDFs could lead to difficulties

F The automatic generation of the test certificates as a PDF file with QR code could turn out to be more difficult than planned.

- **DOC-03**
  - E Saving the PDFs could lead to difficulties
  - F The automatic generation of the test certificates as a PDF file with QR code could turn out to be more difficult than planned.
- **ADM-01**
  - B This could lead to a security gap, as login pages are often used by hackers. Since the role of "admin" has access to personal user data and can make possible changes in the system, this means a greater risk.
- **ADM-02**
  - B This could lead to a security gap, as login pages are often used by hackers. Since the role of "admin" has access to personal user data and can make possible changes in the system, this means a greater risk.
- **ADM-03**
  - B This could lead to a security gap, as login pages are often used by hackers. Since the role of "admin" has access to personal user data and can make possible changes in the system, this means a greater risk.

Requirements checklist:

Are the requirements complete?	Yes
Are the requirements consistent?	Yes
Are the requirements comprehensible?	Yes
Are the requirements ambiguous?	No
Is the requirements document structured?	Yes
Are the requirements traceable?	Yes
Does the requirements document as a whole fit together?	Yes

Test Cases for all requirements:

Test Case ID	TC-01
Requirement ID	SEC-01

Test description	The test case is to verify the creation of an account with test data on our platform
Pre-condition	Open the website of the WeCare application
Input	• E-Mail: kalle.svensson@gmail.com • Name: Kalle Svensson • Username: KalleSvensson • Password: TestCase123!
Test steps	1. Navigate to the registration page 2. Enter email-Adress 3. Enter whole name 4. Enter username 5. Choose a valid password. 6. Repeat the password 7. Agree terms & conditions 8. Press on the signup button
Expected output	1. Created user account 2. The members' information will be stored in the database.
Post-condition	-
Actual Result	1. Created user account 2. The members' information will be stored in the database.
Status (Pass / Fail)	Pass
Test Executed by	Nima Safavi
Test Execution Date	26.05.2022

Test Case ID	TC-02
Requirement ID	SEC-02
Test description	The test case is to verify the sign in of a user to our platform
Pre-condition	1. Created user account 2. Open the website of the WeCare application
Input	• Username: KalleSvensson • Password: TestCase123!

Test steps	1. Navigate to the log-in page 2. Enter username 3. Enter password 4. Press on the log-in button
Expected output	Logged into the account
Post-condition	The member will be taken to the user dashboard.
Actual Result	Logged into the account
Status (Pass / Fail)	Pass
Test Executed by	Nima Safavi
Test Execution Date	26.05.2022

Test Case ID	TC-03
Requirement ID	SEC-04
Test description	The test case is to verify the sign in of a doctor to our platform
Pre-condition	1. Owning doctor account 2. Open the website of the WeCare application
Input	• Username: SaadDoc • Password: Saad001!!
Test steps	1. Navigate to the log-in page 2. Enter username 3. Enter password 4. Press on the log-in button
Expected output	1. Logged into the account
Post-condition	1. The user will be taken to the dashboard for doctors.
Actual Result	1. Logged into the account
Status (Pass / Fail)	Pass
Test Executed by	Dominik Pappe
Test Execution Date	26.05.2022

Test Case ID	TC-04
Requirement ID	SEC-04
Test description	The test case is to verify the sign in of the admin to our platform
Pre-condition	1. Owning admin account

	2. Open the website of the WeCare application
Input	• Username: SaadDoc • Password: Saad001!!
Test steps	1. Navigate to the log-in page 2. Enter username 3. Enter password 4. Press on the log-in button
Expected output	Logged into the account
Post-condition	The user will be taken to the admin dashboard.
Actual Result	Logged into the account
Status (Pass / Fail)	Pass
Test Executed by	Dominik Pappe
Test Execution Date	26.05.2022

Test Case ID	TC-05
Requirement ID	PAT-01
Test description	The test case is to verify a test appointment of a patient.
Pre-condition	1. Log-in to the user account as a patient
Input	• Test type: Covid • Clinic: Abcure Bershamra • Patient name: Kalle Svensson • Date & Time: 30.05.2022, 16:30 • Birth Day: 13.01.1995 • Passport number: 12345678 • Phone number: +4917221418654 • Email: kalle.svensson@gmail.com • Citizenship: Swedish • Personal identity number: 950113-1234
Test steps	1. Choose test type 2. Choose clinic 3. Choose date & time 4. Enter birthday

