LONDON METROPOLITAN UNIVERSITY

islington college

(इस्लिङ्टन कलेज)

Module Code & Module Title

CS5002NI SOFTWARE ENGINEERING

Assessment Weightage & Type

35% Individual Coursework

Year and Semester

2022-23 Spring

Student Name: Anjan Giri

London Met ID: 22085496

College ID: NP01CP4S230104

Assignment Due Date: 7th May 2024

Assignment Submission Date: 6th May 2024

Word Count: 5331

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Table of Contents

Table of Figures

1. Introduction

The provided coursework is the second coursework of the module Software Engineering (Module Code: CS5002NI) accounting for 35% of the total module grades. The coursework requires us to make a systematic plan and design for the system as demanded by the scenario provided in the question. The main goal of the coursework is to make us familiar with various concepts of Software Engineering and to give us understanding of different works related to project management and development.

According to the scenario provided, McGregor Institute is an institute of Botanical Training located at Godawari, Lalitpur. The Ireland based institute has been operating for almost 7 years in Nepal. Different types of courses are provided by the institute which includes undergraduate and postgraduate courses specializing in agriculture along with horticulture. The institute is affiliated with Dublin City University and with the recent hike in number of people being interested in the domain of agriculture, the institute is planning to launch various short-term courses. They also plan on selling different varieties of plants and want to build a community of people with interests in plants by creating a forum where such people can share, react and discuss on each other's ideas.

Based on the above scenario, a system is to be made meeting all the mentioned requirements by the institute. The system should be completed with proper management and in proper schedule to ensure smooth operations of the institute. Various operations should be operated by the system for meeting the requirements and to provide effective learning platform for all plant enthusiasts that come across the institute. For this, different aspects of software engineering are used which include creating Work Breakdown Structure (WBS), Gantt Chart, Use Case, Collaboration Diagram, Sequence Diagram, Class Diagram along with the prototypes for the proposed system.

2. WBS and Gantt Chart

2.1 Work Breakdown Structure (WBS)

Work Breakdown Structure (WBS) is the process of breaking work into smaller tasks in a productive way which is used to make a work more manageable and approachable. It is one of the most important aspects of project management. It helps in determining time required for the completion of the project along with the cost required and complexity of the work. In other words, WBS is a step-by-step process to complete large projects with the help of several small moving pieces.

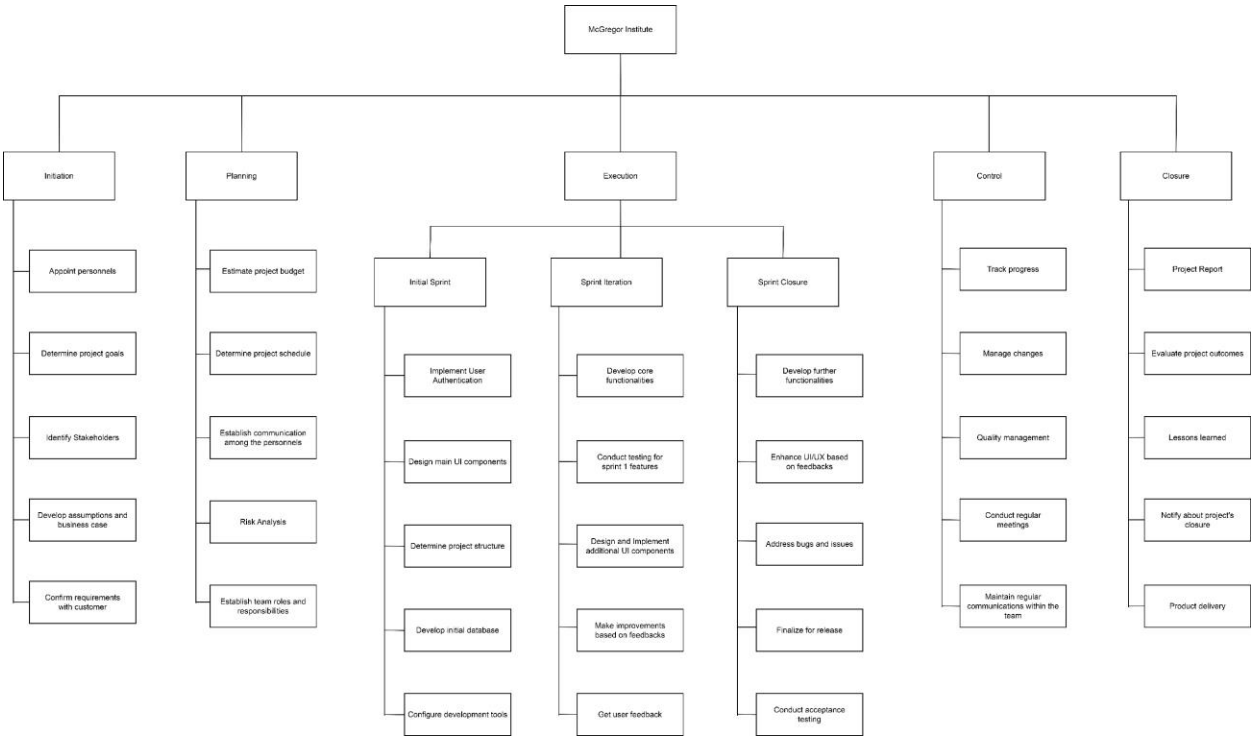

Figure 1: Work Breakdown Structure of the system

2.2 Gantt Chart

Gantt chart is defined as a graphical representation of a project schedule. It is most commonly used chart in project management. It helps in managing and monitoring various tasks while completing a project. The chart shows the project timeline along with both completed and scheduled tasks. (Grant, 2023) Gantt chart helps project managers to keep track of the progress made for completing the work and to manage time, personnels and cost for the project more effectively.

The gantt chart provided below is created based on agile methodology. According to the methodology, there are six phases included in the gantt chart:

a. Planning

b. Designing

c. Development

d. Testing

e. Deployment

f. Review

The initial plans and ideas for the project are completed in the first phase i.e., planning phase. Then, the designing works for the system are carried out where interface of the system, architects to be used and prototypes are determined. The development phase includes three sub phases which are initial sprint, sprint iteration and sprint closure. All the development related works are performed in this phase. Coding, initial testing and developing various functions all are included in the development phase. After that, in testing phase, final tests are carried out before releasing the project or system. Issues are searched throughout the system and are solved and the system is made ready for release. In deployment phase, the finished project is deployed or released after much preparations and planning. Then, reviews from stakeholders and users are taken after the system is deployed. Performance analysis of the whole project is conducted and a well-documented report is prepared.

	ACTIVITIES	ASSIGNEE	EH	START	DUE	STATUS	%
	Planning:		-	05/Mar	27/Mar		0%
1	Appoint project personnels	Unassigned	-	05/Mar	11/Mar	Finished	0%
2	Define project scope and go...	Unassigned	-	12/Mar	14/Mar	Finished	0%
3	Identify Stakeholders	Unassigned	-	15/Mar	19/Mar	Finished	0%
4	Determine project schedule	Unassigned	-	20/Mar	22/Mar	Finished	0%
5	Estimate project cost	Unassigned	-	24/Mar	27/Mar	Finished	0%
	Designing:		-	01/Apr	30/Apr		0%
7	Create design specification	Unassigned	-	01/Apr	05/Apr	Finished	0%
8	Interface design	Unassigned	-	07/Apr	10/Apr	Finished	0%
9	Architecture planning	Unassigned	-	11/Apr	16/Apr	Approval reque	0%
10	Conduct database planning	Unassigned	-	17/Apr	21/Apr	Approval reque	0%
11	Create wireframes or proto...	Unassigned	-	22/Apr	30/Apr	Approval reque	0%
	Development (Initial Sprint):		-	01/May	26/May		0%
13	Develop main UI components	Unassigned	-	01/May	08/May	Started	0%
14	Start backend development	Unassigned	-	09/May	22/May	Not started	0%
15	Perform initial testing	Unassigned	-	23/May	24/May	Not started	0%
16	Get feedback	Unassigned	-	24/May	26/May	Not started	0%
	Development (Sprint Iteration):		-	27/May	19/Jun		0%
18	Develop core functionalities	Unassigned	-	27/May	04/Jun	Not started	0%
19	Integrate developed modules	Unassigned	-	05/Jun	09/Jun	Not started	0%
20	Make improvements based ...	Unassigned	-	10/Jun	14/Jun	Not started	0%
21	Perform testing with added ...	Unassigned	-	16/Jun	18/Jun	Not started	0%
22	Get feedback	Unassigned	-	16/Jun	19/Jun	Not started	0%
	Development (Sprint Closure):		-	20/Jun	11/Jul		0%
24	Develop further functionaliti...	Unassigned	-	20/Jun	01/Jul	Not started	0%
25	Review code	Unassigned	-	02/Jul	05/Jul	Not started	0%
26	Improve system based on f...	Unassigned	-	07/Jul	11/Jul	Not started	0%
	Testing:		-	12/Jul	04/Aug		0%
28	Perform system testing	Unassigned	-	12/Jul	17/Jul	Not started	0%
29	Record issues found in the t...	Unassigned	-	18/Jul	23/Jul	Not started	0%
30	Resolve the issues	Unassigned	-	24/Jul	29/Jul	Not started	0%
31	Conduct user testing	Unassigned	-	30/Jul	02/Aug	Not started	0%
32	Get feedback and resolve is...	Unassigned	-	31/Jul	04/Aug	Not started	0%
	Deployment:		-	05/Aug	30/Aug		0%
34	Deployment planning	Unassigned	-	05/Aug	09/Aug	Not started	0%
35	Pre-deployment preparations	Unassigned	-	11/Aug	16/Aug	Not started	0%
36	Actual deployment	Unassigned	-	18/Aug	19/Aug	Not started	0%
37	Monitor deployed product	Unassigned	-	19/Aug	30/Aug	Not started	0%
	Review:		-	21/Aug	11/Sep		0%
39	Gather feedbacks from stak...	Unassigned	-	21/Aug	26/Aug	Not started	0%
40	Get feedbacks from users	Unassigned	-	22/Aug	30/Aug	Not started	0%
41	Performance analysis	Unassigned	-	01/Sep	05/Sep	Not started	0%
42	Project report	Unassigned	-	06/Sep	11/Sep	Not started	0%

Figure 2: Tasks included in the gantt chart

In the figure above, all the tasks to be performed in each phases are listed down.

