LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

CS5002NI Software Engineering

McGregor Institute

20% Group Coursework

AY 2023-2024

Group Name: Mc Gregor Institute Software Development Team			
S.N.	**Student Name**	**College ID**	**University ID**
1.	**Aaditya Raj Shrestha(L2C2)**	**NP01CP4A220329**	**22068760**
2.	**Dikshya Sharma (L2C2)**	**NP01CP4A220029**	**22067520**
3.	**Binita Bhandari(L2C2)**	**NP01CP4A220036**	**22067525**
4.	**Bishal Bogati(L2C2)**	**NP01CP4A220474**	**22068134**
5.	**Siddhant Bhurtel(L2C2)**	**NP01CP4A220277**	**22068753**

I confirm that I understand my coursework needs to be submitted online via MySecondTeacher 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 mark of zero will be awarded.

Acknowledgement

We would like to express our sincere gratitude to our module leader, lecturer, and tutors for their unwavering help and support throughout our coursework. Likewise, we are also grateful to the RTE Department of our college-“Islington College” for providing such coursework which helps in enhancing our skills.

Likewise, we are especially thankful for the insightful and informative lectures delivered by Mr. Rubin Thapa Sir and the invaluable guidance offered by our tutor Mr. Dipesh Raj Adhikari Sir, which significantly contributed to the enhancement and effectiveness of our work. This coursework would not have been feasible to finish effectively without their guidance.

Moreover, we would also like to thank to all the contributors who shared their valuable knowledge along with the sources that enables us to gather the essential information required for our work. This support not only made it easier for us to complete coursework successfully, but also contributed to improving our capabilities throughout the project.

Abstract

The goal of this assignment is to create a structured software engineering solution for McGregor Institute of Botanical Training, a teaching facility located in Ireland. This coursework describes the goals, requirements, and responsibilities of the project where we are required to work in group to demonstrate practical understanding of structured software engineering (Yourdon). Functionalities including expert suggestions, report preparation, certification examinations, plant purchase, program enrolment, payment processing, and user registration are all included in the complete specification.

The tasks are divided into both group and individual where creating a project charter, a software requirement specification (SRS), and detailed specifications for internal and external models as well as system architecture are among the group's responsibilities whereas individual tasks are focused on specific system functions such as making payments, purchasing plants, report preparation, program enrolment, and certification exams.

The report concludes with a documental structure, including introduction, project charter, SRS, Environmental Model Specification, Internal Model specification, design specifications, assignment diary, individual tasks, and summary. The integration of these elements aims to provide a comprehensive understanding of the proposed system for the McGregor Institute of Botanical Training.

Table of Contents

Table of Figures

Table of Tables

1. Introduction

1.1. Description

This project is a group coursework for the module “CS5002NI - Software Engineering”. The team members are required to perform a structured analysis and design of a system for “McGregor Institute of Botanical Training”. The Ireland based training institute which operates from Godawari, Lalitpur has been working in Nepal for almost 7 years. The institute provides different undergraduate and postgraduate courses related to agriculture along with horticulture specialization.

With the recent increase of the people interested in agriculture, the institute is looking to introduce an online platform that enables the users to view and purchase plants, enrol in courses, take certification exams and engage in discussions regarding plants to name a few functions. The system looks to be a combination of an online store, educational platform as well as a discussion forum that is designed with the intention to retain user engagement for a longer period of time.

The team members are required to demonstrate practical knowledge of Structured Software Engineering and are provided with a set of guidelines. The guidelines include a general and detailed specification of the platform and its functions. The group members are required to work together to create a project charter and SRS. Similarly, each of the members are supposed to create a number of analysis and design specification of a particular part of the system.

1.2. Aims and Objectives

The main aim of this coursework is to show practical knowledge in application of structured software engineering.

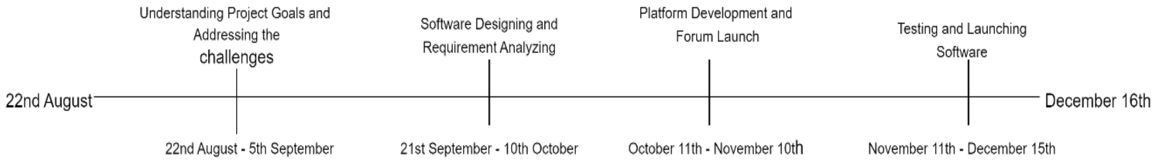
The objectives are:

1. To prepare a sensible project charter and SRS for the given project
2. To work as a group to demonstrate practical understanding of structured software engineering (Yourdon)
3. To create a sensible structured analysis of a system
4. To show the relations among entities
5. To create a proper documentation of the structured analysis process