	5. Enter passport number 6. Enter email 7. Enter email for verification 8. Choose citizenship 9. Enter personal identity number 10. Press the button to book an appointment
Expected output	Booked appointment
Post-condition	-
Actual Result	Booked appointment
Status (Pass / Fail)	Pass
Test Executed by	Dominik Pappe
Test Execution Date	26.05.2022

Test Case ID	TC-06
Requirement ID	PAT-02
Test description	The test case is to verify the retrieval of a PCR test result from a patient
Pre-condition	1. Test performed 2. Added test result by the doctor (TC-07) 3. Log-in to user account (TC-02)
Input	-
Test steps	Click on „Check "to get access to get access of the desired test result as a PDF with QR-code
Expected output	Opened PDF file with test result
Post-condition	-
Actual Result	Opened PDF file with test result
Status (Pass / Fail)	Pass
Test Executed by	Amirhossein Soltaninejad
Test Execution Date	26.05.2022

Test Case ID	TC-07

Requirement ID	DOC-01
Test description	The test case is to verify the access to see the information about one case.
Pre-condition	Log-in as a doctor (TC-03)
Input	
Test steps	Look at the information of one case with the ReportId number and the patient.
Expected output	The patient will have access to the information about one case.
Post-condition	-
Actual Result	The patient will have access to the information about one case.
Status (Pass / Fail)	Pass
Test Executed by	Amirhossein Soltaninejad
Test Execution Date	26.05.2022

Test Case ID	TC-08
Requirement ID	DOC-02
Test description	The test case is to verify the insertion of a test result from a doctor.
Pre-condition	Log-in as a doctor (TC-03)
Input	Negative “PCR test result
Test steps	1. Click on „add test result “ 2. Choose „negative “ 3. Click on „save “
Expected output	1. The patient will have access to his test result.
Post-condition	-
Extension	-
Actual Result	The patient will have access to his test result.
Status (Pass / Fail)	Pass
Test Executed by	Khalil Mardini
Test Execution Date	26.05.2022

Test Case ID	TC-09
Requirement ID	DOC-03

Test description	The test case is to verify the access pdf test report of a test by a doctor.
Pre-condition	1. Log-in as a doctor (TC-03) 2. Inserted test result (TC-08)
Input	
Test steps	1. Click on „Check “
Expected output	Opened PDF file with test result
Post-condition	-
Actual Result	Opened PDF file with test result
Status (Pass / Fail)	Pass
Test Executed by	Khalil Mardini
Test Execution Date	26.05.2022

Test Case ID	TC-10
Requirement ID	ADM-01
Test description	The test case is to verify the ability of admins to create a new user. A new account with properties of a doctor is created for this test case.
Pre-condition	Log-in as an admin (TC-04)
Input	• Name: Sven Svensson • Username: SaadDoc • Email: sven.svensson@gmail.com • Password: Saad001!! • Role: doctor
Test steps	1. Navigate to “Create user” in the navigation bar 2. Enter name 3. Type in username 4. Enter email 5. Enter password 6. Choose role 7. Click on “Done”
Expected output	1. New user has access to the application 2. The account information will be stored in the database.
Post-condition	-

Actual Result	1. New user has access to the application 2. The account information will be stored in the database.
Status (Pass / Fail)	Pass
Test Executed by	Khalil Mardini
Test Execution Date	26.05.2022

Test Case ID	TC-11
Requirement ID	ADM-02
Test description	The test case is to verify the ability of admins to change the role of a user. An existing admin account should be changed to an account with the role doctor.
Pre-condition	Log-in as an admin (TC-04)
Input	
Test steps	1. Navigate to “Roles/Users” in the navigation bar 2. Click on “Update” for the role doctor 3. Click on the check box for the user “Admin1” 4. “Save” the changes
Expected output	1. The account “Admin1” now has the role of a doctor.
Post-condition	-
Actual Result	The account “Admin1” now has the role of a doctor.
Status (Pass / Fail)	Pass
Test Executed by	Mohammadali Rashidfarokhi
Test Execution Date	26.05.2022

Test Case ID	TC-12
Requirement ID	ADM-03
Test description	The test case is to verify the accessibility of admins to delete a user of the system.
Pre-condition	Log-in as an admin (TC-04)
Input	

Test steps	Delete user account “User1” out of the system.
Expected output	The user with the account name “User1” is deleted from the System including the database
Post-condition	-
Actual Result	The user with the account name “User1” is deleted from the System including the database
Status (Pass / Fail)	Pass
Test Executed by	Mohammadali Rashidfarokhi
Test Execution Date	26.05.2022

3. System Interfaces

User Interfaces

The user interface is the most important part with which the user interacts. If it is not designed consistently and predictably for the user, problems will occur during use. As a result, users will not find their way around and will not use the system in the end. The structure should also be designed in such a way that the desired goal is reached within a few clicks.

