**Life Cycle Architecture Milestones**

**Resubmission**

Group 3:

* Temuulen Tsengel
* Chhiring Moktan
* Roshan Saud

Deliverables:

1. Project Vision
2. Domain Class Model
3. ERD Diagram
4. Use Case Description and model
5. Non-Functional Requirement Specification
6. Architectural Notebook
7. Risk List
8. Master Test Plan
9. Homepage
10. Registration page
11. Login page
12. UAT
13. Project Plan
14. Inception Phase Status Assessment

Remote Mental Health Management

“Naaman” Vision

# Introduction

During numerous Covid-19 lockdowns, several social and economic concerns have emerged. One of the biggest being the lack of social interaction between people. Human beings are social in nature, meaning they must interact with each other on regular basis in order to keep their mental health in check. Even though internet and social media exist, it is not enough for most people. This social issue has affected the mental health of people negatively.

Mental health is one of the most essential factors of people functioning. It affects how people think, feel, and how they act. Good mental health is the most important part of coping with stress, interacting with others, and making choices. Low mental health can lead to detrimental things. For example, having mood swings, low energy, lack of motivation, binge eating, self-harm, or in the worst-case scenario it could lead to even suicide.

To address this concern, Sydney Hospital has decided to develop a web-based social media platform and has assigned this task to Group 3. The project will be called Project "Naaman". The name “Naaman” came from a story from the bible. “Naaman” was an army commander who had developed leprosy disease. He was sent to a great prophet named Elisha. Where he expected himself to be blessed and cured of his disease. However, the prophet only sent a messenger telling Naaman to wash in the Jordan river instead of blessing and curing him of the disease. Therefore, he was angry and dejected that he didn't get the miraculous blessing or some sort of ritual to cure him. In the end, he was convinced by his companions to wash in the Jordan river as it was just a simple action. Naaman was cured after washing in the river. The moral of this story was that instead of waiting for some miracle to happen, sometimes a simple action to help yourself is all you need.

Project Naaman’s main goal is to help improve people’s mental health in these difficult times by providing a social media platform where people can share their stories and get online consulting from mental health professionals. It provides a convenient and secure environment where people can share and address their mental health doubts and concerns. Since mental health of people is of private and sensitive nature, Naaman will provide an anonymous platform so that people are able to freely express their doubts and concerns without the fear of being judged by relatives or friends when using the platform.

# Positioning

## Problem Statement

## Table Description automatically generated

Lockdowns and social distancing have caused people’s mental health to start declining due to lack of social interaction. These have caused people to develop lack of motivation and mood swings which may lead to decreased productivity and self-harming tendencies. In order to solve these issues, people need to address their frustrations or concerns they have without the fear of facing judgement or backlash and receive professional help to deal with their problems.

## Product Position Statement

Table

Description automatically generated

Project Naaman was created in order to provide a social media platform to help people who are experiencing mental health issues. The platform will provide a safe space for people to vent and share their stories and have access to professional help. Compared to other social medias, Naaman is providing an anonymous feature due to the private and sensitive nature of mental health. People can share their stories about their doubts and concerns without being afraid of getting judged by others.

# Stakeholder Descriptions

## Stakeholder Summary

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Responsibilities** |
| Sponsor | Sydney Hospital | Sponsoring the project |
| Development team | ITC309 Group 3 | Develop the application |
| Senior leader | Ather Saeed | Provide guidance and assistance |
| Physician | Mental health professional | Providing online consultations and help in resolving mental health issues |
| User | The social media platform user | Registering and receiving mental health support |
| Helpdesk Support | Employee | First help support and managing online booking |

## User Environment

Due to the nature of mental health management being private and sensitive, the user interface will be focused on a single user usage only. The user can take their time and go at their own pace on using the application because expressing their thoughts may not be easy to do so. However, this will only be true for the social media aspect of the application. For the online consultation aspect of the application, users are given specific time ranges to communicate with the physicians. Users can access the application through the browser on any device at any time.

In order to protect the users and provide privacy, users are required to create an account with password and authentication to prove their identity. Users are also able to change the privacy level of their information that are available in public to others. If setting is set to low privacy, their information is available to all others on the application. If setting is set to high privacy, their information is only available to their assigned physicians.

The interface physicians use will be different from the user’s interface. To have access to the physician interface, the physicians are required to create an account, provide proof of their profession, and be authorized as a mental health provider by the admin. This interface will include a patient managing function to help organize their schedule.

