

**Table of Contents**

[**I.**](#_heading=h.gjdgxs) **Executive Summary 2**

[**II.**](#_heading=h.30j0zll) **Competitive analysis 3**

[**III.**](#_heading=h.1fob9te) **Data definition 3**

[**IV.**](#_heading=h.3znysh7) **Overview, scenarios, and use cases 3**

[**V.**](#_heading=h.2et92p0) **Initial list of high-level functional requirements 3**

[**VI.**](#_heading=h.tyjcwt) **List of non-functional requirements 3**

[**VII.**](#_heading=h.3dy6vkm) **High-level system architecture 3**

[**VIII.**](#_heading=h.1t3h5sf) **Team and checklist 3**

[**IX.**](#_heading=h.4d34og8) **References 3**

# Executive Summary

As of now, the COVID-19 pandemic has impacted every area of our lives. Therefore, we cannot have the same routine as we were used to when we must distance ourselves and limit the exposure to others as much as possible. The number of COVID-19 patients is increasing dramatically to the point that some hospitals are struggling to keep up with the rising demand for beds and shortage of staff. Illness severity can range from mild to critical. All deaths usually occurred among patients with critical illness while patients with mild symptoms can be treated at home with the doctor’s instructions. Speaking of it, we would like to introduce our application to ensure everyone with COVID-19 symptoms can get safe home care and prevent the spread of coronavirus through close contact by going to a hospital while they can receive certified doctor’s instructions via a website application called CyberHealth.

CyberHealth is an online health monitoring platform where patients with COVID-19 symptoms can upload their medical history as well as vital signs in order to receive treatment from certified doctors. Licensed doctors will determine if you should be tested or not and provide timely care based on the illness severity. Our virtual care application commits to deliver a seamless user experience, responsive web design, and a user-friendly interface to all users regardless of patients or doctors.

# 

# II. Competitive analysis: (Huy - in process)

|  | CyberHealth | Teledoc | Mdlive | Mymdnow | Private Offices |
| --- | --- | --- | --- | --- | --- |
| Online visit | 5 | 5 | 5 | 5 | 0 |
| Fast & simple process | 5 | 4 | 4 | 3 | 2 |
| User interface |  |  |  |  | 1 |
| Covid’s doctor accessibility |  |  |  |  |  |
|  |  |  |  |  |  |

# Data definition

(Nha Tran: I will create a table for this part, but for now I will write it like this. Feel free to add any ideas on this.)

1. Home – User interface – The introductory page of a website

2. Service – User interface – List what we offer for users

3. Sign in – User service – Existing patients can upload their health information and see their treatment while doctors can see their patients and provide treatment

4. Register – User service – Allow new patients to create an account

5. User - Actors - Refer to both patients and doctors

6. Member - Actors - An user who has an account with the system

7. Non-member - Actors - An user who does not have an account with the system

8. Admin - Actors - An user can see doctor’s information as well as patient’s information

9. Profile - Use case scenarios - Store user’s information

10. Status - Use case scenarios - Display the illness severity of patients

11. Comments - User case scenarios - Leave a note

# Overview, scenarios, and use cases

# Initial list of high-level functional requirements

(Nha Tran: The formatting for this part I will do later. Right now I will focus on the contents. Feel free to edit anything.)

**Non-member**:

1. Browse the main homepage

1.1 Interacting with Home page and Service page

* User can use the navigation bar and interact with the Home page and Service page buttons
* The system shall let this user see the contents of the main Home page and Service page

1. Create an account

2.2 Entering and Storing information

* Users can create an account by entering their information such as username, email address, password, first name, last name, date of birth, phone number, and location
* The system shall validate if the username is registered or not
* The system shall make sure there is no empty field when users create their account
* The system shall display a message if an account is successfully created and redirect the user to the Sign in page.
* The system shall store user’s information provided above

**Member:**

1. Sign in

1.1 Sign in with registered account

* Users can sign in if their username and password are registered with the system
* Users can see their interface depending on which role they are logging in. For example: if he/she is a patient, he/she will be logged in as a patient user which is different from Admin’s view or doctor’s view

# List of non-functional requirements: (HN - Done - Feel free to add)

1. Performance: loading time should not exceed 1 second for users
2. Reliability: users can access the website 98% of the time without failure.
3. Recoverability: if problems happen to the website, it should be recovered no more than three days for major ones and 8 hours for minor ones.
4. Security \*\*\*: Only admin can view doctors and patients information. Only doctors can view a patient's medical record.
5. Compatibility: the website must works on multiple browsers(chrome, safari, etc) and devices(tablet, phone, laptop, ect)
6. Usability\*\*\*: the website must be user-friendly and prioritize user experience.
7. Data integrity\*\*\*: the system must keep all doctors and patient data secure and fully back-up for every record.

\*\*\*: prioritize requirement.

# High-level system architecture: (Huy - Done so far. Add anything.)

1. Lamp server: we will hosting our Fall 2021 project on <https://lamp.cse.fau.edu/~cen4010_fa21_g22/>
2. WhatsApp: is the main communication tool for us.
3. Jira: we will use Jira to track our project process and tasks.
4. Github: will be used to building and editing our project together.
5. MySQL database: will be the main database for the project.
6. Languages:
   1. HTML: will be used to display all the documents on the website.
   2. CSS: will be used to decorate the website and pages.
   3. PHP: will be used to manage data and contents.
   4. Javascript: will be used for most back-end development of the project.
   5. Bootstrap: will be used for the initial construction for the website.

# Team and checklist

Group name: 22

Scrum master:

Product owner:

Front End Developer:

Back End Developer:

​​a) Team decided on basic means of communications

b) Team found a time slot to meet outside of the class

c) Front and back end team leads chosen

d) Github master chosen

e) Team ready and able to use the chosen back and front-end frameworks

f) Skills of each team member defined and known to all

# References

<https://winatalent.com/blog/2020/05/what-are-non-functional-requirements-types-and-examples/>