Look & Feel

Graphical user interface requirements for patients:

Name:	**Patient-UI (User Interface)**
Description:	User interfaces for patients must be designed so that they can be understood by the intended user group (Patients in this case). User comprehension is defined as the ability to understand each operating step of the underlying process from visualization.
Risk:	A patient who is unable to comprehend the “WeCare” application's patient interface may be unable to utilize it correctly. In the end, it may result in decreased efficiency, a user's inability to use the application when logged in as a user (patient), or dissatisfaction with how the application interacts with the user.
Rationale:	When the user (patient) has logged in, the user should be provided with a navigation bar that is situated on top of the page. The navigation bar should contain three options (buttons). The text color of the buttons is blue. However, when hovering the mouse on them, the text color should be changed to green. The last button on the right (logout) should have a border around it. The “Logout” button's text color should change to white, and the background color will change to green once the mouse has hovered on it.

	Below the navigation bar, a header should be presented in grey with three columns (Report id, Patient Sample Name) and the user detail based on the columns will be presented below them. Lastly, by default, there will be no scroll bar presented. However, by booking more tests, more detail will be shown therefore, the scroll bar will be presented.

Graphical user interface requirements for admin:

Name:	**Admin UI (User Interface)**
Description:	The admin graphical user interface should be comprehensive and straightforward and help in doing things quickly in a way that allows the admin to access the UI and apply changes within the "WeCare" application such as adding new users to the system (patients or doctors) and deleting users. Even the admin UI should be organized and designed well so it will facilitate the admin in achieving the designated task.
Risk:	It may not be possible for an admin who is not familiar with the "WeCare" application's admin interface to utilize it properly. As a result, the application can be less efficient, or the admin might not be able to use or manage it. In addition, administrators may not be satisfied with how the application responds to them.
Rationale:	After the admin logs in, the administrator is provided with a navigation bar menu that includes five buttons, namely "Users", "Create User", "Users/Roles", "Create role" and "Log Out." These buttons are located on the right side of the navigation bar menu, while on the left side is the "WeCare" logo. Moving on, by default, when the admin has logged in to the system, a table with different columns named (ID, role name, update, delete) should be visible to the admin. In the delete column, the bottom text should be

	white and the button background should be red. When hovering over this button, the background should be changed to grey.

Graphical user interface requirements for doctor:

Name:	**Doctor UI (User Interface)**
Description:	The user interface for the doctors should be designed in a perfect and comprehensive design structure that will facilitate the doctor's job in achieving the task such as (adding the patient test results to the system and checking the patient personal data).
Risk:	The new doctor to the system might not get used to the doctor user interface straight or have a lack of knowledge in dealing with the doctor UI platform page.
Rationale:	When the doctor has logged in, the doctor will be provided with a navigation bar menu on the upper part of the window that will include two options to select which are "Add Result" and "Log out" they are located on the right side of the navigation bar menu, while on the left side it includes the "WeCare" logo. The doctor will be able to determine the patient's final test result by clicking on the green "Add result" button that eventually will navigate to a new page to allow the doctor to set the final test result. The central part of the page will display the patient's data that are categorized in columns. The patient personal data includes (Report id, patient name, and Sample Name).

Graphical user interface requirements for main page:

Name:	**Main Page UI (User Interface)**
	The user interface for the main page of the "WeCare" application should be easy

Description:	to access and comprehensive for all the users who are (patients, doctors, and admins). The “WeCare” main page provides the users with general information such as (terms and conditions of the application, privacy policy and means of communication with the laboratory). Each user will be able to log in (depending on the role of the user) and be able to access more features within “WeCare” application.
Risk:	The new user to the system might not get used to the main page of the user interface straight or have lack of knowledge in dealing with the UI platform of the main page.
Rationale:	A navigation bar menu with several options (Home, privacy, doctors, contact, login, and register) should be available when the user accesses the "WeCare" main page, regardless of their role. The text color of the log in and register buttons should be white, and the background color should be green. A grey background color should appear once the mouse has hovered over these two buttons. On the main page, an image of laboratory staff should be visible as the background along with a button called book a PCR test. The book a PCR test button operates in the same way as the login and register buttons. There should be a list of doctors above the footer, with a representative image, and two buttons allowing users to swipe right or left. White should be used for the button's text and red for the background. By hovering over the buttons, the text color should change to black and the background color to white.

