**Life Cycle Objective Milestones**

Assignment 2

Group 2:

* Sobana Handi Achini Thisarangi De Silva
* Dehemi Vihara Dissanayake Liyanage
* Gaury Chethana Thanthirigama
* Temuulen Tsengel
* Michael Weisang

Deliverables:

1. Project Vision
2. Domain Class Model
3. Use Case Description and model
4. Non-Functional Requirement Specification
5. Architectural Notebook
6. Risk List
7. Master Test Plan
8. Initial Project Plan
9. Technical Competency Demonstrator
10. Inception Phase Status Assessment

Remote Mental Health Management

“Naaman” Vision

# Introduction

During the COVID era of lockdown, there are a lot of concern regarding the mental health cause by the lack of social interaction. Henceforth, we have decided to tackle this issue and develop a web-based platform for ITC303 called “Naaman”. Project “Naaman” is a remote mental health management platform that are envisioned by our team “Runtime Terror”. Naaman aim is to help decreasing the mental health issue by providing a platform where people can share their stories and get an online consultation.

# Positioning

## Problem Statement

|  |  |
| --- | --- |
| The problem of | Low mental health |
| affects | Social distancing people |
| the impact of which is | Low productivity and suicide tendency |
| a successful solution would be | catharsis |

## Product Position Statement

|  |  |
| --- | --- |
| For | Low mental health people |
| Who | In a mental breakdown territory |
| The Naaman | Social media platform |
| That | Give opportunity to share their story or get online consultation |
| Unlike | Social media where people know who they are |
| Our product | Can provide it anonymously and get expert help |

# Stakeholder Descriptions

## Stakeholder Summary

| **Name** | **Description** | **Responsibilities** |
| --- | --- | --- |
| Developer team | Group 2 “Runtime Terror” | Develop the application |
| Senior leader | Ather Saeed | Provide guidance |
| Physician | Mental health clinician and mental health practitioner | Giving online consultation and help resolving mental health issues |
| User | The social media user | Registering and receive mental health support |
| Helpdesk Support | Employee | First help support and managing online booking |

## User Environment

  Due to the nature of the application being a personal and sensitive issue, the user interface will focus on a single user usage only. The user can take their time on using the application as vocalizing their thoughts is not an easy task. But this will only be true for the social media aspect of the application. For the online consultation’s part, each user is given a specific time range to communicate with the physicians. The users can access the application through the browser on any device at any time.

# Product Overview

## Needs and Features

|  |  |  |  |
| --- | --- | --- | --- |
| **Need** | **Priority** | **Features** | **Planned Release** |
| Sharing stories | High | Posting a public thread | On launch |
| Online consultation | High | 1 on 1 communication between the user and the physician (voice call, video call, or direct messages) | On launch |
| Anonymous stories sharing | Medium | User can decide the account to be anonymous when registering or editing the account. | On launch |
| Prevent harmful user | High | Users need to create an account and go through the authentications before having access to the comment feature. | On Launch |

# Other Product Requirements

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Priority** | **Planned Release** |
| EIC 62304 Standards for medical software | High | On Launch |
| WCAG 2.0 Standard | Medium | On Launch |
| Browser that supports PHP | High |  |
| FAQ Document | Medium | On Launch |

**Domain Class Model**

**Diagram

Description automatically generated**

# Mental Health Management system (Naaman) use case

Diagram

Description automatically generated

# Use-Case: <Access the web >

1 Brief Description

This use case allows to access the Remote Mental Health Management system public webpage for the user.

2 Actors

2.1 User

2.2 System

3 Pre-Conditions

* 1. The user must be a new user or have not logged in to the system

3.2 Website must be online

4 Normal Flow

The use case begins when user access web browser

|  |  |
| --- | --- |
| Actor | System |
| user request the Remote Mental Health Management system over the web browser | **1** check the request and confirm user status (if user has not logged into the session)  **2** create and send the related content over the website for the user. |

The use case ends.

5 Alternate Flows

if the user is register to the system and user logout the system

|  |  |
| --- | --- |
| Actor | System |
| request logout the system | **1** destroy the current login session  **2** create and send the related content over the website for the user. |

6 Sub flows

Request with wrong web address

|  |  |
| --- | --- |
| Actor | System |
| if user request is not correct | **1** check the request and confirm user status (if user has not logged into the session)  **3** if main domain is correct, redirect website main public page (home page with articles)  **2** create and send the related content over the website for the user. |

7 Key Scenarios

New and registered users visit this web page first. if user is new system generate website content using his geological location data. if user is currently registered, using his past data for generating the website content.

8 post-conditions

Add new record in to the user log with login IP address

9 Special Requirements

9.1 Minimum loading time

9.2 Mental Health Management System Security

9.3 UI and UX

9.4Language support

# Use-Case: < Register New User>

1 Brief Description

This use case allows to register new user into the Mental Health Management system

2 Actors

2.1 User

2.2 System

3 Pre-Conditions

3.1 The user must be new user

* 1. Website must be online

4 Normal Flow

The use case begins when user access web page (home page)

|  |  |
| --- | --- |
| Actor | System |
| **1** user request the Remote Mental Health Management system over the web browser | **2** check the request and confirm user status (if user has not logged session)  **3** create and send the related content over the website for the user. |
| **4** user request new registration | **5** The system generates a register form and an image with 4 random rotated letters. |
| **6** The user fills the text of their message, their full name and email. Next, they rewrite the text from the image into the appropriate field and submit the register form. | **7** The system validates the user data and stores the user data  **8** System send email to the user with register confirmation |
| **9** user confirms the email | **10** System create new login for the requested user |

The use case ends.