2. Project Charter

A project charter is a concise, formal document that certifies the existence of a project and gives project managers official permission to start working on it. Before the project is thoroughly scoped out, a project charter paper outlines the project in order to establish a common understanding of its goals, objectives, and resource requirements. Project charters play a crucial role in project management since they serve as a roadmap for the project's fundamentals and serve as a resource for the duration of the project. In order to aid in the approval of the work, the formal document can also demonstrate the project's viability and potential return on investment. (S.Gills, 2023)

Project Charter	
Project Name:	The McGregor Botanical Training System
Project Description:	The McGregor Institute Botanical Training is creating a system where a community of individuals who are interested in plants and horticulture can plan programs to preserve endangered plants and forests and save endangered species and forests. The forum will also serve as a venue for users to post questions and their queries related to botany for the experts to respond to.
Problem Statement:	The main problem that McGregor Institute of Botanical Training is facing is the sudden rise in the number of people that are interested in the field of agriculture and horticulture. The institute is launching short-term certification programs in horticulture to meet this growing demand. Managing and scheduling these new courses effectively is a problem, as is offering a forum for the expanding community of plant enthusiasts to engage, exchange ideas, and consult professionals. The system they intend to put in place will assist them in addressing the difficulty of accommodating and catering to the growing number of people who are interested in plants.
Business Case:	• Fulfilling Market Demand with short-term certification courses • Selling different varieties of plant at minimal fee

	• Improved Community Engagement by building a community of plant enthusiasts. • Improved Environmental Awareness • Improved Knowledge Sharing by promoting collaborative knowledge sharing among plant enthusiasts. Preliminary estimates suggest that it could take approximately 4 months to finish this system and the cost estimated for the completion of this system would be around 100,000 NPR.
Goal Statement:	To create a vibrant plant-lover community in Nepal by offering short-term horticulture certification classes, selling plants at reasonable prices, and starting a lively online discussion board.
Timeline:	This project is anticipated to start on 22nd August, 2023 and be finished by 15th December, 2023. The following are important project milestones: • Understanding Project Goals and Addressing the challenges of the project till 5th September • Software Designing and Requirement Analyzing: September 21st - October 10th • Platform Development and Forum Launch: October 11th-November 10th • Testing and Launching Software: November 11th - December 15th  22nd August Understanding Project Goals and Addressing the challenges 22nd August - 5th September Software Designing and Requirement Analyzing 21st September - 10th October Platform Development and Forum Launch October 11th - November 10th Testing and Launching Software November 11th - December 15th December 16th
Scope:	The scope of creating this system is listed below:

	• Expansion of education offerings reaching a wider audience with short-term horticultural and agricultural certification courses. • Generating revenue by selling varieties of plants at minimal fees • Construct and develop horticultural short-term certification programs for anyone looking for specialized instruction.
Team Members:	Aaditya Raj Shrestha – Project Manager Dikshya Sharma – Cloud Engineer Binita Bhandari – Front-end developer Bishal Bogati – UI/UX Developer Siddhant Bhurtel – Back-end developer

Table 1: Project Charter

3. SRS (Software Requirements Specification):

This SRS portrays the functionalities and features for “the McGregor Institute of Botanical Training” platform. This platform is designed comprehensively, including all the functionalities and capabilities of the system such as different course certification program, plant sales, and community forum.

This document aims to promote and expand agriculture and horticultural knowledge by offering a short-term certification course, facilitating plant sales, and creating a dynamic community forum of individuals passionate about plants in Nepal to all the users registered within the system.

The font style used in this document is Arial with Automatic (Black) Font-Colour. The heading is formatted in a font- size 14, subheadings in a font-size of 13 and the remaining content of documents is formatted in a font-size of 12.

3.1. Functional Requirements:

Functional Requirements are those requirements which specify what a software system must do and how it must function for a provided task which mostly focused on user needs and most fulfil the intended purpose of the system (Jafari, 2020).

The following is a list of the functional requirements for "the McGregor Institute of Botanical Training System":

3.1.1 Users Registration in the System:

The system must provide a service to users to get register by providing a valid user’s credentials and creating a password to access the system’s features.

3.1.2 Program Enrollment:

Anyone interested is welcome to participate in the program. For instance, short-term certification courses or undergraduate, postgraduate, or graduate courses along with both paid and unpaid courses are available for enrollment.

3.1.3 Plant Purchase:

Users can browse a diverse range of plant varieties and make a purchase if they are interested with the available plants. Moreover, they can also store the selected plants in carts and can do a purchase of multiple plants at once.