	Finally, the footer should appear at the bottom and the background color should be grey.

Note for the clarification of the above tables:

Description:
In the description part, a brief explanation of what the interface is about and what it covers.

- Risk:

The risk section will indicate what will happen if the graphical user interface has been designed based on the wrong requirements or if the user does not have any per-knowledge of dealing with the “WeCare” application UI.

- Rationale:

After listing the possible risks that can affect the specific graphical user interface, the rational part in the above tables will indicate the requirements that should be followed in the “Wecare” application to avoid the possible risks.

Layout and Navigation Requirements

The home screen of our application should list useful information about the topic of our system. At the top of the page, a header should be presented to enable the users to navigate the application. Since a user account is required to use the system, the menu item for registration and login should be highlighted. There should also be the option to scroll to the left and right with the help of the buttons on the main page to see the list of the doctors and what they are responsible for.

After a user has logged in, they will see a button at the top right to log out. However, the menu items differ between the user types of Patients, Doctors, and Admin.
The patient also sees a menu item in his menu bar at the top to book an appointment, as this is the main task of our application for the patient. After navigating to the menu item "make an appointment", he sees several forms that he must fill out to book an appointment. After navigating to the menu item "make an appointment", he sees several form fields (including test type, clinic, and time) that he must fill in to book an appointment.

A doctor's home page shows all tests, with a test result. In the navigation bar, the doctor can then add a result for new tests.

The start page of an admin shows all user accounts that have access to the system. In the navigation bar, the admin can choose between "Create User", "User/Role" and "Create Role". Also, the admin should be provided with a table with information about the registered users and their roles.

Consistency

Consistency in design is one of the most essential elements to leave a uniform impression of the brand/company on the one hand and to make it easier for users to use the application, as they have certain expectations after using a part of the application. Therefore, we have different requirements for the application:

- Logo: The logo should be placed at the top of the page (in the top left of the page in the navigation bar) so that the user remembers the page, as images stay in the mind better than words.

- Color: the presentation of the pages should be quite simple, which is why they are mostly white and the elements (buttons, navigation) that should be highlighted are displayed in light green.

- Navigation elements: our navigation should be displayed at the top of the page, the so-called header. To make navigation as easy as possible, there should be a very low hierarchy. The navigation bar should be accessible on every sub-page of the application. The page on which the user is currently located should be highlighted in a green font.

- Terminology: as few abbreviations as possible should be used for terms, as the experience in healthcare and the use of digital devices varies between users. Therefore, basic icons should also be explained when going over them with terms or even be supplemented in important places.

- Placements for entering/presenting data: The entry of data and the presentation should be as central as possible, as this is where users first look for new information as soon as a change is made to the screen.

User Personalization & Customization Requirements

Personalization depends on what type of user logs into the system. As we have 3 different types of users (patient, doctor and administrator) for our system, the home screens and menu items change for the user as each of the users has a different goal.

Interfaces to External Systems or Devices

The initial idea was to connect the “WeCare” web-application with an external payment system that will manage the financial transaction of the “WeCare” application, but due to the reason that this student project needs to be accomplished within a short time (two months) and we as a group members estimated that it requires longer time to connect our “WeCare” application with a platform that handles the payments procedure. Therefore, we did not connect any external system to the “WeCare” application.

Software Interfaces

The software interfaces that will be associated with the WeCare web application platform are as follows:

- Web browser applications such as (Google Chrome, Firefox, Microsoft Edge, etc...) that will run and display the “WeCare” webpage on the default port number will be 8080.

- The IDE programming environment, which will be for now Visual Studio but might change in the future, depends on the availability of the development tools and the “WeCare” web application will be built on C# and . NET.

- The “WeCare” web application will be compatible with working on most of the operating systems, (in our case, it will be on Windows OS).

Hardware Interfaces

- The hardware interfaces vary from cables, sockets, plugs and electrical cables such as hardware interfaces can be Ethernet connections (wired or wireless), USB plugs that could help in transferring data between the “WeCare” system and other devices (USB flash, external hard disks). Electrical cables that supply an energy source to run the device and make the “WeCare” accessible.