5 Alternate Flows

if new user visits the website and try to comment, make a post or other tasks.

|  |  |
| --- | --- |
| Actor | System |
| **1** User request a task (comment, share post or story) | **2** check the user session. if user session not in current register user list, system sends new user register form to the user |
| **3** The user fills the text of their message, their full name and email. Next, they rewrite the text from the image into the appropriate field and submit the register form. | **4** The system validates the user data and stores the user data  **5** System send email to the user with register confirmation |
| **6** user confirms the email | **7** System create new login for the requested user |

6 Sub flows

User doesn't complete and submit new user register form

|  |  |
| --- | --- |
| Actor | System |
| 1 User submit incomplete register form | **2** System starts the validation process and request correct data from the user (the register process restart) |

7 Key Scenarios

There are two main types of new user registration process. One is the user request for the registration process and the other is the system request registration process for the user. The system makes such a request when the user tries to perform a task on the system without signing in.

8 post-conditions

8.1 Store new user details and new login details

8.2 start new temporary session for the new user and update home page

9 Special Requirements

9.1 Minimum loading time

9.2 Mental Health Management System Security

9.3 UI and UX

9.4Language support

# Use-Case: < User Login>

1 Brief Description

This use case allows to login user in the Mental Health Management system

2 Actors

2.1 User

2.2 System

3 Pre-Conditions

3.1 The user must be register user

3.2 Website must be online

4 Normal Flow

The use case begins when user access web browser

|  |  |
| --- | --- |
| Actor | System |
| **1** user request the Remote Mental Health Management system over the web browser | **2** check the request and confirm user status (if user has not logged session)  **3** create and send the related content over the website for the user. |
| **4** user requests the login page | **5** system send the login page to the user |
| **6** users enter login details | **7** System start the validation process.  **8** if user-entered login details correct, system reload the page and set the user session. |

The use case ends.

5 Alternate Flows

user-entered login details incorrect

|  |  |
| --- | --- |
| Actor | System |
| **1** user enter login details | **2** System start the validation process.  **3** if user-entered login details incorrect, System send notification to the user about current status. and count user attempt |
| **3** user restarts the login process |  |

6 Sub flows

Exceeding user login attempt limit

|  |  |
| --- | --- |
| Actor | System |
| 1 Exceeding user login attempt limit | **2** Disable selected user logins and send email to the user. additionally show notification to the user about situation |

7 Key Scenarios

There are two main types of user login process. One is the user request for the login process and the other is the system request login process for the user. The system makes such a request when the user tries to perform a task on the system without sign in. system check the user session-log and identified registered-user. after system ask logging details for the user.

8 post-conditions

8.1 add record for the log-details

8.2 start new temporary session for the new user and update home page

9 Special Requirements

9.1 Minimum loading time

9.2 Mental Health Management System Security

9.3 UI and UX

9.4Language support

# Use-Case: < post article, story or Comment>

1 Brief Description

This use case allows to post an article, post a story, or comment an article of the Mental Health Management system

2 Actors

2.1 User

* 1. System

3 Pre-Conditions

3.1The user must be a registered user

3.2 Website must be online

4 Normal Flow

The use case begins when user logged into the System

|  |  |
| --- | --- |
| Actor | System |
| **1** user request the task (post article, stories or comments) | **2** System start the validation process.  **3** if user session is active, system give permission for the task |
| **4** users do the task (post or comment) | **5** system ask privacy level |
| **6** user defines the privacy level | **7** system store the data and send notification to the user.  **8** send task to the moderator |

The use case ends.

5 Key Scenarios

These are the main function of the Remote Mental Health Management system public page. These are used to share ideas with users. Here the user comments and performs other functions under some control.

6 post-conditions

6.1 add record for the log-details

6.2 The new story, article, comment will be stored correctly in the database with privacy limit in the task.

7 Special Requirements

7.1 Minimum loading time

7.2 Mental Health Management System Security

7.3 UI and UX

7.4Language support

# Use-Case: < Post and comment Approval process >

1 Brief Description

This use case allows approval to the user task of the Mental Health Management system

2 Actors

2.1 Moderator

* 1. System

3 Pre-Conditions

3.1 The moderator must be login

3.2 Website must be online

4 Normal Flow

The use case begins when user log the System

|  |  |
| --- | --- |
| Actor | System |
| **1** check the user task | **7** system shows the user task to the moderator |
| **4** moderator gives to permission for the user task (if post and comment under the rules and condition) | **5** system start to publish the user task in Remote Mental Health Management system public page  **6** the system send notification to the customer about task (post or comment) approval |

The use case ends.