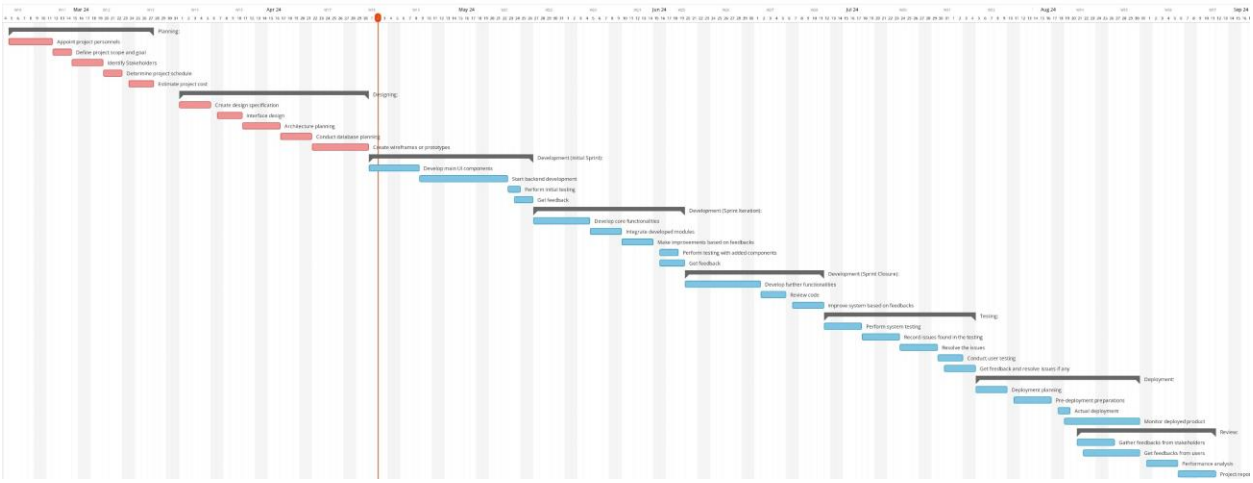

Figure 3: gantt chart of the tasks

The gantt chart is created based on the tasks that are listed in Figure: 2. The timeline for each tasks are well distributed and managed properly which has resulted in effective creation of the schedule for completion of the project.

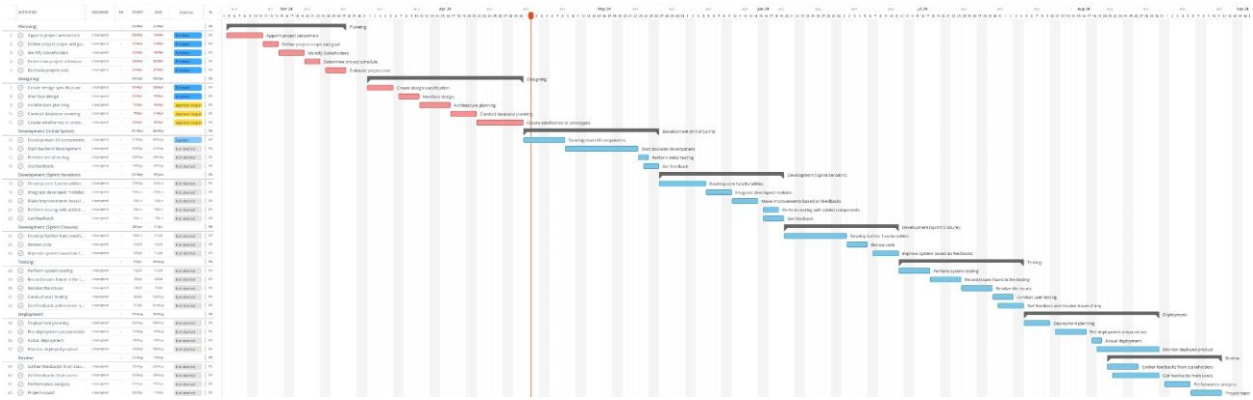

Figure 4: full gantt chart along with the tasks included

3. Use Case

Use cases are a method utilized in system analysis to define and organize system requirements. They include different arrangements of interactions between users and systems inside a given context, all aiming to achieve specific objectives. This results in a detailed document outlining each user's steps to complete a task, promoting understanding of user-system interactions, and encouraging successful communication among extend partners and designers. (Brush, 2024)

Use case is used to model a current or proposed system. There are various notations used in a use case:

- Use Case: It is the system function or process.

- Actor: Actor is an entity that interacts with a use case which plays a role in a business.

- Communication Link: It is a link that is used to connect actors to use cases. It indicates that an actor and use case communicate with each other.

- Boundary: Boundary covers the entire system defined in the requirement. It is used to enclose all use cases.

There are many relationships used in a use case which are discussed below:

- Association: It is present in every use case. It relates an actor with a use case and is denoted by a simple line.

- Generalization: It is a parent-child relation which is denoted by a directed arrow with a triangle arrowhead where a child is an enhancement of parent. This relation can be present in both actor and use case.

- Extends: It shows optional functionality of a system. It is denoted by a dotted arrow with the ‘extends’ label.

- Includes: It adds new functionality which is not defined on the base use case. It is denoted by a dotted arrow with the ‘include’ label.

3.1 Use Case Diagram

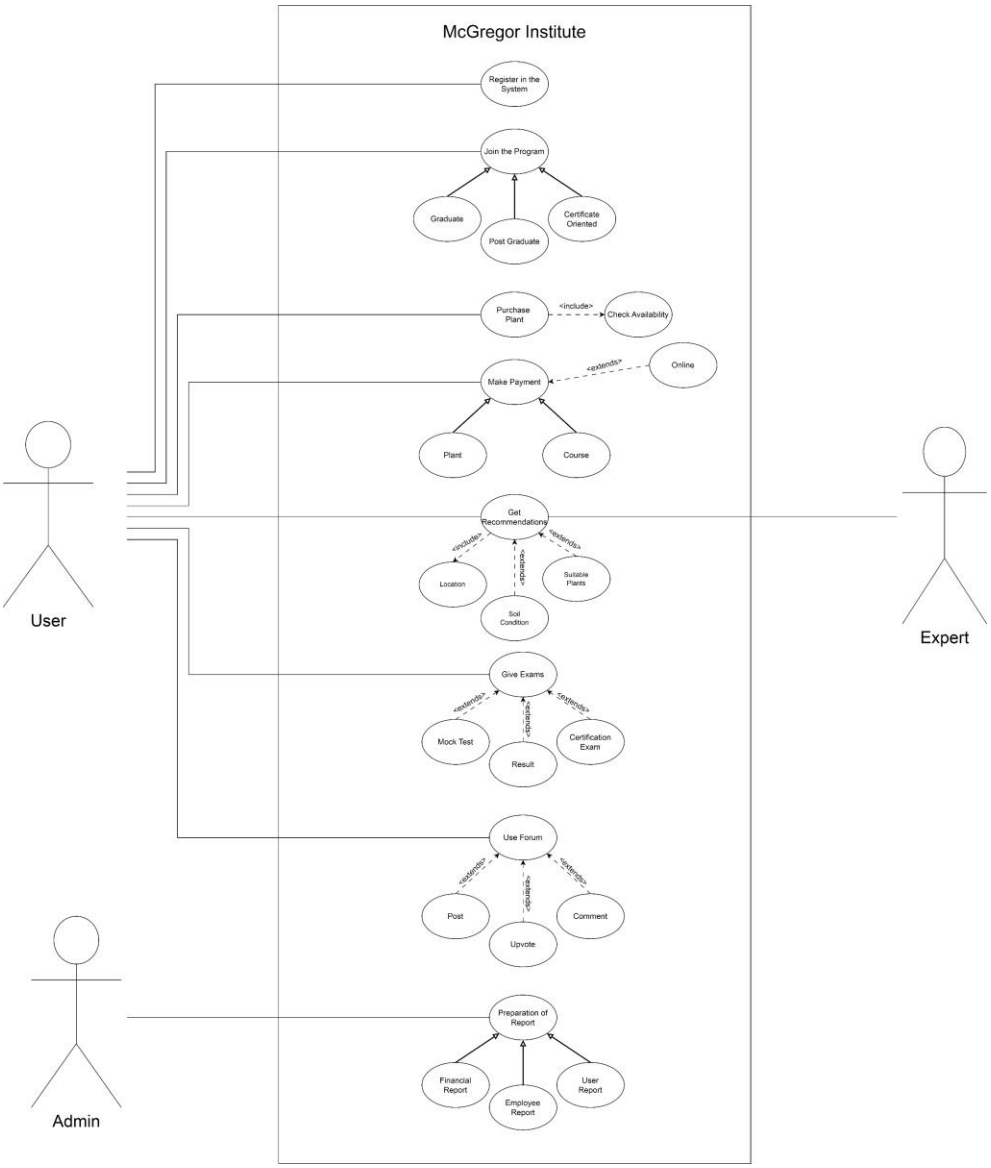

Figure 5: Use Case of the system

3.2 High Level Use Case

3.2.1 First Case

Use Case: Register in the System

Actors: User

Description: A user goes into the system and registers by providing his/her details including name, contact, address, gender, date of registration and so on.

3.2.2 Second Case

Use Case: Join the Program

Actors: User

Description: User joins the program of his choice through the system that are available in the institute which are graduate, post graduate and certificate oriented.

3.2.3 Third Case

Use Case: Purchase Plant

Actors: User

Description: The user purchases the plant of his/her choice by exploring through the system. While purchasing a plant, first its availability is checked and then only if the plant is available, the purchase is made.

3.2.4 Fourth Case

Use Case: Make Payment

Actors: User

Description: The user makes payment of any of the two transactions whether purchasing plant or enrolling in a course. The payment can be made through either online methods or cash. The payment can be made for both type of transactions.