3.1.4 Payment:

The system should comprise a feature of allowing customers to make payments for both enrolled courses and plant purchases through the system. Moreover, it should be able to record and safely store the specifics information of all these transactions carried out.

3.1.5 Ask for recommendations:

Users have the facility to consult with the experts for the recommendations of appropriate crops or plants for their soil types according to their locations. Also, users can share images of the soil condition and experts will respond accordingly with suitable suggestions. In addition, they can also get personalized recommendations from any agriculture and plant experts who is aware about users' location along with soil type.

3.1.6 Report preparation:

The admin of the system has the capability to generate a detailed financial report, employee's reports and the individual user reports which provide the detailed understanding into financial transactions, employee's performance, and user activities.

3.1.7 Take Certification exams:

The system has provided users with a service to take mock tests as per their convenience and review their results. Likewise, they can also actively participate in certification exams to evaluate their knowledge and skills.

3.1.8 Forum:

Users can actively participate in discussions about plants, express their opinions and findings through posts, and interact by approving other viewpoints and commenting on other findings posts within the platforms.

3.1.9 Get Notifications:

The system should provide notifications to the users regarding relevant information and updates based on their actions and engagements within the system.

3.2. Non- Functional Requirements:

Non-Functional Requirements are those requirements that are not directly concerned with services or functionalities which the system delivers but characterize system behaviours and constraints as a whole. It mostly focused on the system performance and behaviours. They described how the system should perform and behave rather than what it should do (GeeksforGeeks, 2023).

The following is a list of the non-functional requirements for "the McGregor Institute of Botanical Training System”:

3.2.1 Design and Implementation Constraints:

These constraints mostly focused on the design and the development of the system to make the system more feasible along with inheriting all the limitations and restrictions that significantly impact this process of designing and developing the system. These constraints help to set a realistic expectation and create a system fulfilling system objectives with all the resources available. Also, these constraints help to shape the functionality, security, and users’ interactions.

The design and implementation constraints for the “McGregor Institute of Botanical training System” are listed below:

- To use any system services, a user must first register and become a member of the system.

- Detailed records of users, experts, staffs, and admins must be stored in any data store to facilitate effective information retrieval.
- Users and Staffs have no access to alter data within the system.
- Students are free to choose from a wide range of courses, including undergraduate, postgraduate, graduate, and short-term certification programs.
- Students can choose between paid and unpaid courses depending on their preference.
- The system should plan accordingly to run the courses and programs within the predefined budgets and timeline.
- Customers can freely browse plants, make selections, and can purchase if they wish and complete transactions according to their preferences.
- The admin should get access to all institute details, including students, customers, staff, experts, and plants for generating a report accordingly.
- Any users can actively engage in forums for discussions and conversations on diverse topics.
- Users can express their opinions, research findings and discoveries through forum posts and can interact on forums by liking and commenting on each other's posts.
- Customers have the facility to provide feedback, and comments on plants along with their prices and other system services.
- Any available botanical experts can provide recommendations to users on crops, plants, and soil types based on user location.
- The system should notify all the users in case of any changes made within the system through notifications facilities.

- The system’s data should be regularly backed up to ensure its security and reduce data leakage.
- Students and customers have the facility to inform the system in case of cancellations of registration by users and plant purchase by customers.
- The system should set up some policies and procedures for users, staff, and customers.

3.2.2 External Interface Required:

External Interfaces are the important constraints that determine how a system communicates with the external environment. It includes the set of functionalities and protocols along with various requirements that makes external communication more interactive.

The external interface required for the “McGregor Institute of Botanical training System” are listed below:

3.2.2.1 User Interfaces:

The user interface enhances the system services, so it should be pleasant and user friendly for easy interactions. The user interface design prioritizes visual appeal and simplicity whether it is accessed via desktop, mobile, or web platforms.

Additionally, through the user interface users can easily navigate in the system for using features like registration, purchase of plants, enrollment in courses, participations in forum and so on.

Overall, the system prioritizes for a smooth and efficient user experience by using responsive design in order to make interface fun and accessible on variety of devices.

3.2.2.2 Hardware Interface:

Hardware Interface mostly involves the various physical components required for a system that are essential to support its operation smoothly. The system must be compatible with the standard hardware configurations used to ensure accessibility across diverse devices.

Here, a system is designed to work smoothly with different types of physical equipment including computers, servers, mobile phones. The system must include a powerful processor, CPU (Central Processing Unit), RAM (Random Access Memory) and SSD (Solid State Drive) or HDD (Hard Disk Drive) with larger storage capacity to ensure that the system operates well on various devices like smartphones, laptops, and so on.