5 Key Scenarios

This use case is described about user added post and comment approval process. Users come in different ways. therefore, system needs to control the activity. because, some users try to break the rules and regulation, it will be affected for the website quality. This approval function avoids that kind of things in system.

6 post-conditions

6.1 add record for the log-details

6.2 The system start to publish user added post or comment in the public page

7 Special Requirements

7.1 Minimum loading time

7.2 Mental Health Management System Security

7.3 UI and UX

7.4 Language support

# Mental Health Management system (Naaman) private page (online consultation booking)

Diagram

Description automatically generated

# Use-Case: <Make Private Appointment>

1 Brief Description

This use case allows to access the Remote Mental Health Management system to make private appointment for the user.

2 Actors

2.1 User

2.2 System

3 Pre-Conditions

3.1 The user must be a registered user

3.2 Website must be online

4 Normal Flow

The use case begins when user logged into the System

|  |  |
| --- | --- |
| Actor | System |
| **1** User requests to make private appointment for the consulting. | **2** system sends appointment-make form. |
| **3** User enter required data and submit the form. | **4** The system checks the available date and time  **5** The system shows the available date and time for the user |
| **6** User select date and time | **7** the System request a confirmation from user. |
| **8** User give the confirmation | **9** System store the new appointment |

The use case ends.

5 Alternate Flows

The Mental Health Management system check the user behaviours and suggest the private appointment to user for the consultation

|  |  |
| --- | --- |
| Actor | System |
|  | **1** the Mental Health Management system send email to the user explain benefits about consulting |
| **2** if user like to make appointment, user click the link in email | **3** the system navigate user to the appointment form  **4** system sends appointment-make form. |
| **5** User enter required data and submit the form. | **6** The system checks the available date and time  **7** The system shows the available date and time for the user |
| **8** User select date and time | **9** the System request a confirmation from user. |
| **10** User give the confirmation | **11** System store the new appointment |

6 Sub flows

user cancels the appointment before confirmation

|  |  |
| --- | --- |
| Actor | System |
|  | **1** system asks user about appointment condition (remove or add to the draft) |
| **2** User select draft | **3** The system stores theappointment as a draft appointment |

7 Key Scenarios

This is the Mental Health Management system private section. in this section can make appointment to the user for meet a consultant. if user need specialist about Mental Health, user can use this part for the meet a specialist.

8 post-conditions

8.1 Store the new appointment

8.2 Redirect web page for the home page

8.3 Show notification to the use about appointment details

9 Special Requirements

9.1 Minimum loading time

9.2 Mental Health Management System Security

9.3 UI and UX

9.4Language support

# Use-Case: < View user's Appointment>

1 Brief Description

This use case allows consultants to view the user made appointment

2 Actors

2.1 Consultant

2.2 System

3 Pre-Conditions

3.1 The moderator must be login

3.2 Website must be online

4 Normal Flow

The use case begins when user logged into the System

|  |  |
| --- | --- |
| Actor | System |
| **1** Consultant requests to view the user's made appointment list | **2** System shows the appointment |
| **3** Consultant is selecting the related appointment and check the date and time  **4** Consultant is making approval to the selected appointment | **5** System store the approval  **6** system send notification to the user about the appointment approval. |

The use case ends.

5 Key Scenarios

this use case allows the Consultant view the appointment and confirm. after confirmation, system will send the notification to the user about the situation. in that notification will have Consultant name and other details. after receiving conformation, user can ready to consulting.

6 post-conditions

6.1 add record for the log-details

* 1. The system is sending notification and email to the user

7 Special Requirements

7.1 Minimum loading time

7.2 Mental Health Management System Security

7.3 UI and UX

7.4 Language support

**“Naaman”**

**Non-Functional Requirements Specification**

# Introduction

## 1.1 Purpose

This document's main goal is to show the criteria for a remote mental health management platform that improves mental wellness through online consultation. The paper contains a thorough description of the functional and non-functional needs that have been suggested. The goal of this project is to create a simple way to keep track of mental health management documents.

## 1.1 Intended Audience and reading Suggestions

This document acts as a contract between the parties involved. The purpose of this document is to help stakeholders, the design team, developers, project managers, and the testing team better understand the system requirements.

## 1.2 Scope of Development Project

This web based remote mental health management platform is an online community designed to manage mental health consultations of users. This application provides the features to share thoughts, online consultation booking and access to professional help. Also this website provide the features blogging information all at once. The ultimate goal of this complete system is to upgrade users’ mental health to a better state.

The project is simple to apply in a variety of circumstances. Reusability is possible in this system as there is flexibility in all the modules. Therefore, new features can be added in to the system. HTML, CSS, JavaScript, PHP are the language used to develop this software due to their advantages in terms of speed, security, cost efficiency and scalability compared to other languages.

## 1.3 References

**Books:**

* Designing data- Intensive Applications by Martin Kleppmann
* The Algorithm Design Manual (Handcover) by Steven S. Skiena
* Software Requirements and Specifications: A Lexicon of Practice, Principles and
* Software Requirements (Microsoft) Second Edition by Karl E. Wiegers
* Software Engineering: A Practitioner’s Approach Fifth Edition by Roger S. Pressman

**Websites:**

* <https://www.softwareadvice.com/>
* <https://theappsolutions.com/blog/development/how-to-develop-social-media-app/>
* https://www.sciencedirect.com/book/9781558608436/designing-data-intensive-web-applications

# 2. System-Wide Functional Requirements

[Statement of system-wide functional requirements, not expressed as use cases. Examples include auditing, authentication, printing, reporting.]