3.2.5 Fifth Case

Use Case: Get Recommendations

Actors: User and Expert

Description: The user can provide their location and even the condition and quality of the soil which are viewed by the experts and the expert can recommend the type of plant that can be suitable for the conditions provided by the user.

3.2.6 Sixth Case

Use Case: Give Exams

Actors: User

Description: The user can give examination of his/her choice. He/she should be able to give mock tests to evaluate their abilities and certificate exams to gain certificates in particular fields. The user is also able to check results of the examinations.

3.2.7 Seventh Case

Use Case: Use Forum

Actors: User

Description: User can use forum to perform various kinds of activities such as posting his/her own opinion on various topics related to plants and the user can also comment on other's opinions and upvote them as well.

3.2.8 Eight Case

Use Case: Preparation of Report

Actors: Admin

Description: Admin can prepare report of all the ongoing things in the institute. He/she can make report of financial status of the institute along with employee report and user report.

3.3 Expanded Use Case

3.3.1 Use Case 1

Use Case: Join the Program

Actors: User

Description: User joins the program of his choice through the system that are available in the institute which are graduate, post graduate and certificate oriented.

Typical Course of Events:

Actor Action	System Response
1. A user registers into the system.	2. Presents personal information and home page.
3. User clicks on programs.	4. Provides different programs to choose from.
5. The user chooses the program that he/she wants to enroll into.	6. Checks whether the program is paid or unpaid.
7. If the program is paid, user pays the fee and if unpaid, no need to pay any fee.	8. Validates payment and the user is enrolled in the program.

Alternative Courses:

Line 5: If the user does not choose any program to enroll into, the use case ends.

Line 7: If the user does not make payment for the programs that are paid programs, the use case ends.

3.3.2 Use Case 2

Use Case: Purchase Plant

Actors: User

Description: The user purchases the plant of his/her choice by exploring through the system. While purchasing a plant, first its availability is checked and then only if the plant is available, the purchase is made.

Typical Course of Events:

Actor Action	System Response
1. User goes into the purchase plant page.	2. Displays all the plants.
3. Selects a plant.	4. Displays the information about the selected plant.
5. Clicks on buy.	6. Checks the availability of the plant.
7. Selects payment type and pays for the plant.	8. Validates payment and purchase is made.

Alternative Courses:

Line 6: If the plant is not available, use case ends.

Line 7: If the payment is not made, use case ends.

4. Collaboration Diagram

Collaboration diagram is an illustration of relationships between software objects. It is also know as communication diagram. Collaboration diagrams are used to show how objects perform the behaviour of a certain use case by interacting with each other. It shows the links and messages that are sent among the objects. They are the main source of information to determine class responsibilities and interfaces. A collaboration diagram consists of various components like objects, actors, link and messages.

There are various steps for creating a collaboration diagram. The steps are listed below:

Step 1 - For creating a collaboration diagram, first, the domain classes should be identified and separated. The domain classes are the classes or objects that are affected by the use case processes and also contribute to them. In this case, the domain classes are program, enrollment, payment and user. Then, draw the object symbols for each of the domain classes.

Step 2 - After domain classes are identified, add a control object in the diagram. Control object has same name as the use case. The control object for the diagram is ‘Join The Program’.

Step 3 – Now, add a boundary object in the diagram. The boundary object also has the same name as the use case with the addition of UI (User Interface). It interacts and with the actor and the boundary object for the diagram is ‘Join program UI’.

Step 4 – Add the actor in the diagram. The actor interacts with the system interfaces and is drawn beside the boundary object. It is represented by stick figure. The actor present in this diagram is User.

Step 5 – The final steps for creating a collaboration diagram is to add associations and messages in the diagram. The relations or associations are represented through a simple line connecting each object in the diagram and the messages are represented through

arrows in the diagram. There are various messages present in the following collaboration diagram.

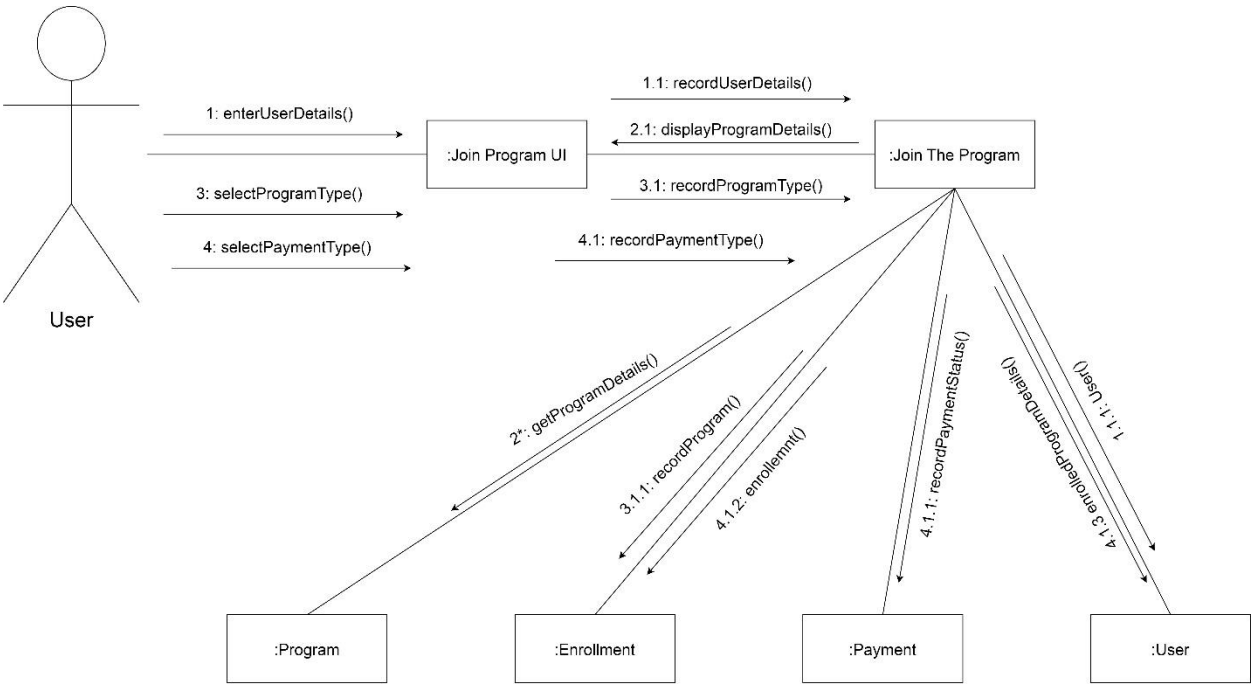

Figure 6: collaboration diagram for 'join the program' use case

Here, the actor in the collaboration diagram is ‘User’ that interacts with the interface of the system which is the boundary object ‘Join Program UI’. The user enters his/her details through the interface. The entered details is then recorded in the control object which displays the entered details to the user through the boundary object. The user also enters payment details and chooses a certain program from the interface which are then recorded in ‘Join The Program’ object. Then, each domain classes get messages from the control object which include getting program details, payment status and so on. The classes also store information on enrollment, which program that the user is enrolled in and the details of the user.

5. Sequence Diagram

Sequence diagram is an interaction diagram that describes how the processes or operations are carried out in the system. It shows the order of interactions between objects and represent the messages sent. It helps to understand requirements for a new system or to document existing processes. Sequence diagrams are time focused and are used to forge the system's object interaction. Another purpose of the diagram is that it helps non-technical person to explain the flow and is also used to document future system flow. Such diagrams consist of object lifeline, messages, fragments, actor, domain classes, boundary and control object, etc.

Like collaboration diagram, there are various steps for creating a sequence diagram which are listed below:

Step 1 - The first step in creating a sequence diagram is to determine the domain classes. Similar to the collaboration diagram, the domain classes in the sequence diagram are user, program, payment and enrollment.

Step 2 – Now, add the control object with its lifeline. Control object has same name as the use case. The control object for this diagram is 'Join The Program'.

Step 3 – Then, the boundary object with its lifeline is added in the diagram. It is responsible for the interactions between the actor and the use case. The boundary object also has the same name as the use case with the addition of UI (User Interface). The boundary object for the diagram is 'Join program UI'.

Step 4 – Add the actor with its lifeline on the diagram. The actor interacts with the boundary object to execute tasks. The actor in this diagram is User.

Step 5 – After the actor is added, now add all the required messages in the diagram from actor, boundary object, control object along with the fragments containing loop (loop fragment) and if else logic (alternative fragment).

Step 6 – Add domain classes objects with their lifelines and also add messages directed to them. Terminate the lifeline of each object after the operations are completed using a X symbol.