# Product Overview

## Needs and Features

Table

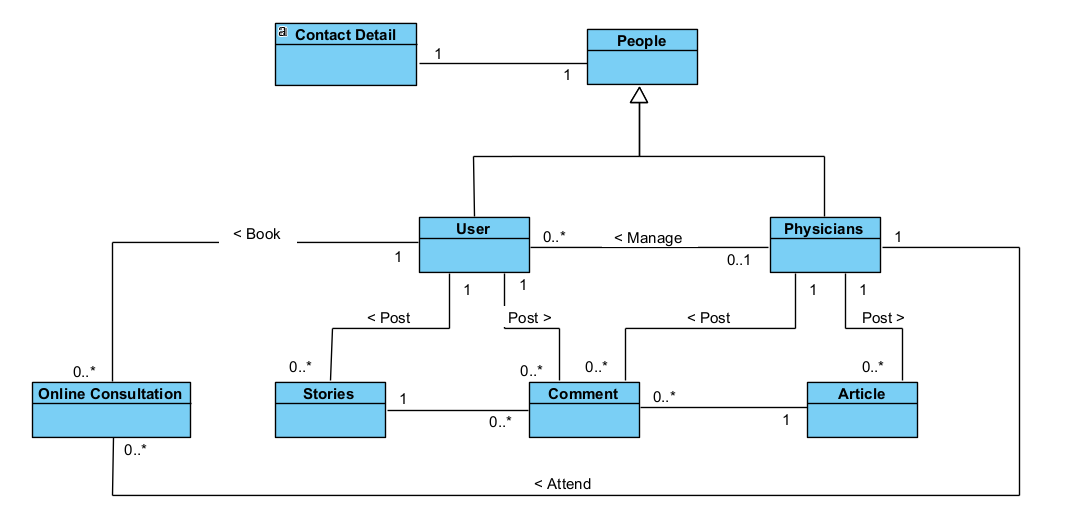
Description automatically generated

# Other Product Requirements

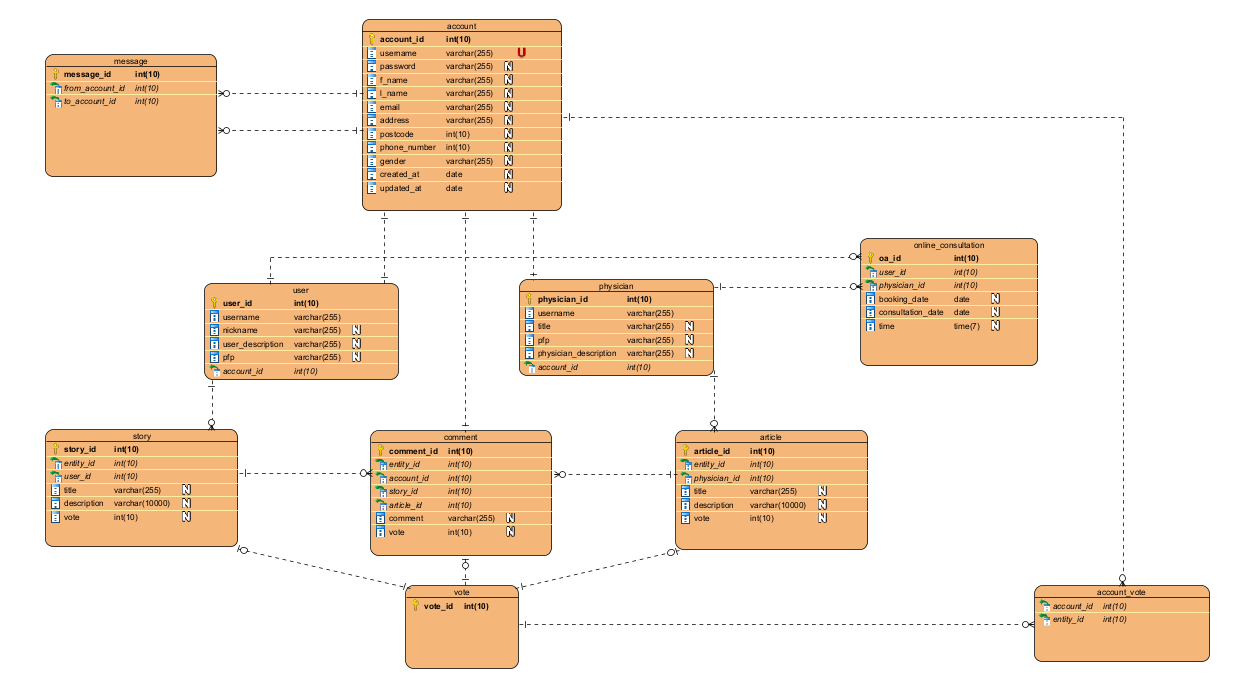
Table

Description automatically generated

Domain Class Model



ERD diagram



# Mental Health Management system (Naaman) public page

Diagram

Description automatically generated

Use case description

The above figure shows the public page of Project Naaman remote mental health management system. The main functions of the mental health management public page:

* The user can visit the mental health management system
* Register or log in to the system
* Post articles and share stories

Public page must be a news feed page so new users are able to get an overall understating of the system. The moderator can log into the system and control user behaviours. The selected web project is an open public project. Since the platform will be available to everyone, the moderator is necessary to control and mitigate inappropriate behaviour by users. Additionally, posts made by new users will be held automatically for a short amount of time until they are reviewed by the moderator and given permission to publish.

# Use-Case: <Access the web >

**6.1 Brief Description**

This use case allows access to the Remote Mental Health Management system public webpage with selected strong and valuable article and stores for the user. (Mental health management system public website must be with post sharing page)

**6.2 Actors**

**6.2.1 User**

**6.2.2 System**

**6.3. Pre-Conditions**

* + The user must be a new user or have not logged in to the system previously
  + User using browser to use Mental health management system must accept cookies
  + Honeypots and Website must be online. Server-side honeypots are a security method which is used by administrators for preventing attacks, avoid hacking, and identify malicious activities.

**6.4 Normal Flow**

The use case begins when user accesses web browser

|  |  |
| --- | --- |
| Actor | System |
| User makes request to the Remote Mental Health Management system over web browser | 1. Redirect the original web hosting web server after checking the user request status (check the security issue and redirect) 2. Check the request and confirm user status (if user has not logged into the session) 3. Create and send the related content over the website for the user. |

The use case ends.

**6.5 Alternate Flows**

If the user registers 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. 3. Load the main page in Mental health management system (with the selected post) |

**6.6 Sub flows**

Request with wrong web address

|  |  |
| --- | --- |
| Actor | System |
| if user request is not correct | 1. Redirect the original web hosting web server after checking the user request status (check the security issue and redirect) 2. Check the request and confirm user status (if user has not logged into the session) or user browser hasn’t any Mental health management system cookies. 3. If main domain is correct, redirect website main public page (home page with articles) 4. Create and send the related content over the website for the user. |

**6.7 Key Scenarios**

* New and registered users visiting the Mental health management system must have accessed public web page first
* The admin panel in this project will decide considering about web site security therefore original web server IP address must be hidden
* Configuring the honeypot and user access to the system within the honeypot can prevent unauthorized access
* The moderators can check the Mental health management system website security issues
* The moderators can control the user behaviours and control the website
* If user is new, system generates website content using their geo location data
* If user is currently registered, use their past data for generating the website content

**6.8 post-conditions**

* Add new record into the user log with login IP address
* Create a cookie for the user

**6.9 Special Requirements**

* Minimum loading time
* Mental Health Management System Security
* UI and UX
* Language support

# Use-Case: < Register New User>

**7.1 Brief Description**

This use case allows new user to register into the Mental Health Management system. In this stage system will check the user’s browser cookies because if browser has any cookies related to the Mental health management system automatically login due to user’s cookies providing their past information with the website.