* Account creation and option for the user to decide the privacy level for the shared story
* Sharing/reading/commenting on stories
* People comment get point or reduce a point based on how helpful the comment is
* Online consultation booking
* Access to professional help
* Type of support based on the issue (Stress, Trauma, Trust issue, etc.)
* Professional article/ Info Hub

# 3. System Qualities

## 3.1 Usability

## The application should prioritize ease of use. Naaman is an application that depend on the user opening themselves and sharing their issues. This means that the user interface must be simple to use and provide consumers with a relaxing and pleasant experience. The mental health specialists and clients will utilize the suggested system that we are developing. As a result, the system's functions must conform to the criteria established by the users.

## 3.2 Reliability

The application needs to have a minimal downtime because the application is tackling a vulnerable issue. People with a low mental health have a hard time opening themselves and are fragile. Any kind of interruption can led to an unfortunate circumstances.

## 3.3 Performance

The application needs to have a quick response time and startup as those small windows of time can result in the user changing their mind from consulting. On the other hand, the application only needs a regular capacity as there won’t be a massive concurrent users and limited expert for online consultations.

* The system's performance should be quick and precise.
* This system will manage expected and unforeseen failures in a way that prevents data loss and extended periods of downtime. As a result, it should include built-in error checking to detect incorrect usernames and passwords.
* The system should accommodate a high number of transactions data without any fault. So, it should be able to handle large data amount.
* **Response Time** - Average response time shall be less than 2 second.
* **Throughput** - The system shall accommodate transactions data per minute.
* **Recovery Time** - In case of a system failure, redundant systems shall resume operations within 30 seconds. Average repair time shall be less than 1 hour.
* **Start-up/Shutdown** Time - The system shall be operational within 1 minute of starting-up.
* **Capacity** - The system accommodates 10000 concurrent users.
* **Utilization of Resources** - A maximum of one million Transactions may be stored in the database by the system. Old transactions must be backed up and removed from the operational database if the database expands over this limit.

## 3.4 Supportability

Naaman is a web-based and need to be compatible to browser for laptop and mobile phone. Due to the constant prolonged lockdown and social distancing, Naaman need to be easy to upgrade to prepare for more user.

In addition, the system should be designed to be expandable. It should be simple to add new feature needs or accommodate changes to current ones.

# 4. System Interfaces

There are many types of interfaces as such supported by this remote mental health management platform namely; User interface, software interface, hardware interface and communication interface.

## 4.1 User Interfaces

The user interface for the software should be compatible to any browser such as Internet explorer, Mozilla or chrome.

### **4.1.1** **Look & Feel**

The user interfaces should be simple and give a soothing/comforting feeling.

The system will ensure that all web pages have a consistent appearance and feel with using of icons and toolbars

### **4.1.2** **Layout and Navigation Requirements**

Navigation bar at the top of the website. Navigation bar should embrace predictability. Also navigation bar should have a clear hierarchical structure with every category and subcategory. Also navigation bar should contain a home button which can redirect users to the home page.

The user interface is available providing following functionalities to make it easy to access .

* Application will have menus and navigation bars in all pages at the same position.
* In application the logo should be appeared in each page.
* Search boxes should be appeared.
* Application should have an option to get help.

### **4.1.3** **Consistency**

This application should

* Gives a positive experience to user.
* Enables users to do their tasks in an efficient and quick way.
* Improve the website usability and learnability.
* Have an option to get help.
* Use the shades of the same color pallet. It shouldn’t be containing much varieties of colors.

### **4.1.4** **User Personalization & Customization Requirements**

As Consultants and clients are the system's users, members are expected to have a basic understanding of computers and how to use the internet. The person in charge of the program should have a better understanding of the system and how to repair any minor issues that may arise as a result of disk crashes, power outages, or other factors.

## 4.2 Interfaces to External Systems or Devices

### **4.2.1** **Software Interfaces**

* Front end client :
* Database server :
* Back End :
* Operating system : Windows XP or above
* Browser : Any latest browser

### **4.2.2** **Hardware Interfaces**

Since this remote mental health management platform must run over the internet, all the hardware which require to connect the internet will be hardware interface for the system. As for e.g. Modem, WAN-LAN, Ethernet Cross–Cable.

Software

* Processor: Pentium(R)Dual-core CPU or higher
* Hard Disk: 240GB or more
* RAM: 4GB or more

### Mobile

* Mobile App • Processor: 64-bit CPU with at least 4x Arm®
* Hard Disk: 16GB or more
* RAM: 2GB or more

### **4.2.3** **Communications Interfaces**

Because this remote mental health management platform must operate over the internet, all of the hardware required to connect to the internet will serve as the system's hardware interface. For the suitable use there must be a good internet connection among users. It will use the HTTP protocol for communication over the internet. And the intranet communication will be through TCP/IP protocol suite.