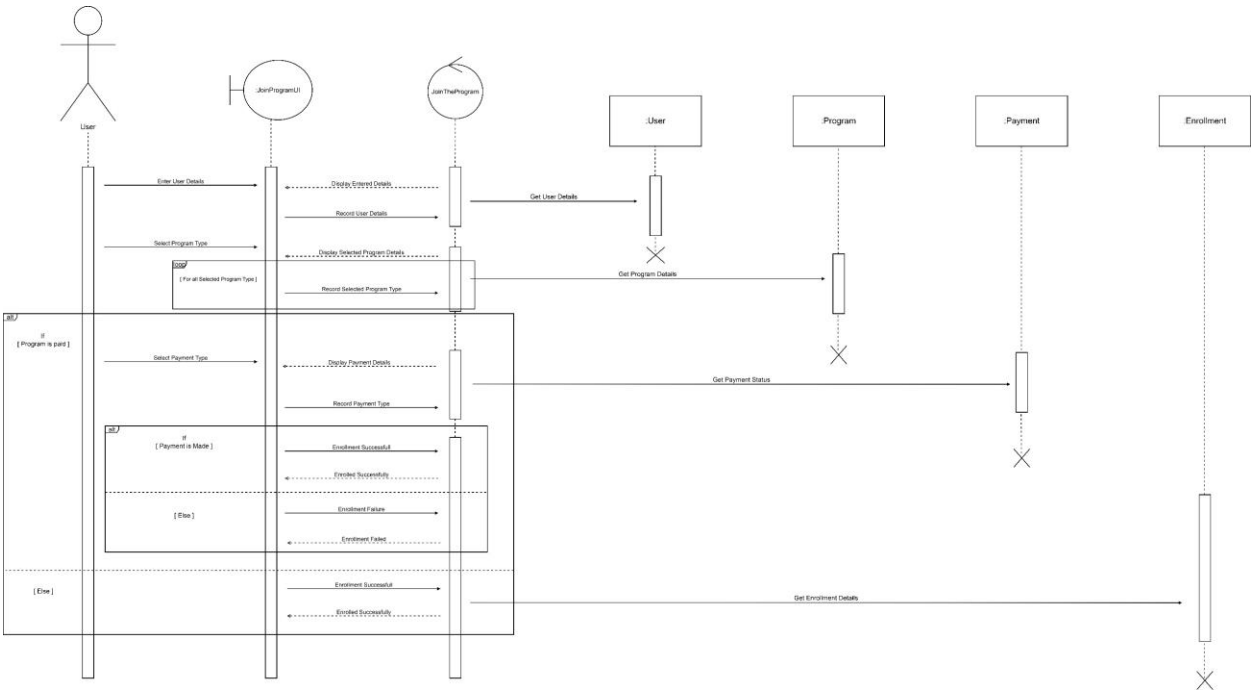

Figure 7: Sequence diagram for "join the program" use case

Here, the actor in the sequence diagram is ‘User’ that interacts with the interface of the system which is the boundary object ‘Join Program UI’. The user enters his/her details through the interface. The entered details is then recorded in the control object which displays the entered details to the user through the boundary object. The user also enters payment details and chooses a certain program from the interface which are then recorded in ‘Join The Program’ object. Then, each domain classes get messages from the control object which include getting program details, payment status and so on. The classes also store information on enrollment and the details of the user.

The diagram also contains various fragments. A loop fragment is present in the diagram which records the selected program type for every selected programs. Two of the alternative fragments are used in the diagram. Such fragments are used to implement if else logics. First alternative fragment checks whether the selected program is paid or unpaid and the second alternative fragment checks whether the payment is made or not.

6. Class Diagram

Class diagram is a type of illustration diagram which is used in software engineering to visually represent structure and relationships of various classes present in the system. It is a structure diagram for designing and modeling software. Class diagram also shows attributes of a class, its methods and connections with other classes.

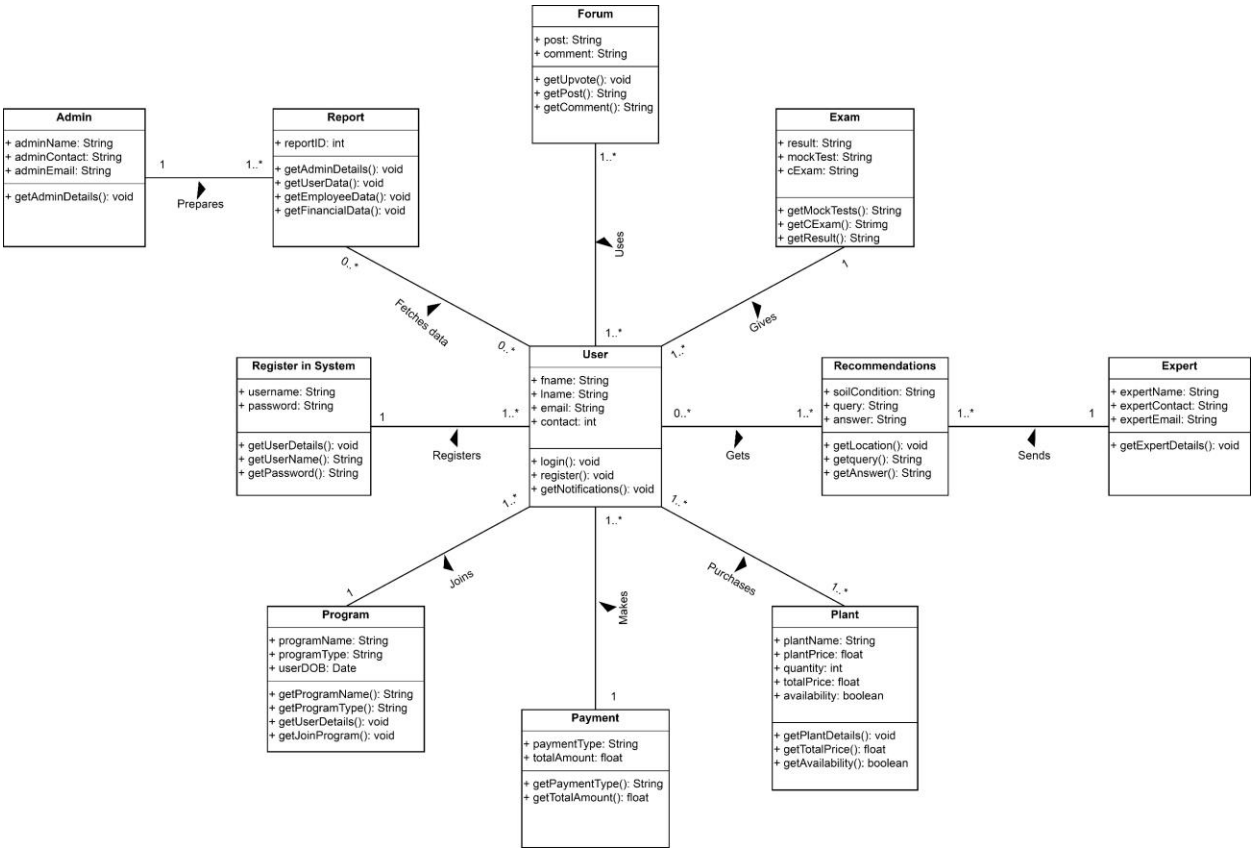

Figure 8: class diagram for the system

7. Further Development

7.1 Architectural Plan

Software Architecture is defined as an outline that allows to define a structural schema for all kinds of software system. It provides predefined set of roles and responsibilities along with subsystems. It also provides rules and roadmap foe defining relationships. Architectural plan is important in a project for defining basic characteristics of the project. Choosing a right architectural plan to meet our objectives and to complete the project effectively is very important.

Layered Architecture

For this project, the architectural plan that is followed is that of Layered Architecture Pattern. It is a most common architecture pattern which is generally built around a database. Layered Architecture is also known as tired architecture. In this architectural plan, the code is arranged in a layer pattern so the data enters the top layer and works its way down to each layer until it reaches the bottom layer.

The layered architecture pattern is generally used for the applications that are needed to build in short period of time. It is preferred by the enterprise applications that require traditional IT processes. This architectural plan is not that complex meaning that it is appropriate for inexperienced developers as well. Although it is easy to use and has many advantages, performance wise, it may be inefficient sometimes.

There are four distinct layers in the layered architecture pattern. Each layer has distinct role with the application. Each layer performs its own operations and work together to complete the pattern. It enables to modify components within a certain layer without affecting other layers. The layers in the architecture pattern are:

- Presentation Layer

- Business Layer

- Persistence Layer

- Database Layer

7.2 Design Pattern

There are many design patterns that can be used to design the project The design pattern that is preferred and used to design this project is the MVC design pattern. It is a design pattern that separates an application into three main components Model, View and Controller. This makes it easier to manage, implement and maintain the codes in the programs. It allows reusability of components and promotes more effective way of software development.

The three main components of the design pattern have their own functions and roles.

Model – It is an object carrying data.

View – It is responsible for the visualization of data.

Controller – It is responsible for maintaining data flow from model and view and vice-versa.

7.3 Development Plan

In the initial phase of the development, the personnels are assigned to the project and different resources required for the project are allocated. A project manager is determined to oversee the state of the project and to ensure the smooth flow of tasks while completing the project and he/she is also responsible to ensure that the project is completed within a certain time in certain allocated cost. Planning is carried out and cost estimation and schedule are also set for taking a systematic approach to carry out the project.

After the initial tasks are completed, the main work for developing the system begins. Design and implementation of the designs along with coding and systematic arrangement of codes are the major factors that are prioritized in the development plans. The programming language used for developing the required system is Java

Programming Language. Java is used as it is easier to integrate the frontend and backend codes of the system together using Java programming language as it also supports integration with HTML and CSS codes which are used for frontend designing of the system. Database designing and implementation is also required for the system. The database used in the system is through mySQL database. It is preferred over other databases as mySQL is a familiar database for all levels of developers and is easy to use through different queries. It also helps to retrieve and manage data in more effective way. Different tools are used for completing the operations and ensuring proper management of the development works. Eclipse is used for coding purposes and to maintain proper management of different files. PhpMyAdmin is used for creating and maintaining database as it is also easier to integrate and connect with the codes in the Eclipse application.

For the coding part of the system, the functions to be developed are also divided into different sections. At first, user registration part should be completed before moving on to other functions of the system. This helps in smooth flow while testing and creating other functions of the system. After the user registration is complete, the development goes to next stage where core functionalities are developed which includes joining the program, purchasing plants and making payment while enrolling in a program or buying a plant. Then, the functions for getting recommendations, giving exams and using the forums are completed after which all the data are collected and page for the admin to prepare a report on required topic is created. Each page is integrated with both frontend and backend development and is completed within the projected time.

7.4 Testing Plan

Testing is one of the most important sectors while developing a system or a software. It is the process of assessing the functionality of the program or system. It helps to check and ensure the desired outcome of the program. After the design and coding are completed, the source code for each aspect is tested out. The created modules are integrated to form a whole system which is then tested to check the output given by each code. The main aim of testing is to find errors.

There are different methods and techniques of testing in software engineering. Each method carries out testing in their own way and rules but the final goal of each type and technique of testing is to find as many errors as possible. Some of the methods of testing are:

Regression Testing

Regression Testing is a method of testing that is performed to ensure that the functionalities that are already present in the system are still working properly after some changes are made elsewhere in the system. When a software is corrected, some aspect of that system or software configuration are changed. This testing helps in making sujre that these types of changes do not affect other functionalities in a system and does not introduce any irregularities or additional errors in the system. It can be conducted both manually and using automated tools.