**7.2 Actors**

7.2.1 User

7.2.2 System

**7.3 Pre-Conditions**

* The user must be new user
  + Website must be online
  + User’s browser does not have Mental health management system’s cookies

**7.4 Normal Flow**

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

|  |  |
| --- | --- |
| Actor | System |
| * 1. User requests the Remote Mental Health Management system over the web browser | * 1. Redirect the original web hosting web server after checking the user request status (check the security issue and redirect)   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. |
| * 1. User requests new registration | * 1. The system generates a register form and an image with 4 random rotated letters. |
| * 1. 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. | * 1. The system validates the user data and stores the user data   2. System send email to the user with register confirmation |
| * 1. User confirms the email | * 1. System creates new login for the requested user |

The use case ends.

**7.5 Alternate Flows**

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

|  |  |
| --- | --- |
| Actor | System |
| **1.** User requests 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.** Check the bowser cookies related to the Mental health management system |
| **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  **8.** System pull the cookies to the user’s web browser for using next time autologin. |

**7.6 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) |

**7.7 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 request when the user tries to perform a task on the system without signing in if user registered Mental health management system and user doesn’t login to the Mental health management system will be check the browser’s cookies and if bowser has any cookies related to the Mental health management system will be autologin as a login user and giving the permission to the user for post and comment.

**7.8 post-conditions**

* Store new user details and new login details
* Start new temporary session for the new user and update home page
* Pull new cookies to the user’s browser

**7.9 Special Requirements**

* Minimum loading time
* Mental Health Management System Security
* UI and UX
* Language support

# Use-Case: < User Login>

**8.1 Brief Description**

This use case allows the user to login into the Mental Health Management system. If user tries to login from user’s previous device and it has some cookies related to the Mental health management system, system automatically login to the system as cookies related user. (Cookie’s expiry dates also check in this stage)

**8.2 Actors**

8.2.1 User

8.2.2 System

**8.3 Pre-Conditions**

8.3.1 The user must be register user

8.3.2 Website must be online

**8.4 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.** Redirect the original web hosting web server after checking the user request status (check the security issue and redirect)  **3.** Check the request and confirm user status (if user has not logged session)  **4.** Check the bowser cookies related to the Mental health management system  **5.** 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. I**f user-entered login details correct, system reload the page and set the user session. |

The use case ends.

**8.5 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 | **4.** when user login attempt increases more than 2 times, Mental health management system suggests reset password to the user |

**8.6 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 |

**8.7 Key Scenarios**

There are two main types of user login process. First one is user requests for the login process and the other is the system requests login process for the user. The system makes such request when the user tries to perform a task on the system without sign in. system checks the user session-log and identified registered-user and check the user’s current login device’s browser cookies and its status then if that status is matching the user login details Mental health management system redirect the user to public web page with login. If there don’t have any matching user login session or cookies system ask logging details from the user.

**8.8 post-conditions**

* Add record for the log-details
* Start new temporary session for the new user and update home page
* Start user track system for monitoring user behaviours
* pull the new user cookies to the user devise if that device does not have any valid Mental health management system cookies.

**8.9 Special Requirements**

* Minimum loading time
* Mental Health Management System Security
* UI and UX
* Language support

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

**9.1 Brief Description**

This use case allows to post an article, post a story, or comment an article of the Mental Health Management system. This task is main features in Mental health management system public page. User should be login to the Mental health management system before accessing this task because system need to identify the exact user.

**9.2 Actors**

2.1 User

* 1. System

**9.3 Pre-Conditions**

3.1The user must be a registered user

3.2 Website must be online

**9.4 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, or if user have valid cookies 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.

**9.5 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. This public page allows the user for experience sharing with others and adding some value for others experience. (Example comment) in this stage as a developer, developers can’t predict all public users are genuine therefore Mental health management system should have some restrictions. (Example; holding post, tracking user behaviours and enhance security using hide original web server IP address.

**9.6 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.

**9.7 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 >

**10.1 Brief Description**

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

**10.2 Actors**

2.1 Moderator

* 1. System

**10.3 Pre-Conditions**

3.1 The moderator must be login

3.2 Website must be online

**10.4 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.

**10.5 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.

**10.6 post-conditions**

6.1 add record for the log-details

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

**10.7 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. In this stage user can make appointment for meeting the specialist related to their problem type.

**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 and suggest the reliable specialist  **5** The system checks the specialist’s available date and time  **6** The system shows the available date and time for the user |
| **7** User select date and time | **8** the System request a confirmation from user. |
| **9** User give the confirmation | **10** 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 and explain system identify problem indirectly. |
| **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. | **5** the system checks and suggest the reliable specialist  **6** The system checks the specialist’s available date and time  **7** The system checks the available date and time  **8** The system shows the available date and time for the user |
| **9** User select date and time | **10** the System request a confirmation from user. |
| **11** User give the confirmation | **12** 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 | 1. The system stores theappointment as a draft appointment 2. **system send email about draft appointment** 3. **stem send user entered data to the** specialist because if user had critical problem, specialist must give more attention. |

**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 meeting a specialist. The system and moderators give some attention for the user and if some users behaviours are abnormal system directly suggest private appointment for meet the specialist and moderators do same thing using system. This process is just suggested and not recommend this function just perform for protect human life

**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 to 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 be ready to be 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

Graphical user interface, text, application, email

Description automatically generated

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generatedText, letter

Description automatically generated

Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

A screenshot of a computer

Description automatically generated

Users must rely on the system that the help option provides immediate response from the above navigation link because it can be urgent, and users may need to get support from psychiatrics more often if the user interface makes it easy for them to use the system.