The minimum hardware requirements for the running of the WeCare web applications are as follows:

Processor	1.9 GHz x86- or x64-bit dual-core processor
Memory	2 GB RAM
Display	VGA option with resolution of 1024 x768
Others	Excellent set of plugs, sockets, cables, and computer equipment

Fulfilling these hardware requirements will ensure the functionality of the WeCare web-application

Communications Interfaces

The communication interfaces that will be linked to the WeCare web application are as follows:

- Local area network that will facilitate shifting critical input data between the admin side in the WeCare web application and the database server.
- The Internet connection will allow establishing a direct link between the customer (Client-side) and the WeCare web-application

4. Business Rules

4.1 Business Rules

4.1.1 Patient log-in
If the patient has registered on our platform, he can log in to the site.

4.1.2 Doctor log-in
If a doctor wants to log in to our site, an admin must create an account for him.

4.1.3 Book appointment
If the patient has logged in to our application, then the patient can book a test.

4.1.4 Data form validation
The users should fill out a request form when they want to book a PCR-Test, and it is essential to enter a social ID or passport number in the form to have a valid QR-Code. As a result, the input data for this section must be filled or the booking will not be registered in the system.

4.1.5 Test status
When the patient has completed a test, they can see the status on the home page.

4.1.6 Insert test result
If the patient has taken a test, then the doctor can enter a test result into the system.

4.1.7 Open test result
If the test result has been entered into the system by a doctor, the PDF file with the QR code can be retrieved by the patient or doctor.

4.1.8 Log-out
Users (patients, doctors, and admins) can only log out if they have previously logged in.

4.2 Business rules categorization

Business Rules	Customer Validation Rules	Process Flow Rules
4.1.1	X	
4.1.2	X	

4.1.3	X	
4.1.4		X
4.1.5		X
4.1.5		X
4.1.6		X
4.1.7		X
4.1.8	X	

5. System Constraints

During the design of this project, several constraints were outlined. Those constraints include the following:

- **Constraints of server**

Because the project is a student project and due to the need to provide and spend budget and time. It is not possible to use online servers to provide the web application with an online server.

- **Constraints of time**

Because the time allocated for the project is about two months, it is not possible to add additional features such as an online payment system.

- **Constraints of platform support**

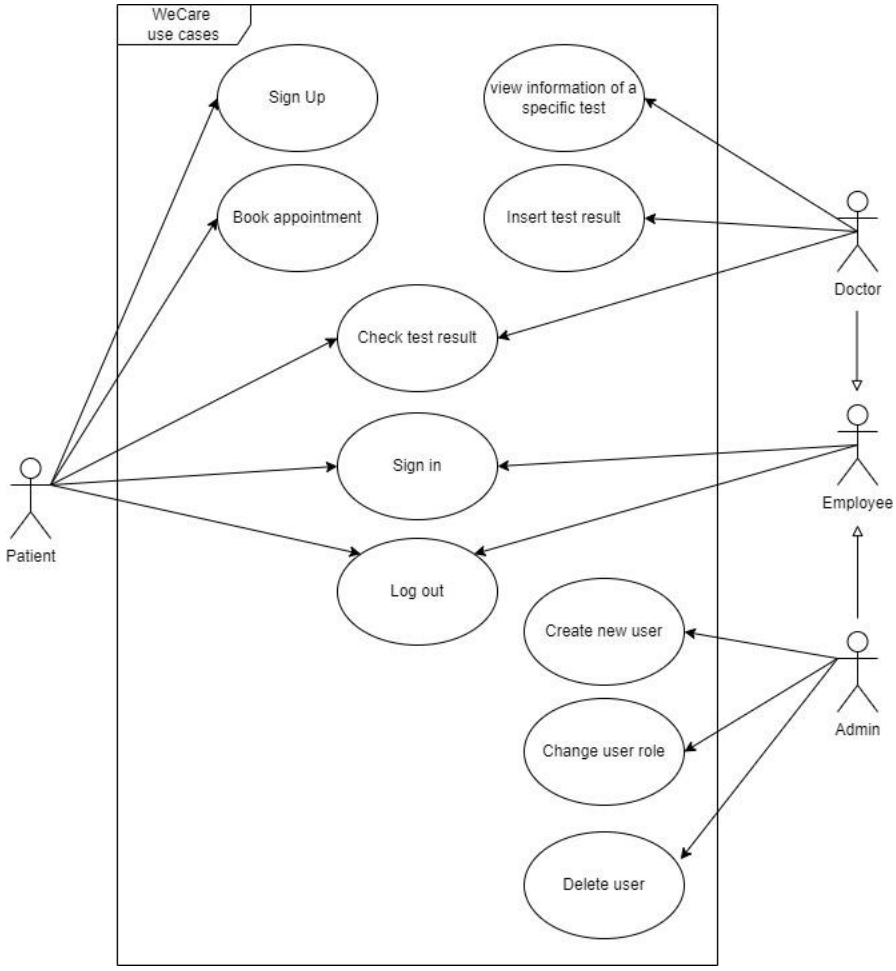
Due to a lack of time and lack of experience in developing mobile applications, our application is limited to the web browser.