Black Box Testing

Black Box Testing is a method of testing in which testers evaluate the functionality of the system. In this testing, the testers do not have knowledge of the internal details, structure of code or program. It is also known as behavioural testing or specifications testing. In black box testing, the testers simply interact with the system's interfaces and test its inputs and outputs. There are various techniques in black box testing:

Equivalence Class Partitioning – It is a technique where input values are divided into valid and invalid classes and a certain value is selected to represent each partition as test data. Then, those values are used to carry out the testing.

Boundary Value Analysis – It is a technique where boundaries for input values are determined and the values are selected that are at the boundaries or just inside and outside of the boundaries as test data. Then, those values are used to carry out the testing.

White Box Testing

White Box Testing is a method of testing in which testers have access to the internal code and structure of the system. It is also known as glass box testing or clear box testing. In this method, the testers are able to see through the software's inner workings. It reduces communication overhead between testers and developers. It is also easy to automate and helps in improving code and development practices. White box testing is sensitive to changes made in code base and can be quite expensive than black box testing.

There are some steps that are followed while performing white box testing. At first, the test scenarios are designed and prioritized based on priority numbers. Then the study of code to examine resource utilization are carried out and the time taken by methods and operations to operate are also studied. Then the handling of the data by the non-public methods and interfaces are checked. After that, testing of control statements and logics are tested. This includes testing loop statements and conditional statements for determining their accuracy for different inputs. At last, security testing is conducted to check every possible security loopholes.

7.5 Maintenance Plan

After a system or software is completed, it should be maintained properly to ensure its functionality for long term. Different types of tools can be used to maintain the changes that may be made in the future. The system is monitored continuously to keep on checking irregularities that may occur in the system. The system's performance and scalability are also monitored on regular basis. If required, some changes are made in programs based on the feedback received from the users. Some bugs may occur or encountered which needs continuous addressing and fixing for which various team members are assigned. Documentation is also kept up-to-date as per the changes made or addition of any new features.

8. Prototypes

8.1 Home Page

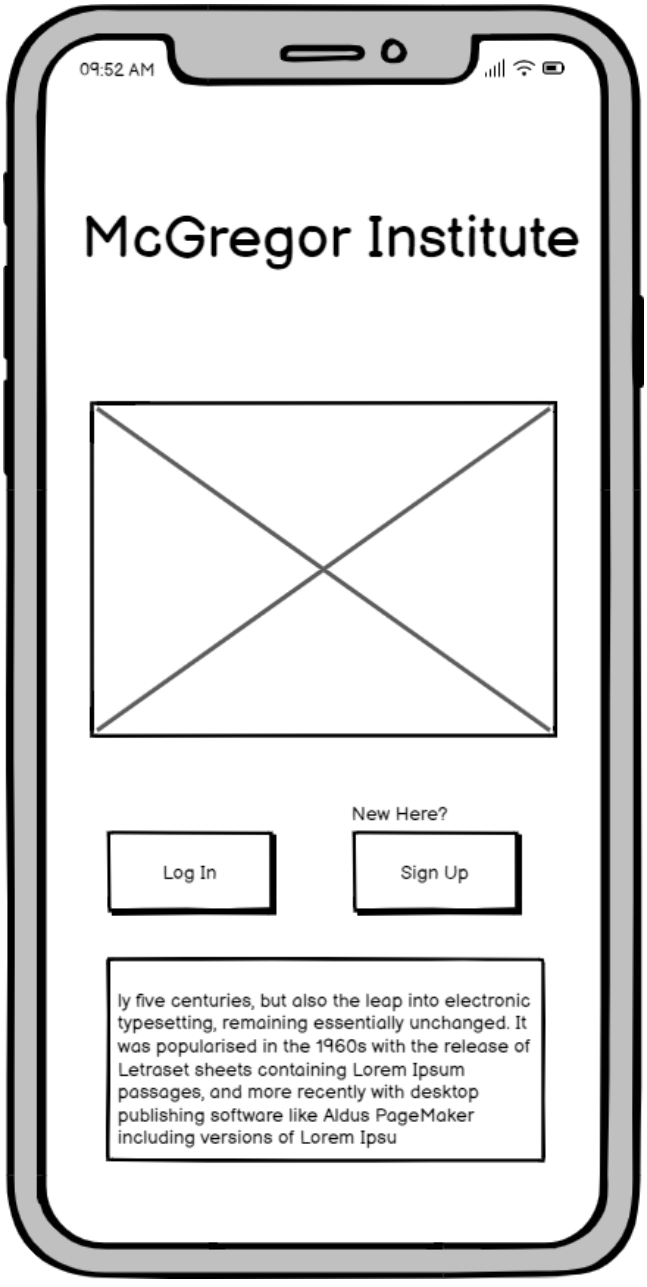

Figure 9: prototype for home page

This is the first page where a new user will be directed to. If the user already has an account in the system, he/she should press log in and if the user is new, he/she should press sign up.

8.2 Register Page

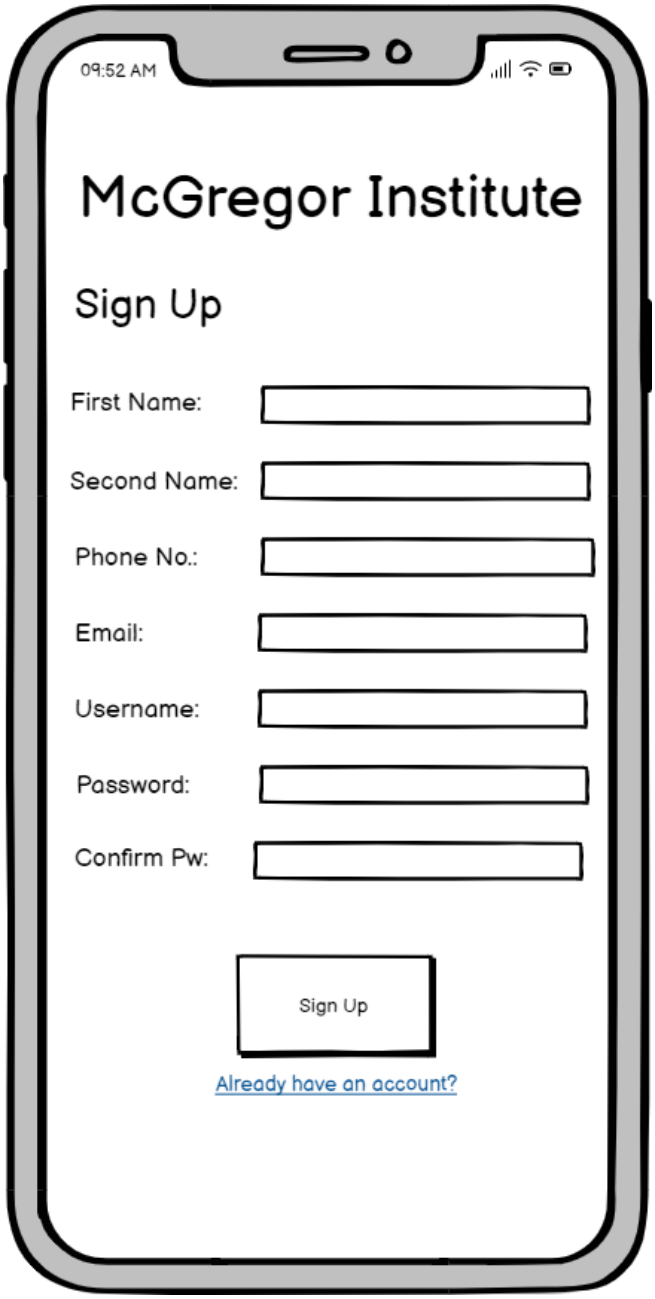

Figure 10: prototype for register page

This is the register page. After pressing sign up in the main home page, the user is directed to this page where he/she will provide all their details and be registered in the system with their username and password.

8.3 Login Page

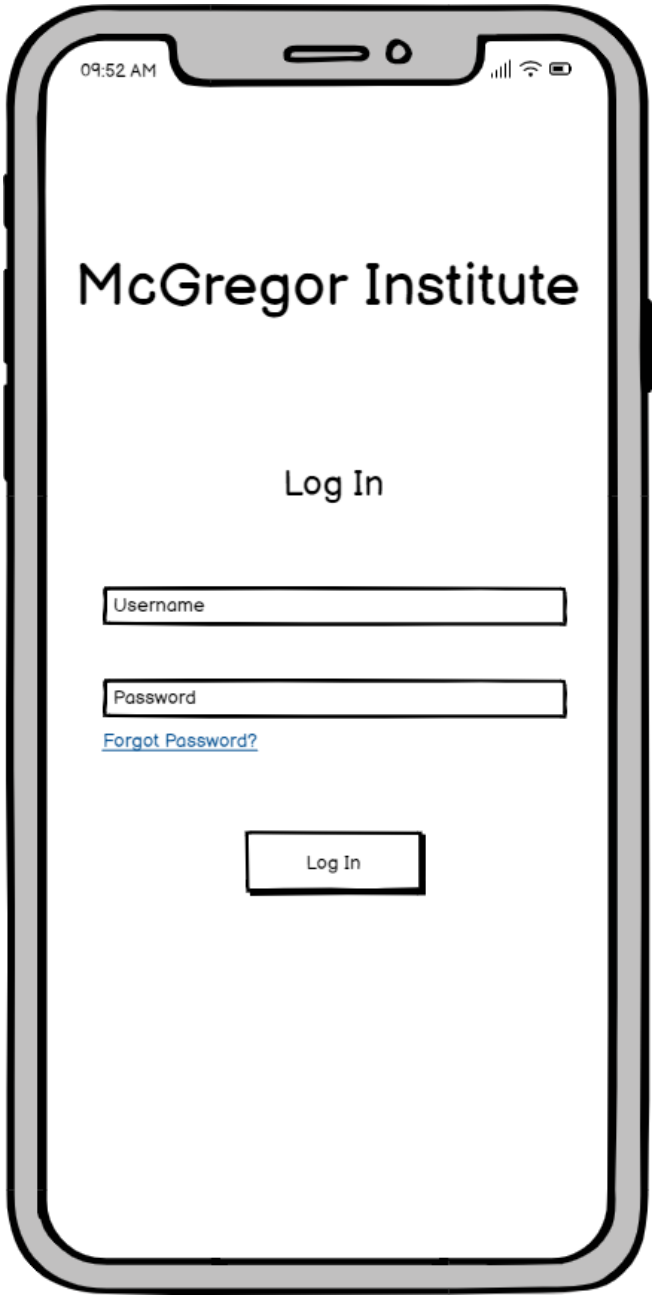

Figure 11: prototype for login page

This is the log in page. After pressing log in in the main home page, the user is directed to this page. Here, the user should log in to the system using his/her registered username and password.