# 5. Business Rules

Anything that takes and fulfills company policies and procedures is referred to as a business rule. A rule can be used to enforce company policy, make a decision, or infer new information from current data. This contains the rules and restrictions that users of the System must follow. This covers the project's cost as well as any discounts offered. Illegal protocols should be avoided by users.

* All users need a valid email address to sign up. This can be considered as valid if the user can prove that they have access to the email address.
* User’s sign-in and sign-up times must be tracked.
* If the submit button pressed on some form or document n the application and all the mandatory fields are present, the system should send a confirmation email to the user.

# 6. System Constraints

As this is a mental health management platform certain boundaries must be maintained accessing client’s information. Most of these are center on maintaining the privacy of people. Also the software must be able to store data indefinitely if there is any consequences. If there is huge damage to the database due to the virus, or catastrophic failure, such as a disk crash. The recovery method should restore a past copy of the database that was backed up to archival storage and reconstructs a more current state by reapplying or redoing the operations of committed transactions from the backed-up log, up to the time of failure. Payment gateways should be secured

# 7. System Compliance

## 7.1 Licensing Requirements

A software license is a document that grants the right to one or more copies of a software tool to a customer. Using and disseminating the program without a license would be deemed a copyright violation.

To execute software, valid licenses are required to:

## Work with application development software

## In development, test, and production mode, to utilize web server, application server, and database software

These are the license required to the “Naaman” mental health management platform;

**Perpetual software license**

A perpetual software license is a license to use software that does not expire. The consumer is not required to pay for support or to upgrade the software version they purchased.

Perpetual licenses are preferred by customers because they are straightforward and easy to administer. They may, however, receive obsolete tools, leaving them dissatisfied and resulting in a negative reputation for the service.

**Subscription software license**

A subscription, on the other hand, is a license that is renewed each year. Customers usually renew their licenses every year, which includes support and updates for the duration of the coverage term. The license will be automatically cancelled unless the client renews it. Subscription licenses provide a constant stream of revenue for the publisher rather than a lump sum payment, which also means a lower initial cost for the consumer. A connection is formed between the customer and the supplier as a result of this continual engagement, making it simpler for the provider to anticipate their requirements.

**Consumptive software license**

There is a recurring charge for the consumptive license as well, however it is dependent on consumption. The client pays more the more they utilize the product.

The payment mechanism for a consumptive license is the one that is most closely connected to the value the consumer received from the product while simultaneously providing a reasonable revenue stream for the supplier. However, one disadvantage is that more management is necessary on both ends.

## 7.2 Legal, Copyright, and Other Notices

Naaman should be offer the disclaimers, copyright, word mark, trademark and product warranties of the [Organization name].

When original copyright holders donate source code, documentation, and other content to this project, the copyrights in those contributions generally remain with the original copyright holders.

So, it should be

* Copyright The “Naaman” Authors.
* Copyright The “Naaman” Contributors.
* Copyright Contributors to the “Naaman” project.

These statements are meant to convey the following information:

* The work is copyrighted;
* The contributors of the code licensed it, but retain ownership of their copyrights; and
* The project was licensed for distribution as part of the named project.

Also there should be

* Limitation of warranty
* Permission to reproduce manuals

## 7.3 Applicable Standards

This project should follow these standards.

* Clear definition of purpose
* Simplicity of use
* Ruggedness (difficult to misuse, kind to errors encountered)
* Delivered on time and when needed
* Reliability
* Efficiency (fast enough for the purpose it was created)
* Minimum development cost
* Conform to standards
* Clear, accurate and precise user documentation
* Clear, accurate and precise technical documentation.
* Development standards for all stages of the System Development Life Cycle
* Minimum software development activities, deliverables, and acceptance sign-off criteria

# 8. System Documentation

The user interface, user manual, online assistance, and installation and management guide must all be sufficient to teach users on how to use the system without difficulty. Users should be provided with certain amenities by the administrator in the form of,

* Backup and recovery
* Forgot password
* User registration
* Data replication
* Updating the server
* Maintaining files and report

**Remote Mental Health Management**

**“Naaman”**

Architecture Notebook

There is guidance within this template that appears in a style named InfoBlue. This style has a hidden font attribute that allows you to toggle whether it is visible or hidden in this template. Use the Microsoft® Word® menu **Tools > Options > View > Hidden Text** check box to toggle this setting. There is also an option for printing: **Tools > Options > Print**.

# Purpose

The purpose of this document is to describe in detail the architectural goals and philosophy, assumptions and dependencies, architectural requirements, decisions, constraints, architectural mechanisms, key abstractions, architectural framework, and views that will act as a blueprint for the design and implementation of project “Naaman”.

# Architectural goals and philosophy

Project “Naaman” is about creating a safe and comforting social media environment for mental health management. It is a platform where people are able to share their stories, struggles or what they are going through and get professional help from trained mental health professionals and medical professionals.

Due to the system being heavily regarding mental health management, it has to be reliable, with no server downtime, user friendly with good navigation design, and have good performance with fast loading times. The user interface has to be easily understandable with no corner page. A backend interface is necessary to provide the backend programmers easy access to do updates and maintenance.