Text

Description automatically generated

Maintaining consistency is crucial for a remote mental health web application because the users can be emotionally tired when they use the system. The eye-catching and friendly environment which is consistent through all pages will help calm down the system users' minds.

Text

Description automatically generated

Graphical user interface, text, application, letter, email

Description automatically generated

Since the application is based on the web, all the users ranging from physicians to day-to-day users must have a reliable internet connection in order to continue the processes of the system.

A screenshot of a computer

Description automatically generated with medium confidence

Text

Description automatically generated

Back-up systems must be maintained in several remote databases in order to secure data and recover easily.

Graphical user interface, text, application, letter

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence Graphical user interface, text, application, letter, email

Description automatically generated

Graphical user interface, text, application, letter, email

Description automatically generated

**Remote Mental Health Management**

**“Naaman”**

**Architecture Notebook**

# 1. 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”. This document’s focused audience is the stakeholders which include sponsors, the development team and anyone relevant to the Naaman project’s development.

This revised version for Architecture Notebook of project Naaman addresses the last two stages of the Unified Process model. Which are construction phase and transition phase. In these stages the main purpose is to develop and build the application. In the previous two phases of the Unified Process model the focus was to develop the concept of the project as well as design the specific requirements of the project deliverables. However, in the construction and transition phase, the focus is solely on developing and testing the application itself since the concept was already designed and documented in the inception and elaboration phases.

# 2. Architectural goals and philosophy

Project Naaman’s main goal and philosophy is to create a safe and convenient social media environment for mental health management. It is a platform where people can share their stories, doubts and concerns or what they are going through and get professional help from trained mental health professionals and medical professionals. The motivation behind project Naaman is to help people experiencing mental struggles and frustrations. Covid-19 has impacted people’s lives enormously in the past several years. Most people have lost their jobs, have been stuck at home for a long time or even lost people dear to them. While it is impossible to go and help every person who need help, it is possible to provide an online platform where they can begin to help themselves. Expressing their inner frustration or concerns can help people move on from those concerns in a healthy way.

Even without the effect covid-19 has had on people, taking care of your mental health is a shunned notion in many cultures. For example, parent’s usually do not realize their child misbehaving is a sign of them mentally struggling. Young adults are usually met with harsh criticism when they try to reach out to people about their mental health. Men are taught to be strong and not to cry when they go through difficult times and are seen as weak if they do.

Project Naaman’s is goal is to break these stereotypes of mental health management and provide an online platform for people to reach out and take care of their mental health.

Due to the system being heavily regarding mental health management, it must be reliable, with no server downtime, user friendly with good navigation design, and have good performance with fast loading times. The user interface must 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 must 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.

# 3. 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** |
| Temuulen Tsengel | temuulen2830@gmail.com |
| Chhiring Moktan | Moktanchhiring07@gmail.com |
| Roshan Saud | 74saudhero@gmail.com |

# 4. 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.

# 5. 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 |

# 6. Key abstractions

* User - People who are going through stressful times, experiencing difficulties, or getting counselling for mental illnesses such as:
  + Clinical depression – constant depressed mood, loss of interest in daily activities
  + Anxiety disorder – strong feelings of worry, anxiety or fear that disturb daily activities
  + OCD (obsessive compulsive disorder) – Unreasonable thoughts and fears that lead to repetitive behaviours
  + PTSD (post-traumatic stress disorder)
  + ADD (attention-deficit/hyperactivity disorder)
* Remote Mental Health Management system - Front end 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

# 7. 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. The biggest advantage of using three tier architecture is that each tier can be developed and modified simultaneously without affecting the others.

**Client layer** - covers user interface. The user interface should be user friendly with good navigation. Client layer covers all aspects of user related decisions. CodeIgniter (PHP) is an open-source software for building websites it is being used to develop the client layer.

**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. CodeIgniter (PHP) is an open-source software for building websites it is being used to develop the server layer.

**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. MySQL community edition is being used to develop the database layer.

# 8. Architectural views

* Account creation - Users can create an account. Once they do, they have the option to decide the privacy level of stories they share.
* Sharing stories - Users can read and comment on other stories.
* Comments - Users can give points or reduce points on comments depending on how helpful it was or not.
* Online appointments - Users can 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.
* Privacy levels - Users can change their privacy levels. Low privacy will make it, so the user’s information is available to the public and their physicians. High privacy will make it, so the user’s information is only available to their physicians (mental health professionals).
* Peer reviewed articles - Professional articles are available. These articles may consist of studies on mental health, and advice on how to manage your mental health better.

Risk Management



Master Test Plan



**Post sharing page (Naaman home page with sample posts)**

Used language is PHP and framework is CodeIgniter 3 also using HTML, CSS, bootstrap library as frontend development. This combination is the most common combination in industrial development. Therefore, the development team select these tools and technology for the developing this selected project. CodeIgniter 3 is the most popular PHP framework because it is lightweight in that case if developing php application with CodeIgniter it will run smooth and function without slowing in low performance server it will be benefit for startup project. CodeIgniter run with MVC architecture.