- **Constraints of resources**

Since this is a student project, the lack of financial resources can affect the project, as it limits the project in terms of developing commercial features like having contracts with banks and online banking systems such as Klarna for the payments transactions.

6. Use Cases

6.1 Use-Case Diagram



6.2. Use-Cases:

6.2.1 Sign up

Brief description	The patient wants to sign up for the "WeCare" application.
Actors	Patient
Precondition	Accessing the WeCare web application via the web browser.
Basic Flow of Events	1. Navigate to the registration page 2. Enter email-address 3. Enter full name 4. Enter username 5. Choose a valid password. 6. Repeat the password 7. Agree terms & conditions 8. Press on the signup button
Alternative flows	1. If in step <2> of the basic flow the patient types in an email without the "@" in it then 1.1 The patient will see an error message that an email address without the "@" sign is invalid. 1.2 The patient will enter a new email address 2. If in step <2> of the basic flow the patient types in an email without anything before the "@" sign then 2.1 The patient will see an error message that an email address isn't entered correctly. 2.2 The patient must add alphabets to the email address before the @ sign. 3. If the patient cancels the registration in any step, then 3.1 There will be no created user account for booking an appointment for a test.

Subflow	-
Key scenarios	1. Created user account
Post conditions	1. The member will be taken to the user dashboard.
Specific requirements	-

6.2.2 Sign in:

Brief description	The user wants to sign in to the WeCare platform by inserting the login data (email and password) of his account.
Actors	Patient, doctor, and admin
Precondition	1. Existing user account
Basic Flow of Events	1. Navigate to the log-in page 2. Enter username 3. Enter password 4. Press on the log-in button
Alternative flows	1. If in step <4> of the basic flow username or password is wrong, then 1.1 The user will see an error message that the login credentials are wrong. 1.2 The user will enter new login credentials 1.3 The user will press the log-in button again.
Subflow	1. Password forgotten 1.1 The user presses the “forgot password” button. 1.2 The user has to type in his email address. 1.3 The user has to press the button to receive an email. 1.4 The user will receive an email to get access to his user account again.
Key scenarios	The member will be taken to the user dashboard.
Post conditions	Logged in to the user account
Specific requirements	-

6.2.3 Book appointment**:**

Brief description	The patient wants to book an appointment for a test (Covid, Chlamydia, HIV or Pregnancy) in one of the laboratories. To do this, he or she must select a time and enter his or her personal data.
Actors	Patient
Precondition	1. Created user account 2. Logged in to account
Basic Flow of Events	1. Choose test type 2. Choose clinic 3. Choose date & time 4. Enter birthday 5. Enter passport number 6. Enter email 7. Enter email for verification 8. Choose citizenship 9. Enter personal identity number 10. Press the button to book an appointment
Alternative flows	-
Subflow	-
Key scenarios	Booked appointment
Post conditions	The patient will get back to the dashboard.
Specific requirements	-

6.2.4 View test information**:**

Brief description	For the doctor to prepare for the test, he wants to see detailed information such as the type of test and information about the person.
Actors	Doctor
Precondition	1. Logged into doctor account
Basic Flow of Events	1. The doctor looks at the information on a test on the start page.
Alternative flows	-
Subflow	-

Key scenarios	The doctor now has precise information about the test and can prepare for it.
Post conditions	-
Specific requirements	-