8.4 User Home Page

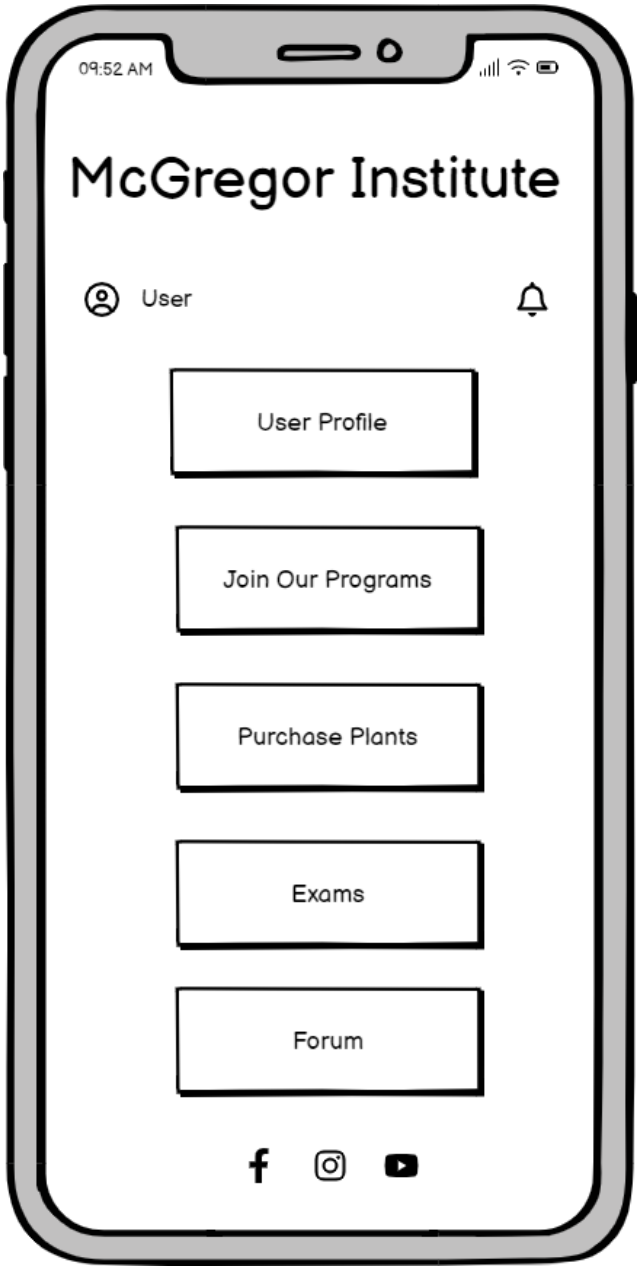

Figure 12: prototype for user home page

This is the user home page. After the user is logged in to the system, he/she is directly redirected to this page. This page contains a lot of options as to where to go next. The user can go to purchase plant, join program, user profile, forum or exam sections. Additionally, the social media accounts of the institute are provided on the footer which are present in every page where the logged in user can go to along with notification button at the top right and user profile picture and user name in the top left.

8.5 Join Program UI

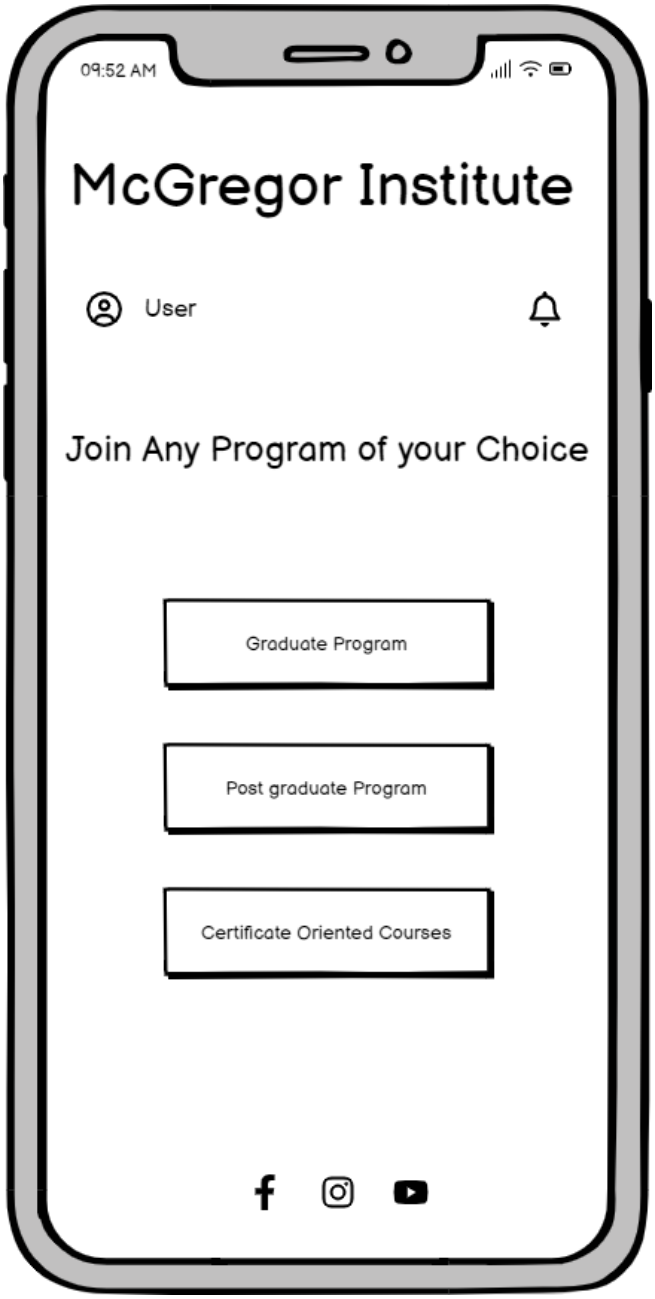

Figure 13: prototype for join program UI

After pressing the “Join Our Programs’ button, the user is directed to this page. Here the user can find the choices of the programs that he/she can join.

8.6 Join the selected program UI

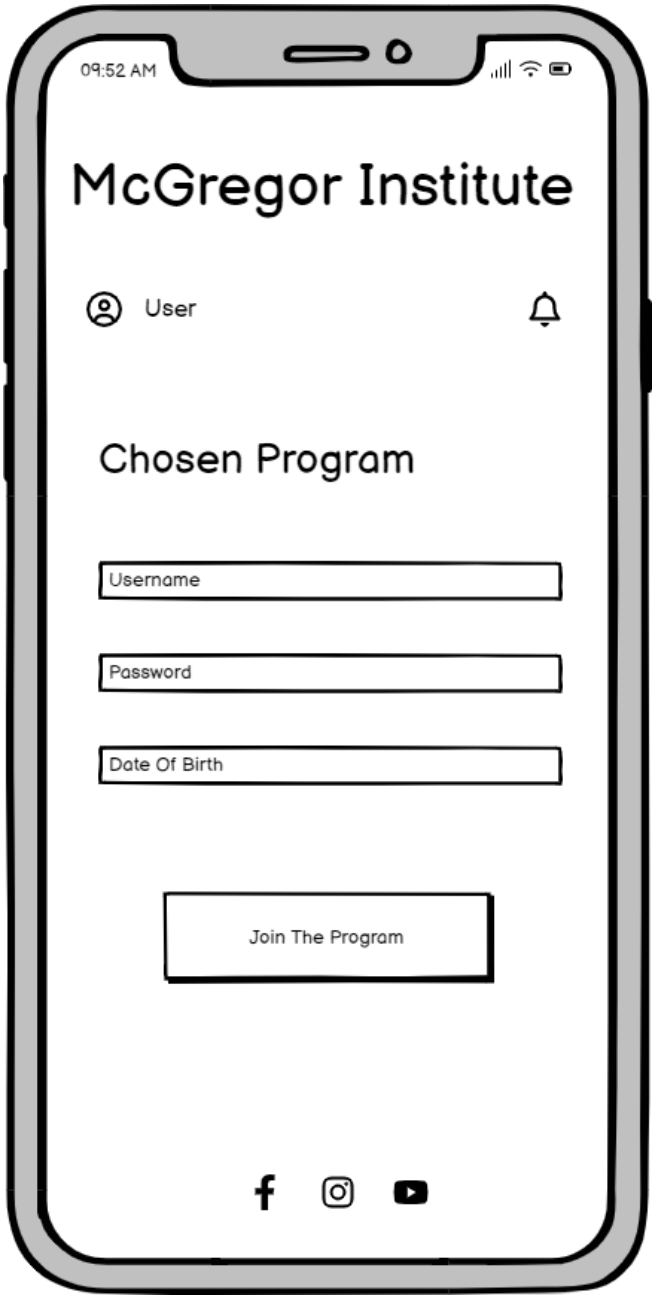

Figure 14: prototype for join the selected program UI

This page comes after the user chooses a program in the join program user interface. The user provides the required details and clicks on “join the program” button to join the chosen program.