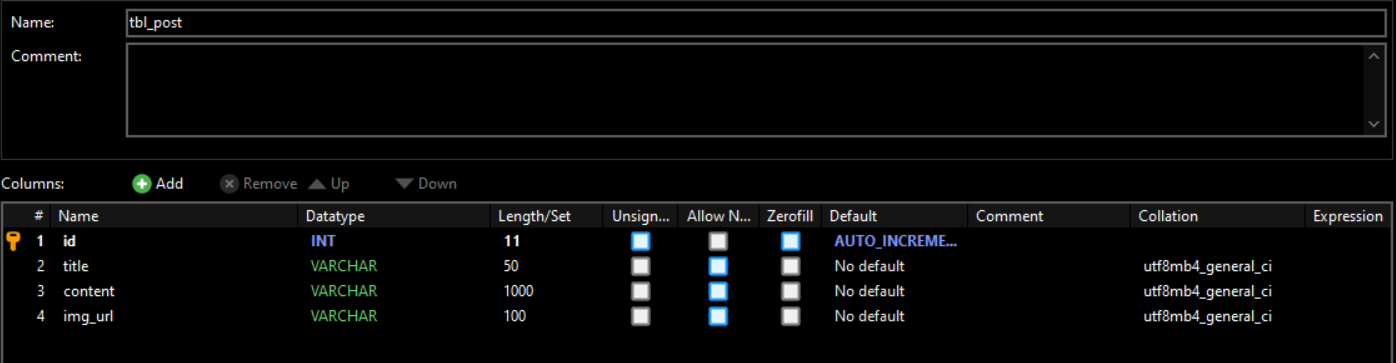
The Model-View-Controller (MVC) is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller. Each of these components are built to handle specific development aspects of an application. MVC is one of the most frequently used industry-standard web development frameworks to create scalable and extensible projects.

CodeIgniter is designed to deliver maximum performance in less time within a clean environment. To achieve this, each developing process is designed in a simplified way.

From technical point of view, it is dynamically instantiation (libraries are loaded on request which makes it light-weighted), loose coupling. (Components rely very less on each other) and component individuality (each class and its functions are narrowly focused only towards their purpose).

This post loads from database

**Database table for using post storing**



**Functioning user interface**

Graphical user interface, application, website

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Test topic 1, test topic 2 and 3,4 are the sample post in database. that posts load in web browser dynamically while the page is loading. Profile pictures and other images must be load dynamically because this content depend on login user. At the same time user can make post or comment using this interface and make appointment.

**Database connection**

Text

Description automatically generated

This is the main database connection in CodeIgniter. Current hosted server is local host (XAMPP). MySQL is the default database driver. Database function is shown below.

**Graphical user interface, application

Description automatically generated**

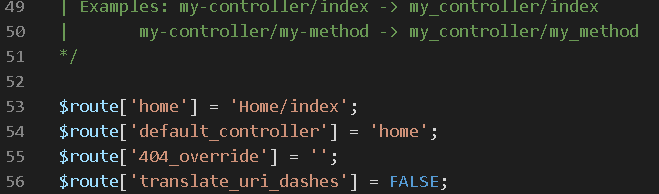
**Base URL configuration code lines (application/config/config.php)Text

Description automatically generated**

Typically, this will be CodeIgniter base URL.

**URL routing configurations**

application/config/routes.php



This route indicates which controller class should be loaded if the URI contains no data.

**View code lines**

Application/view/template/header

Text

Description automatically generated

This is the application view header for using commonly in the application

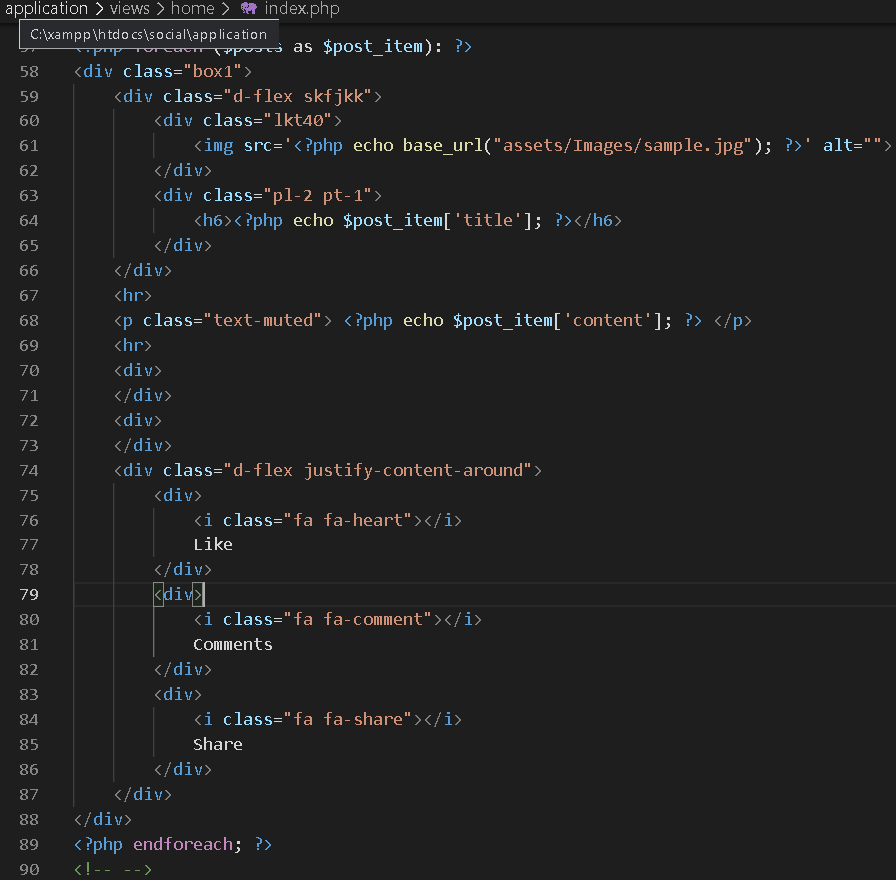
Application/views/home/index.php

Text

Description automatically generated

This is the main home page in application

**Dynamic post loading code lines** (Application/views/home/index.php)



This code line is used to load posts from the database to the user interface. Data sent by the model and past the controller and finally showing the view.

Path is Model > Controller > View

**Controller code lines**

**Text

Description automatically generated**

Controllers control the application function and connect view and model.

**Models code lines**

Text

Description automatically generated

Models is the database connection layer. Models connect to the database and get data to the system and pass the data to the controller. In CodeIgniter has query language for dealing with database it working as MSSQL query.