6.2.5 Insert test result**:**

Brief description	After the patient has taken the test, the Doctor wants to enter the test result into the system so that the user can access the corresponding record.
Actors	Patient
Precondition	1. Completed test 2. Logged into doctor account
Basic Flow of Events	1. Click on „add test result “ 2. Choose „negative “ 3. Click on „save “
Alternative flows	1. If the completed test was positive, then the doctor should enter “positive” in step <2> of the basic flow
Subflow	-
Key scenarios	The result of the test performed has been inserted into the system.
Post conditions	The patient will have access to his test result.
Specific requirements	-

6.2.6 Check test result**:**

Brief description	Patients or doctors would like to have access to the PDF file with QR code after the test result has been entered into the system, which serves as proof of the test.
Actors	Patient, doctor
Precondition	1. Logged into user account 2. Booked appointment 3. Completed test 4. Inserted test result by a doctor

Basic Flow of Events	1. Click on „Check "to get access to the desired test result as a PDF with QR-code.
Alternative flows	-
Subflow	-
Key scenarios	Open the PDF-file with all information about the patient, the test and the result.
Post conditions	-
Specific requirements	-

6.2.7 Create new user account**:**

Brief description	Admins are able to create new accounts for users. Since, for example, doctors and admins cannot register, new accounts must be created for them.
Actors	Admin
Precondition	1. Logged into admin account
Basic Flow of Events	1. Navigate to “Create user” in the navigation bar 2. Enter name 3. Type in username 4. Enter email 5. Enter password 6. Choose role 7. Click on “Done”
Alternative flows	-
Subflow	-
Key scenarios	Created user account
Post conditions	The new user can now log in to our application
Specific requirements	-

6.2.8 Change user role**:**

Brief description	The admin has the ability to change the role of a user.
Actors	Admin
Precondition	1. Logged into admin account
Basic Flow of Events	1. Navigate to “Roles/Users” in the navigation bar

	2. Click on “Update” for the role doctor 3. Click on the check box for a user 4. “Save” the changes
Alternative flows	-
Subflow	-
Key scenarios	The role of a user has now changed.
Post conditions	The user now has different access to the application.
Specific requirements	-

6.2.9 Delete user**:**

Brief description	To keep the data up to date and to prevent misuse, the admin can remove certain user accounts from the system.
Actors	Admin
Precondition	1. Logged into admin account
Basic Flow of Events	1. Delete a user account out of the system by pressing the “delete” button on the dashboard.
Alternative flows	-
Subflow	-
Key scenarios	Deleted user from the system
Post conditions	The deleted user account no longer has access to the system.
Specific requirements	-

6.2.10 Log-out of the application**:**

Brief description	After the user has reached his desired objective, he wants to log out of the application again.
Actors	Patient, doctor, admin
Precondition	1. Logged in to account
Basic Flow of Events	1. The user presses the log-out button on the top right corner.
Alternative flows	-
Subflow	-

Key scenarios	1. The user will be logged out from the application and comes back to the home page.
Post conditions	-
Specific requirements	-

Appendix – Time Report

Date	Member	Activity	Reviewed By	Time (hours)
2022/04/18	Nima Safavi	Writing Section 4	Nima Safavi	6 hours
2022/04/19	Mohammadali Rashidfarokhi	Writing Section 1 and 2	Dominik Pappe	10 hours
2022/04/19	Khalil Mardini	Writing Section 3.2	Amirhossein Soltaninejad	1 hour
2022/04/19	Dominik Pappe	Writing Section 3.1	Nima Safavi	5 hours
2022/04/20	Amirhossein Soltaninejad	Writing Section 5 Grammar correction	Mohammadali Rashidfarokhi	8 hours
2022/04/20	Dominik Pappe	Writing Section 1	Khalil Mardini	2 hours
2022/04/20	Khalil Mardini	Writing section 3.2&3.2.1&3.2.2	Mohammadali Rashidfarokh	4 hours
2022/04/21	Mohammadali Rashidfarokhi	Writing Section 7	Khalil Mardini	3 hours
2022/04/20	Mohammadali Rashidfarokhi	Writing Section 6	Amirhossein Soltaninejad	4 hours
2022/04/20	Khalil Mardini	Writing section 6	Dominik Pappe	1 hour
2022/04/21	Nima Safavi	Section 6 diagram	Dominik Pappe	2 hours
2022/04/21	Dominik Pappe	Writing section 6	Khalil Mardini	1 hour

Reference

[1] Oliveira, S., Galina, L. and Pijoan, C., 2001. Development of a PCR test to diagnose Haemophilus parasuis infections. Journal of veterinary diagnostic investigation, 13(6), pp.495-501.