* **Reliable** - because the system is about health management, it must be reliable. People will be using the system to share personal information and contact mental health professionals or medical professionals, because of this function the backend server has to be reliable to avoid losing user personal data, appointment schedule information. In cases of natural causes out of our control such as disastrous weather conditions, there must be back up generators to keep the backend server running during such events.
* **User friendly** - the user interface must be easy to understand and give a welcoming feeling to users. It should not look rigid and complicated. Users should be able to adjust to using the system quickly, not get confused and stressed at first glance. The system should have good navigation, where users are able to understand where to go easily depending on what they are looking for. Such as a big drop down menu for sharing stories or making appointments. Lastly, there must not be corner pages, which are pages where there is no “back” navigation. All interfaces should have an option to go back, or an option to go to the main page. Users should not feel stuck while using the system.

**Fast performance** - the system should have fast performance. It should have fast load times and fast response times. Users should not be stuck waiting for the system to load. It should run smoothly and be dynamic.

# Assumptions and dependencies

|  |  |
| --- | --- |
| **Assumption and Dependencies** | **Descriptions** |
| Internet | Naaman is a web-based project, therefore it is necessary to obtain a domain name, as well as web hosting services. |
| Client base | Our main client base are people experiencing difficult times, having mental challenges, stress, loneliness etc. |
| Server | A server is necessary to keep the system running. |
| Database server | A database server is necessary to document and maintain sensitive user information. |
| **Project team** | **Contact information** |
| Sobana Handi Achini Thisarangi De Silva | achinithisarangi86@gmail.com |
| Dehemi Vihara Dissanayake Liyanage | dissanayake7793@gmail.com |
| Gaury Chethana Thanthirigama | chethana9804@gmail.com |
| Temuulen Tsengel | temuulen2830@gmail.com |
| Michael Weisang | weisangm1@gmail.com |

# Architecturally significant requirements

“Naaman” will be a dynamic website that will have a reliable server with backup generators in case of power outages or natural disasters, user friendly and robust user interface with fast response and load times.

A backend interface is necessary to ensure server maintenance is up to date. Programmers working on the website after development finishes need an interface to make future maintenance and updates of the website efficient.

Database server is essential to maintain and document essentially everything that happens on the website. Including user information, appointments or booking details, and version control of regulations.

# Architectural Mechanisms, Constraints, and Justifications

|  |  |  |
| --- | --- | --- |
| **Architectural Mechanisms** | **Constraints** | **Justifications** |
| Reliability | Natural causes (weather disasters) | We need the system to be reliable to ensure no loss of data occurs. |
| User Friendliness | Clustered or confusing interface design | We need a user friendly user interface to allow users to easily use the system |
| Fast Performance | Internet speed | We need fast performance to avoid user frustration over slow loading times |

# Key abstractions

* User - People who are going through stressful times or experiencing difficulties
* Remote Mental Health Management system - User interface of website “Naaman”
* Server - Backend server which runs the website
* Database - Server to maintain and document user sensitive data
* Backend interface - Interface for programmers to update and maintain the system
* Mental health professionals and medical professionals

# Layers or architectural framework

This system will be developed using the three-tier architecture, which are client layer, server layer, database layer. By using the three-tier architecture we ensure each aspect of the system is addressed and covered in development.

**Client layer** - covers user interface. The user interface should be user friendly with good navigation. Client layer covers all aspects of user related decisions.

**Server layer** - covers server requirements.The server must be reliable to ensure the system is always available. In cases of unpredictable disastrous situations, a backup generator must be there to keep the server running.

**Database layer** - covers database server. The database server will contain the client’s sensitive information, it must have good security. The backend interface should cover all aspects of updating and maintaining the system.

# Architectural views

* Account creation - Users are able to create an account. Once they do they have the option to decide the privacy level of stories they share.
* Sharing stories - Users are able to read and comment on other stories.
* Comments - Users are able to give points or reduce points on comments depending on how helpful it was or not
* Online appointments - Users are able to make appointments with mental health professionals
* Access to professional help - Users have access to professional help in many ways (professional advice posts or booking a session with professionals).
* Support based on types - Users have the option to choose what kind of help they need based on what they are going through (symptoms).
* Peer reviewed articles - Professional articles are available.

**Risk List**

****

**Master Test Plan**

****

Naaman

Project Plan

# Introduction

This software development plan is essentially a project timetable. It provides the start and finish dates for each phase of the software project, as well as the procedures necessary to accomplish each phase of work. This plan consists of the information about the project organization, project practices and measurements, project deployment and project milestones and objectives.

Project Organization part describe the team members, their roles in the project and also their responsibilities. The project organization is a framework that makes it easier to plan and execute project operations. Its major goal is to establish an environment that encourages team members to engage with each other with the least amount of disruptions, overlaps, and conflict as possible. The effectiveness of the plans is measured using project practices and measurements. Cost and schedule variables are examples of project output variables, while scope, budget, schedule, and safety performance are examples of project outcome measures. The deployment part contains deployment strategies and procedures for changing or upgrading an application's operating instance. The last part of this project plan is the project milestones and objectives. This part defined the moment in the life cycle of a project that is used to track progress toward the end objective. Milestones are used in project management to signify the start or conclusion of a project, external evaluations or input, budget checks, the submission of a significant deliverable, and so on.