**User Signup Page (Naaman registration page with sample posts)**

**Functioning user interface**

Graphical user interface, application

Description automatically generated

This is the user registration page that allows user to sign up with the system. It collects user first name, last name, email and password. Its password must contain at least 3 characters.

This UI has designed ensuring the responsum’s that allows user to use the website from using any of their devices.

**UI Code Text

Description automatically generated**

**Controller code lines**Text

Description automatically generatedUser data fields are validated and saving into the database are happening here. Then, also invalid data parameters are handled, and errors are returning to the user. User password is saved as a password hash to ensure the maximum system security.

**User Account Model**

Graphical user interface, text

Description automatically generated with medium confidence

This class represents the user account model. It contains all the user data fields. CodeIgniter base model class has been extended.

**Routes**

Text

Description automatically generated

**Unit Testing** Text

Description automatically generated

PHP unit test has added totest the user signup.

# 

# Registration page and Login page

Here I developed, registration page and login page using Codeigniter 3, a famous PHP framework. Moreover. HTML, CSS, and Bootstrap 4.7 is used to develop the frontend. Our team chose this selection since Codeigniter is lightweight and it is most commonly used in industry. It follows MVC architecture and performs functions smoothly without unnecessary delay.

The Model-View-Controler (MVC) is the common industry standard which is used to develop websites with scalable features integrated. It consists of three main logical components

* **Model**  - Logic regarding data is handled.
* **View**  - Shows the user the information from the model.
* **Controler** - Manages the flow of data into a model object and refreshes the display as data changes.

Since we need to deal with sensitive details in the login page and registration page, for security purposes I added md5 encryption.

**Database table**

**Graphical user interface, text, application, chat or text message

Description automatically generated**

**Functioning User Interface**

**Login Page**

**A picture containing text, electronics, indoor, computer

Description automatically generated**

**Registration Page**

**A picture containing text, monitor, indoor, electronics

Description automatically generated**

**Database connection**

Text

Description automatically generated

**Login page**

Here, PHP is used to connect with the database. “md5” encryption method is used to protect sensitive information. The data validations are added. The users can use this page to log into the system.

* **Backend code**

**Text

Description automatically generated**

* **Frontend code**

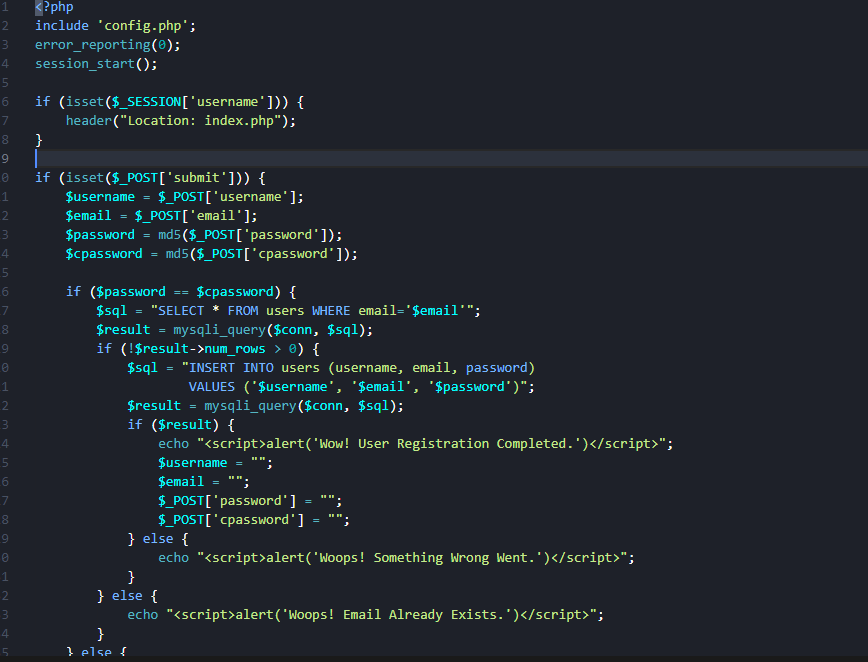
**Text

Description automatically generated**

**Registration page**

Here, PHP is used to connect with the database. “md5” encryption method is used to protect sensitive information. The data validations are added. The users can use this page to register into the system.

* **Backend code**

****

* **Frontend code**

**Text

Description automatically generated**

**Logout**

All the login sessions are destroyed here. The users can use this page to log out from the system.

Graphical user interface, text

Description automatically generated

# Internal User Acceptance Testing (Alpha)

## Scenario Description

* To test the base functionality of the executable architecture
* Test scenarios are a set of tests to validate the executable architecture ability to perform the CCRD use case
* Internal user acceptance testing is performed by the developer to check the functionality of the current state of the product

## Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| Version # | Date | Author | Description |
| 0.1 | 29/09/2021 | Michael Weisang | Initial Draft |
| 1.0 | 11/10/2021 | Michael Weisang | Initial Version |

## Test Scripts

The following scripts will cover this scenario:

* 1.1 Test Script\_access the web



* 1.2 Test Script\_Registration



* 1.3 Test Script\_login



* 1.4 Test Script\_Posting



## Use Case

* Access the web
* Register new user
* User login
* Post article, story, or comment

## User Groups

* Group 2 “Runtime Terror” Script 1: Access the web

### Script Description

* This test script testing the ability of the user to access the website

### Testing Requirements

This test script covers the following specific testing requirements:

* The website is online
* The user has never visited the website

### Setup

* none

### Teardown

* Document the test result

### Script Steps

| **Step #** | **Test Action** | **Expected Results** | **Pass/ Fail** |
| --- | --- | --- | --- |
| 1 | Users request the Remote Mental Health Management system over the web browser using link | The browser displays the website |  |

### Test Execution

|  |  |  |  |
| --- | --- | --- | --- |
| Date/Time | Tester | Test Phase | Status |
| 11/10/21 13:35 | Michael Weisang | Step 1 | Failed |
| 11/10/21 15:30 | Dehemi | Step 1 | Success |