8.7 Purchase Plant Interface

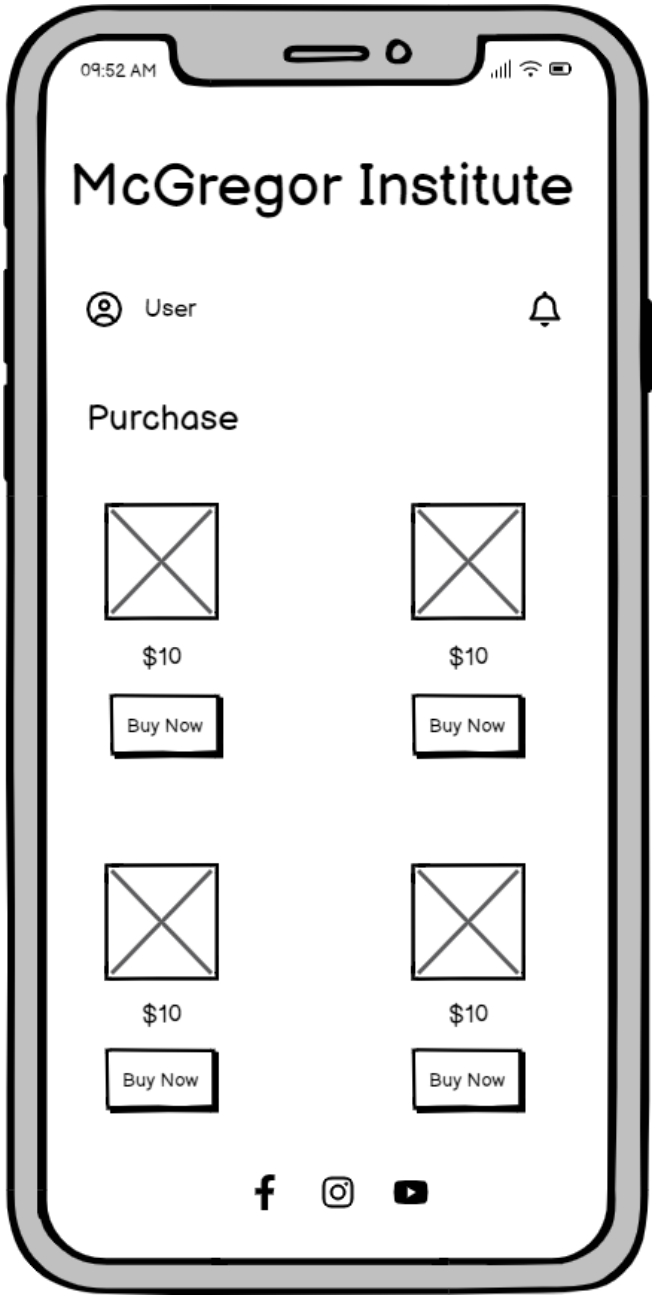

Figure 15: prototype for purchase plant interface

If the user presses “purchase plant” in the user home page, he/she is directed to this page. List of various plants along with their prices are present in this page from where the user can purchase any plant of their choice, if available.

8.8 Payment Interface

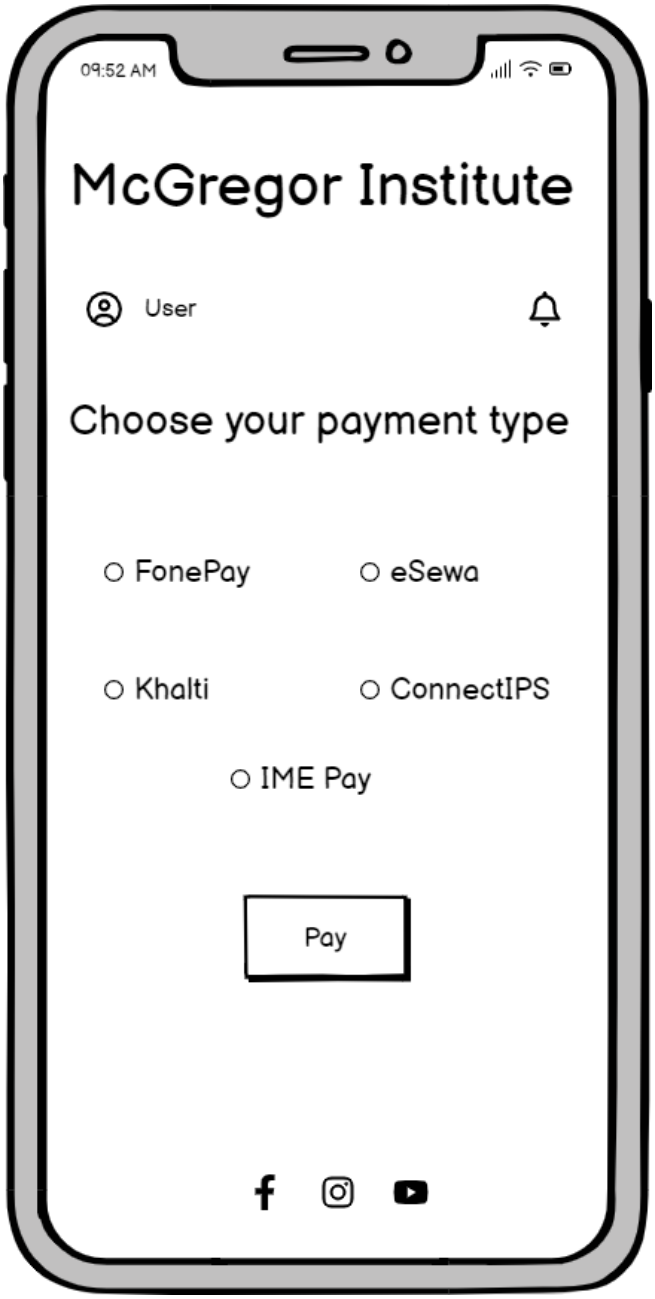

Figure 16: prototype for payment interface

When “join the program” button in join the selected program UI is pressed or when “buy” button in purchase plant page is pressed, the user is directed to this page to complete the payment for the transactions made.

8.9 Give Exam UI

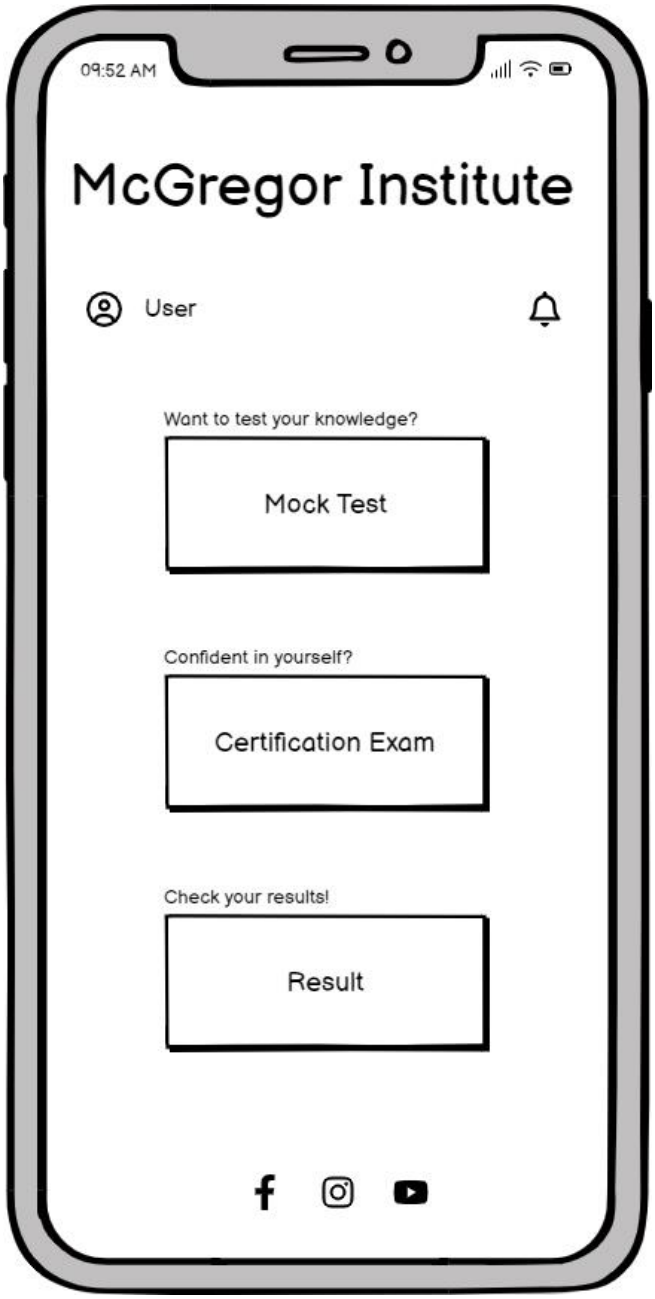

Figure 17: prototype for exam UI

The user is directed to this page when he/she presses “exams” in user home page. This page gives various options to the user according to their needs whether to give mock tests, certification exam or check results.