# Project organization

|  |  |  |
| --- | --- | --- |
| Team member | Project Role | Responsibility |
| Sobana Handi Achini Thisarangi De Silva | Completer | Pay attention to details.  Make sure that the job is completed before the deadline |
| Dehemi Vihara Dissanayake Liyanage | Resource investigator | Explore opportunities and contacts.  Negotiate for resources |
| Gaury Chethana Thanthirigama | Shaper | Drive team forward.  Determine the most effective methods for solving issues. |
| Temuulen Tsengel | Coordinator | Clarify goals, promote decision making.  Understand the importance of each team member's contribution. |
| Micheal Weisang | Leader | Provide support and encourage to the corporation.  Managing the day-to-day activities of the team |

# Project practices and measurements

Management practices

* Define roles and responsibilities of team members - Roles of team members has been well defined for productivity.
* Define meeting types - Meetings are very important and should be defined in detail (meeting participants, context, average duration etc.).
* Define documentation strategy – The application requires a documentation strategy (backlogs, lessons learned info, bug items etc.) for consistency, understandability, and more successful software creation.
* Define and track software quality measurements - Some measurement standards should be chosen to monitor code quality and growth based on the attributes of the program type.
* Keep track of issues and requirements – Requirement management define the software's scope and provide a traceable foundation for functional testing. Those issues/requirements are better kept and managed with tools that offer greater functionality than text editing.

Technical practices

* Create a Minimum Viable Product (MVP) - This will aid in the rapid and low-cost deployment of applications.
* Keep the code as simple and consistent as possible - When it comes time to update, simplicity and a consistent coding style will make things a lot easier for the development team.
* Test continuously - Continuous end-to-end testing will increase the trust in the quality of your code.
* Use a variety of resources to double-check the code - This helps developers detect issues early in the development process and allows them to learn from one another and improve their coding abilities.
* Make realistic time and budget projections - Stress is exacerbated by short deadlines and limited funds. On the other hand, too much time may encourage developers to postpone, and too large a budget may lead to waste.

Measurements

* Quality of the code - For premium software, bug-free and semantically accurate code is critical. Quantitative and qualitative metrics can be used to classify code quality requirements.
* Performance - Software security metrics assess a software program's intrinsic security and verify that no unauthorized alterations are made to the product before it is delivered to the customer.
* Usability - Because all software products are designed for end users, usability and user-friendliness are key quality indicators.

# Deployment

* Recreate - This method results in service downtime that is dependent on the application's shutdown and boot times. Advantages of this process are it will be easier to setup and application state will be renewed.
* Ramped - The ramping deployment technique is gradually deploying a version of an application by replacing instances one by one until all of them have been deployed. To enhance the deployment time, we can change the following parameters: parallelism, max surge, max unavailable

Advantages are the limited number of users have been given access to the new version, easy to keep track of the mistake rate and performance and Quick reversal.

* A/B testing - A/B testing deployments include sending a portion of people to a new feature under controlled settings. Advantages of this process are several versions can be running at the same time and can get complete control over the flow of traffic.

# Project milestones and objectives

Table

Description automatically generated

Table

Description automatically generated

**Technical Competency Demonstrator**

****

# Inception Phase Status Assessment

## 1. Assessment against Objectives of the Inception Phase

### 1.1 Do we know what we are trying to achieve?

The aim of the project is to minimize mental health issues. This is embodied in the completed Vision Document.

[https://interact2.csu.edu.au/courses/1/S-ITC303\_202160\_SM\_I/groups/\_68438\_1//\_1173827\_1/A2\_Vision%20Document.docx](https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1/_1173827_1/A2_Vision%20Document.docx)

We understand the main functional requirements of the project which are:

* Share their stories
* Read other people's stories and communicate with them
* Online Consultation
* User Registration

This is shown in the completed Use Case model embodied in “Use Case Specification”.

[https://interact2.csu.edu.au/courses/1/S-ITC303\_202160\_SM\_I/groups/\_68438\_1//\_1173828\_1/Full%20Use%20Case%20Description.docx](https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1/_1173828_1/Full%20Use%20Case%20Description.docx)

We understand the main Non-Functional requirements of the project which are:

* Ease of use
* User interfaces provide a nice gentle comforting feeling
* The physicians can access user information assigned to them
* Can be accessed anywhere using device with supported browser
* Mobile number authentication

This is shown in the completed Non-Functional Requirement model embodied in Non-Functional Requirement Specification

[https://interact2.csu.edu.au/courses/1/S-ITC303\_202160\_SM\_I/groups/\_68438\_1//\_1173830\_1/NFR.docx](https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1/_1173830_1/NFR.docx)

### 1.2 Do we know how we are going to achieve it?

We have a good idea of how we are going to achieve our aims. We are going to use a three-tier architecture. This is shown in the completed Architectural Notebook.