* Script 2: Registration

### Script Description

* This test script testing the ability of system to create an account from user inputted data

### Testing Requirements

This test script covers the following specific testing requirements:

* The website is online
* The user has never made an account

### Setup

* none

### Teardown

* Document the test result

### Script Steps

| **Step #** | **Test Action** | **Expected Results** | **Pass/ Fail** |
| --- | --- | --- | --- |
| 1 | Users request the Remote Mental Health Management system over the web browser using link | The browser displays the website |  |
| 2 | Users go to registration page | The system displays registration page with form |  |
| 3 | Users fills the form and submit it | The system validates the data and record it to the database |  |

### Test Execution

|  |  |  |  |
| --- | --- | --- | --- |
| Date/Time | Tester | Test Phase | Status |
| 9/10/21 13:35 | Sobana | Step 3 | Success |

* Script 3: Login

### Script Description

* This test script testing the ability of system to create a login session

### Testing Requirements

This test script covers the following specific testing requirements:

* The website is online
* The user has an account

### Setup

* none

### Teardown

* Document the test result

### Script Steps

| **Step #** | **Test Action** | **Expected Results** | **Pass/ Fail** |
| --- | --- | --- | --- |
| 1 | Users request the Remote Mental Health Management system over the web browser using link | The browser displays the website |  |
| 2 | Users go to login page | The system displays login page with form |  |
| 3 | Users fills the login detail and login | The system validates the data and record the sessions |  |

### Test Execution

|  |  |  |  |
| --- | --- | --- | --- |
| Date/Time | Tester | Test Phase | Status |
| 9/10/21 13:35 | Gaury | Step 3 | Success |

* Script 4: Posting

### Script Description

* This test script testing the ability of system to post a story to the homepage

### Testing Requirements

This test script covers the following specific testing requirements:

* The website is online
* The user has an account

### Setup

* none

### Teardown

* Document the test result

### Script Steps

| **Step #** | **Test Action** | **Expected Results** | **Pass/ Fail** |
| --- | --- | --- | --- |
| 1 | Users request to post the story | The system validates the login sessions  The system expands the post story section |  |
| 2 | Users input the story and post | The system records the data to the database  The system displays the new post in the homepage |  |

### Test Execution

|  |  |  |  |
| --- | --- | --- | --- |
| Date/Time | Tester | Test Phase | Status |
| 9/10/21 13:35 | Dehemi | Step 3 | Success |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Scenario | Tester | Result | Comment |
| 1 | Access the web | Michael | Fail | The reason of failure is still unknown  It speculated be the laptop problem |
|  |  | Dehemi | Passed |  |
| 2 | Registration | Sobana | Passed |  |
| 3 | Login | Gaury | Passed |  |
| 4 | Posting | Dehemi | Passed |  |

Feedback

|  |  |  |
| --- | --- | --- |
| Category  (Aesthetic/function) | Application/system component | Comments |
| Aesthetic | Homepage | The homepage is nice and elegant with a comfortable feeling but the navigation bar at the top of the page looks empty |
| Function | Homepage | Write an article supposed to be a function available for physicians only |
| Aesthetic | Registration page | There is no design yet for the page |
| Function | Registration page | There is a lack of detail still needed in the form to create an account |
| Aesthetic | Login Page | The login page is nicely design and aesthetic |
| Function | Integration of the unit | The integration between the page has met an error |

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 |
| Temuulen Tsengel | Completer | Pay attention to details.  Make sure that the job is completed before the deadline.  Clarify goals, promote decision making. |
| Chhiring Moktan | Shaper | Drive team forward.  Determine the most effective methods for solving issues. |
| Roshan Saud | Resource investigator | Explore opportunities and contacts.  Negotiate for resources.  Provide support and encouragement to 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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Subject** | **Phase** | **Iteration** | **Dates** | **Primary objectives** (risks and use case scenarios) |
| ITC309 – Software Development Project 2 | Construction Phase | C-1 | 28/02 – 14/03 | Team meeting planning  Implement 2nd Highest Priority Use Case(s)  Complete Development and Integration Testing for 2nd Highest Priority Use Case(s) |
| C-2 | 14/03 – 28/03 | Complete Internal User Acceptance Testing for 2nd Highest Priority Use Case(s)  Implement 3rd Highest Priority Use Case(s)  Complete Development and Integration Testing for 3rd Highest Priority Use Case(s) |
| C-3 | 28/03 – 11/04 | Complete Internal User Acceptance Testing for 3rd Highest Priority Use Case(s)  Implement 4th Highest Priority Use Case(s)  Complete Development and Integration Testing for 4th Highest Priority Use Case(s) |
| Mid-Session Break 04/04 – 15/04 | | |
| C-4 | 11/04 – 25/04 | Complete Internal User Acceptance Testing for 4th Highest Priority Use Case(s)  Deliver Initial Operation Capability Milestone (IOCM)  Complete Construction Phase Project Assessment |
| Transition Phase | T-1 | 25/04 – 09/05 | Deploy Application in Production Environment  Complete 1st Round External User Acceptance Testing  Resolve Any Identified Issues |
| T-2 | 09/05 – 23/05 | Complete 2nd Round External User Acceptance Testing  Resolve Any Identified Issues |
| T-3 | 23/05 – 06/06 | Contingency  Deliver Product Release Milestone (PRM)  Complete Final Project Assessment |

# Elaboration Phase Status Assessment

## 1. Assessment against Objectives of the Elaboration Phase

### 1.1 Has ‘end-to-end production level support for the most critical, core (risky, difficult) use case, using the chosen software architecture, in the intended production environment’ been achieved?

*Yes,* we *have partially* achieved this objective. This is demonstrated in the online class via presentation.