8.10 Forum Page

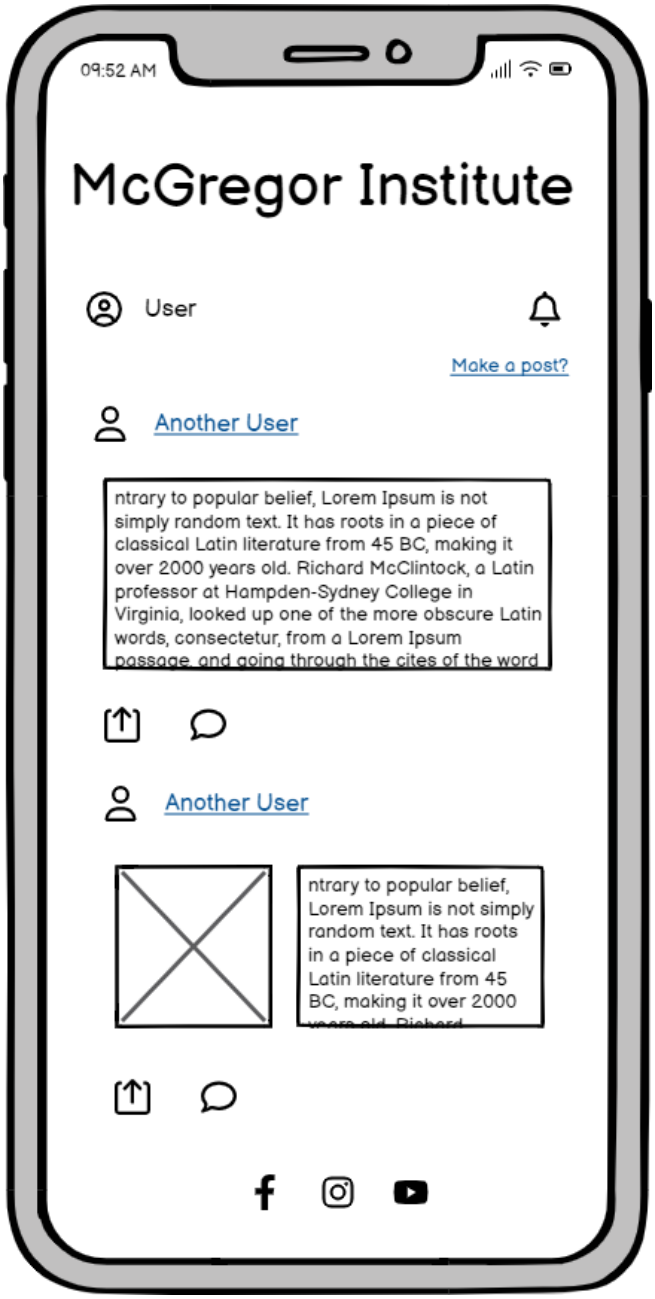

Figure 18: prototype for forum page

User is directed to this page when he/she presses forum in user home page. This page lets users interact with each other through posts and comments. All users can share their opinions via posts and other users can upvote the posts or express their views through comments.

8.11 Admin Panel for Report Preparation

Figure 19: prototype for admin panel for report preparation

This page is the admin panel for gathering data on different topics. To access this page, a dedicated admin account should be logged in onto the system. This page provides options to get data on whether users, employees or financial status.

8.12 Report Preparation page

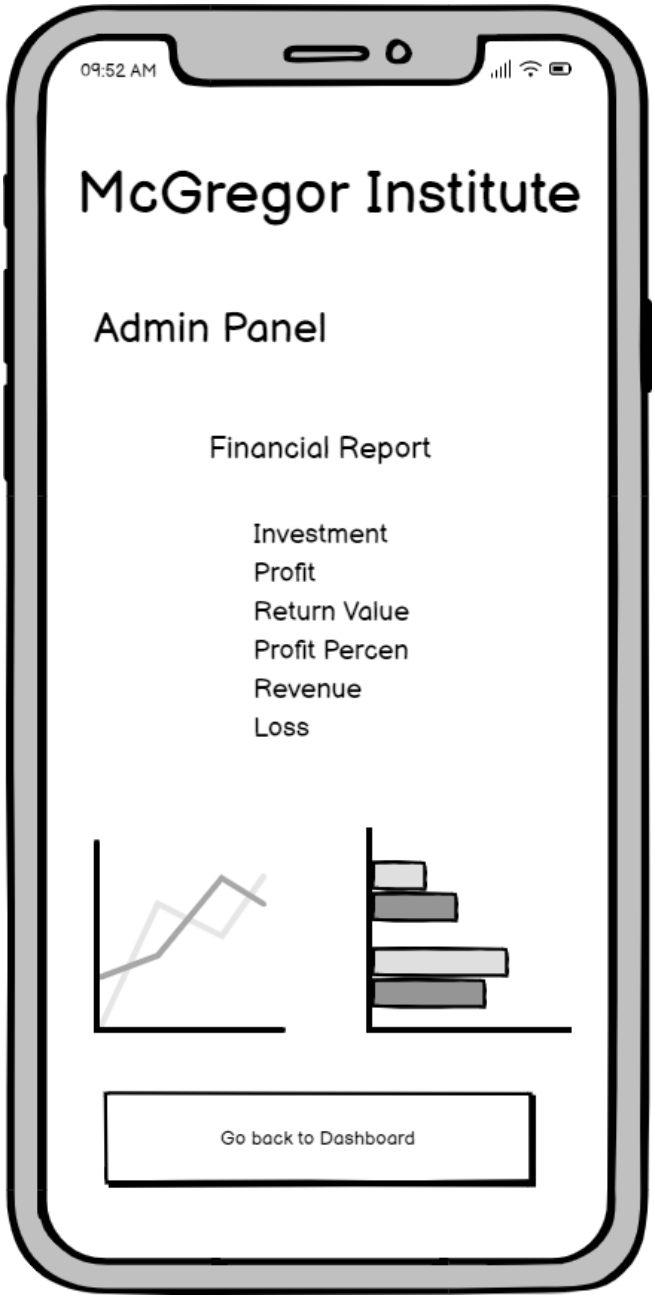

Figure 20: prototype for report preparation page

This is the report preparation page for financial report. The admin is directed to this page after pressing finance in the admin panel for report preparation. The admin can get various financial data of the institute and prepare a solid report on this page.

8.13 Ask Recommendations Interface

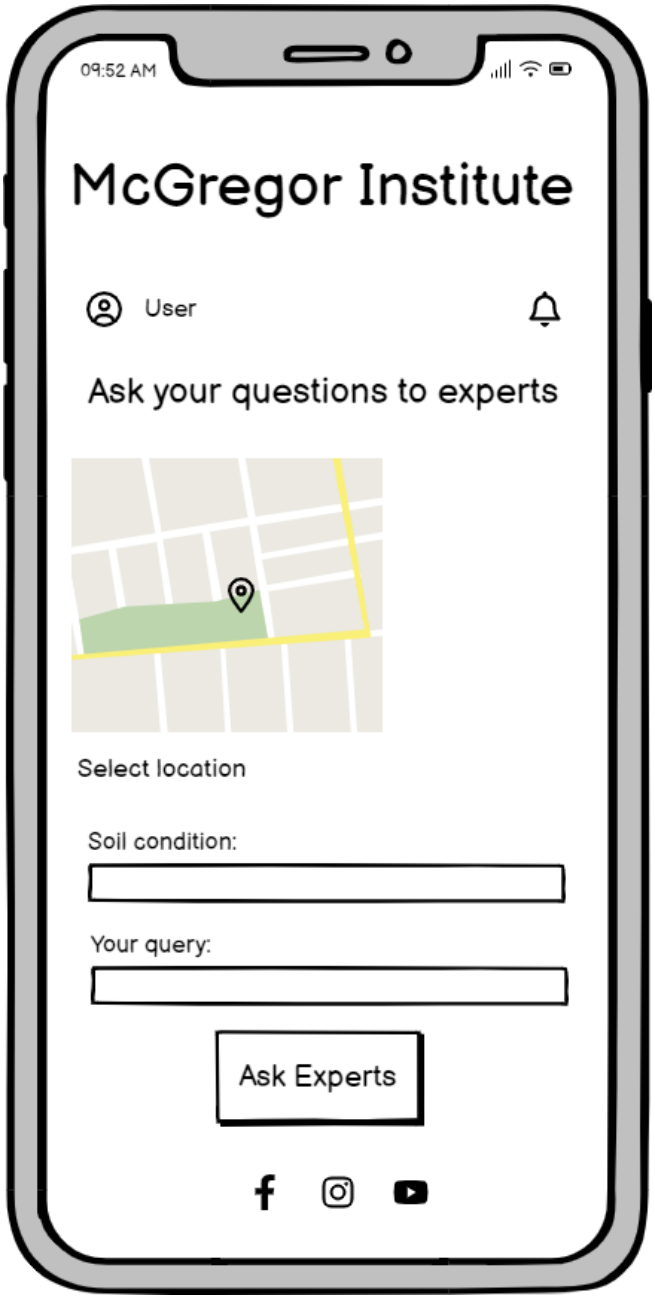

Figure 21: prototype for ask recommendation page

The user is directed to this page after pressing ask recommendations in the user home page. Here, a user can select a location and type in his/her queries for experts to answer.

8.14 Notifications Panel

Figure 22: prototype for notification panel

This is the notification panel. This shows up after the bell icon in the top right is pressed. This section stores all notifications received by the user and displays them in a list.

8.15 Invalid login attempt (pop-up message)

Figure 23: prototype for error messages

This is a pop-up message that is seen when an invalid attempt is made to log in into the system.

9. Conclusion

In conclusion, the overall system for McGregor Institute of Botanical Training is designed meeting all the requirements mentioned in the provided scenario. This includes the system for registering in the system by the user. A user can also join the program of his/her choice through the system. The system has functionalities for purchasing any plants of the user's choice along with simple payment interface to complete the transactions. Users can also give exams and get recommendations directly from experts based on their queries. The system also consists of a forum page where users can interact with each other through posts, comments and upvotes. All the data stored in the system can be managed properly and a full report of users, employees or financial status of the institute can also be made within the system.

For this, all of the required diagrams are completed and development plans are also made following various aspects of software engineering. Upon completion of the coursework, I learned different aspects of software engineering and various ways to utilize them. The coursework helped in getting familiar with different topics related to project management as well as the topics related to designing and development of the system. Various tools were used for completing the coursework including MS word for preparing a full report of the work, draw.io for creating the diagrams present in the documentation and so on to effectively complete the assigned task.

As a result of taking part in this process, I was able to demonstrate my practical understanding of structured software engineering and work really well even on short period of time. The coursework helped me gain valuable experience of working on real life-based scenario which will undoubtedly help me in future in every field.

References

Brush, K. (2024, 04 30). *Use Case*. Retrieved from TechTarget: https://www.techtarget.com/searchsoftwarequality/definition/use-case

Grant, M. (2023, 12 21). *Gantt Charting*. Retrieved from Investopedia: https://www.investopedia.com/terms/g/gantt-chart.asp#:~:text=A%20Gantt%20chart%20is%20a,resources%2C%20planning%2C%20and%20dependencies.