[https://interact2.csu.edu.au/courses/1/S-ITC303\_202160\_SM\_I/groups/\_68438\_1//\_1173837\_1/Architecture%20Notebook.docx](https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1/_1173837_1/Architecture%20Notebook.docx)

We have a good understanding of the project specific risks facing our project and how we are going to deal with them. The risks are:

* Legal risk
* Work overload
* Incompetence
* Deadline

Our evolving understanding of risks is shown in the ongoing risk list shown in the initial “risk management”.

<https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1//_1173838_1/Risk%20Management.xlsx>

We have a good understanding of how we are going to check that our application delivers the intended functionality and system properties. Our key areas of concern and the test strategies we will use to address these concerns are as follows:

* User interface
* User experience
* Accessibility management
* Basic feature

This is shown in the Initial Master Test Plan.

[https://interact2.csu.edu.au/courses/1/S-ITC303\_202160\_SM\_I/groups/\_68438\_1//\_1173876\_1/Master%20Test%20Plan.doc](https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1/_1173876_1/Master%20Test%20Plan.doc)

We have a good understanding of the dependencies and likely completion times for different parts of the project. Target completion dates for key aspects of the project are as follows:

Table

Description automatically generated

Table

Description automatically generated

This is shown in the Initial Project Plan.

***Please maintain flow of information***

### 1.3 Skills required

Our project requires skills using the following key tools and technologies:

* PHP <server side>

For managing business logic <Authentication>, PHP will be used

* MySQL <Database>

For Data tier, MySQL database server will be used.

* JavaScript <client side>
* HTML

We have demonstrated that we have the skills to use these technologies through the implementation of a technology competency demonstrator.

[https://interact2.csu.edu.au/courses/1/S-ITC303\_202160\_SM\_I/groups/\_68438\_1//\_1173839\_1/Simple%20HTML%20Website.zip](https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1/_1173839_1/Simple%20HTML%20Website.zip)

## 2. Deliverables

### **2.1 Vision Document.**

* Project Introduction
* Positioning
* Stakeholder Description
* Product Overview
* Other Product Requirement

[https://interact2.csu.edu.au/courses/1/S-ITC303\_202160\_SM\_I/groups/\_68438\_1//\_1173827\_1/A2\_Vision%20Document.docx](https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1/_1173827_1/A2_Vision%20Document.docx)

No Issue

### **2.2 Architectural Notebook**

* Architectural Goals and Philosophy
* Assumptions and Dependencies
* Architectural Mechanisms, Constrains, and justifications
* Architectural Framework
* Architectural View

[https://interact2.csu.edu.au/courses/1/S-ITC303\_202160\_SM\_I/groups/\_68438\_1//\_1173837\_1/Architecture%20Notebook.docx](https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1/_1173837_1/Architecture%20Notebook.docx)

No Issue

### **2.3 Risk Management**

* Risk List
* Risk Response Strategy

<https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1//_1173838_1/Risk%20Management.xlsx>

No Issue

### **2.4 Use Case Documentation**

* Use Case Description
* Use Case Diagram

[https://interact2.csu.edu.au/courses/1/S-ITC303\_202160\_SM\_I/groups/\_68438\_1//\_1173828\_1/Full%20Use%20Case%20Description.docx](https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1/_1173828_1/Full%20Use%20Case%20Description.docx)

No Issue

### **2.5 Non-Functional Requirement Specification**

* System-Wide Requirement Specification
* NFR Specification (Usability, Reliability, Performance, Supportability)
* System Interface
* Business Rule
* System Constraints
* System Compliance
* System Documentation

<https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1//_1173977_1/NFR.docx>

No Issue

### **2.6 Master Test Plan**

* Assignment Formulation
* Documentation
* Test Strategy
* Approaches
* Organization
* Infrastructure
* Management
* Test Process Risks and Countermeasures

[https://interact2.csu.edu.au/courses/1/S-ITC303\_202160\_SM\_I/groups/\_68438\_1//\_1173876\_1/Master%20Test%20Plan.doc](https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1/_1173876_1/Master%20Test%20Plan.doc)

At the time of this document creation, the master test plan cannot be fully developed as there is some part of the project that has not been clearly defined.

### **2.7 Initial Project Plan**

* Project organization
* Project Practice and Measurements
* Deployment
* Project Milestones and Objectives

<https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1//_1174030_1/Project%20Plan.docx>

No Issues

## 2.8 Domain Class Diagram

[https://interact2.csu.edu.au/courses/1/S-ITC303\_202160\_SM\_I/groups/\_68438\_1//\_1173965\_1/Basic%20Domain%20Class%20Diagram.PNG](https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1/_1173965_1/Basic%20Domain%20Class%20Diagram.PNG)

## 3. General Issues

### 3.1 Time

Group 2 “Runtime Terror” has a diverse availability time, making it hard to make time for group discussions.

### 3.2 Deliverables

In the process of creating the deliverables and assigning tasks between each member, there are some deliverables that are almost forgotten.

### 3.3 Master Test Plan

Designing the master test plan have proved to be difficult with some part of the project that have not been properly defined

## 4 Summary – Overall Project Progress

 So far, the project is starting to move forward and every member already built a decent idea or mental image on what the project is and the aim of the project. The project is not envisioned to be beyond our capabilities and is within a reasonable time to be developed.