During the Inception Phase, we identified *posting stories* as the critical core use case. This is because *Posting stories is the main core function of the website.*

We identified *Three-tier architecture* as a feasible approach to addressing the requirements of the projects as outlined in the updated and continuing Architectural Notebook, which may be accessed here:

<https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1//_1194042_1/A3%20revised%20Architectural%20notebook.docx>

The main architectural elements which are demonstrated by the executable architecture are:

*Registration*

*Login*

*Posting a story*

*View a story*

Those aspects of the architecture not addressed include:

*Posting a comment*

*Posting an article*

*Booking a consultation with a physicians*

Correct support for the CCRD use case by the executable architecture *was partially* achieved as demonstrated and documented in the following user acceptance tests.

*Registration*

*Login*

*Posting a story*

Actual test results can be accessed from here: *<insert link to completed user acceptance tests here>*

### 1.2 Have all critical and significant project risks been addressed and mitigated?

The following list identifies the most critical and significant product, technical and project management risks to the project. Mitigation strategies identified and applied and the current status of the risk are also listed.

<https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1//_1194187_1/Risk%20Management%20%281%29.xlsx>

### 1.3 Have the initial Vision, Requirements (Scope), or Architecture changed?

During the Elaboration Phase, our understanding of the projects aims evolved as follows:

Vision

*There is no major change in the vision document. There are only a further, more detailed explanation or description added to the vision document. The reason for the added detail is because of the brief information provided is not enough.*

Requirements (Functional)

*There is no major change in the Use case description as in the initial use case, the CCRD use case is already mentioned. The change happening in the document is only a further detail added to the current use case so that it will be more specific.*

Requirements (Non-Functional)

*There are no major change in the Non-Functional Requirement documents. There is only a minor added information to further explain the NFR requirement.*

Architecture

*There are no major changes in the architectural style. There are some minor detail added for the architectural notebook to be more specific and detailed.*

### 1.4 Have the initial Project Plan or Master Test Plan changed?

During the Elaboration Phase, our understanding of the best way to implement the project evolved as follows:

Project Plan

*The project plan is not changed at all*

Master Test Plan

*There is a major change happening in the master test plan. One of the reasons of the massive change is that the initial master test plan are not in the best state with a lot of information lacking. In the revised master test plan, a lot more detail has been put and some of the information like planning are changed.*

## 2. Deliverables

*For each deliverable:*

Text

Description automatically generated

<https://interact2.csu.edu.au/courses/1/S-ITC309_202230_SM_I/groups/_73699_1//_1263536_1/Revised%20Vision%20Document.docx>

No Issue

Text

Description automatically generated

<https://interact2.csu.edu.au/courses/1/S-ITC309_202230_SM_I/groups/_73699_1//_1263539_1/Revised%20Architectural%20notebook.docx>

No Issue

Graphical user interface, text

Description automatically generated

<https://interact2.csu.edu.au/courses/1/S-ITC309_202230_SM_I/groups/_73699_1//_1263552_1/Risk%20Management%20%281%29.xlsx>

No Issue

Text

Description automatically generated

<https://interact2.csu.edu.au/courses/1/S-ITC309_202230_SM_I/groups/_73699_1//_1263537_1/Revised%20Use%20Case%20Description.docx>

No Issue

Text

Description automatically generated

<https://interact2.csu.edu.au/courses/1/S-ITC309_202230_SM_I/groups/_73699_1//_1263544_1/NFR.docx>

No Issue

Graphical user interface, text, application

Description automatically generated

<https://interact2.csu.edu.au/courses/1/S-ITC309_202230_SM_I/groups/_73699_1//_1263551_1/Master%20Test%20Plan.doc>

No issue

Text

Description automatically generated

<https://interact2.csu.edu.au/courses/1/S-ITC309_202230_SM_I/groups/_73699_1//_1263535_1/Revised%20Project%20Plan.docx>

No Issues

## 2.8 Domain Class Diagram

<https://interact2.csu.edu.au/courses/1/S-ITC309_202230_SM_I/groups/_73699_1//_1263545_1/Domain%20class%20diagram.png>

## 2.9 ERD Diagram

<https://interact2.csu.edu.au/courses/1/S-ITC309_202230_SM_I/groups/_73699_1//_1263547_1/ERD.png>

## 2.10 Homepage and sharing document

<https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1//_1193982_1/Home%20page%20with%20post%20sharing.docx>

## 2.11 Registration document

<https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1//_1194192_1/User%20Signup%20Page.docx>

## 2.12 Login Diagram

<https://interact2.csu.edu.au/courses/1/S-ITC303_202160_SM_I/groups/_68438_1//_1194208_1/Log%20in.docx>

## 2.13 User Acceptance Test

<https://interact2.csu.edu.au/courses/1/S-ITC309_202230_SM_I/groups/_73699_1//_1263542_1/UAT%20Test%20Scenario%20and%20result%20resubmission.docx>

## 2.14 Executable Architecture

<https://github.com/Temuulen2830/ITC303-Remote-Health-Management/tree/Dehemi>

<https://github.com/Temuulen2830/ITC303-Remote-Health-Management/tree/Sobana>

<https://github.com/Temuulen2830/ITC303-Remote-Health-Management/tree/main/Login_registration>

Major Issue in codding. The developer delivers the code late and did not use the specification specified in the project. This has led to no time for integration

## 3. General Issues

*For each issue*

### 3.1 Time

*Due to long due date in the assignment, the team member keeps on delaying the project until there is not enough time to properly finish the deliverables.*

*The issue is still ongoing and for the next phase, there will be a stricter monitoring*

### 3.2 Coding

*Due to delaying the task assigned, the team member ends up rushing the coding near the end of the term which led to major issue in the end.*

*The issue is still ongoing and for the next phase, there will be a increased pressure to finish on schedule.*

## 4 Summary – Overall Project Progress

*The project is progressing albeit slowly. Most of the aim for the elaboration phase has been fulfilled except the executable architecture. The unit is developed nicely, but the integration has major problem. With the current pace, the project is facing risk of failure if there is no major change taken by the team. Thankfully in the next phase (Construction phase), all the major issue faced in this phase can be fixed and reworked.   
  
There is no change for the project scope and project plan*