3.2.2.3 Software Interface:

Software Interface comprises of the software tool that helps in defining various functionalities and features of the system outlining its performance, and what a software should do within a system.

Here, a system used software interface to connect to a number of applications, such as database for users, courses, plants, staffs, and other so on to store their respective data along with external services for handling payment and many more. Its integration in system facilitates effective communication, ensuring smooth interaction with a range of software components along with enhancing overall system’s functionality.

3.2.2.4 Communication Interfaces:

Communication Interface involves in the communication between a system and its external entities. It looks after how it interacts with those external entities like users, experts, and other entities within the system.

Here, a system uses communication protocols like HTTP (Hyper Text Transfer Protocol)/HTTPS (Hyper Text Transfer Protocol Secure) for web interactions and integrating with external services through APIs (Application Program Interface). It ensures efficient communication both inside and outside the system to facilitate user notifications, forum updates and interactions with other systems. So that the information flows in the system through this interface.

3.2.3 Other Non-Functional Requirements:

Those requirements are not directly concerned with the system but play role in enhancing the systems. Some of the other non-functional requirements for "McGregor Institute of Botanical training System" are:

3.2.3.1 System Performance:

This system prioritizes higher speed performance along with fast responsiveness to the task asked to be performed by any users within the system in order to provide a smooth and simple user experience.

3.2.3.2 Scalability:

The system is designed accordingly to handle a growing number of users accommodating the provision of courses, and selling plants over a time, ensuring smooth scalability and efficient management of increasing demand of the users.

3.2.3.3 Reliability:

The system's reliability is essential for implementing diverse range of services including user registration, course enrollments, plant purchases, ensuring accuracy and uninterrupted processing's under all circumstances.

3.2.3.4 Availability:

The system should be made accessible for every time i.e. 24/7 and for performing various activities, such that user can access various services whenever they require them.

3.2.3.5 Maintainability:

The system should be maintained on a regular basis so that system run smoothly whenever used by the user. The system should be tested, debugged, and modified on a regular basis with minimal downtime.

3.2.3.6 Portability and Compatibility:

The system's portability and compatibility are essentials to ensure, it's supported across diverse devices and platforms which facilitates smooth data transmission and accessibility, and enabling user to easily interact with the system effortlessly regardless of any platforms in order to maintain easy data access, promoting a user-friendly experience across different platforms and user can consistently access the system.

3.2.3.7 System Security:

The system security is essential to secure sensitive information, such as personal and financial data of users which have chances to be decrypted during data transmission and storage. So, the system must integrate strong authentications procedures to prevent unwanted access and guarantee the security of users and financial data within the system.

3.2.3.8 Usability:

The system should provide a proper interface along with user-friendly dashboards for the user to access the services properly with clear navigation and instructions.

3.2.3.9 Documentation and Training:

The system should have proper documentation including system architecture guides and user manuals, to promote efficient system understanding.

Additionally, regular training sessions must be provided for the staff to enhance the efficient system utilization along with skilful use of the system.

3.2.3.10 Data Backup and Recovery:

The system's data should be regularly updated and should be backed up in order to address all the possibles failures and malfunctions. If in case of system failure, it should have the provisions of recovering the data. So, Data should be regularly backup.

3.2.3.11 Regulatory Compliance:

The system should maintain the provision of regulatory compliance from the users to enhance the features of the system for user convenience along with data protection, privacy, and regulatory standards. This helps to mitigate the various risks that may arise in the system and helps to maintain trust with the users using the system.

3.2.3.12 Auditability and Performance Monitoring:

The system should maintain all the user's activity logs and should monitor all the mechanisms of the system performance enhancing the accountability and transparency within the system.

Additionally, the system performance should be regularly monitored to ensure the system’s smoothness along with maintaining a high-performing system by identifying the areas for improving the system and ensuring user satisfaction.

3.3. Goals of Implementation:

The several goals of implementing this “McGregor Institute of Botanical training System” are listed below:

- To implement an efficient registration system for users and utilize system features effectively.
- To facilitate easy enrollments in various certification courses for those individuals interested in horticulture education.
- To implement a feature of getting expert recommendations based on locations and soil conditions.
- To develop a user-friendly platform for facilitating easy plant selection, efficient cart management, and convenient bulk purchase for varieties of plants.
- To ensure a secure and efficient payment processing system for both plant purchase and course enrollments.
- To enable users to engage in forum for discussion and sharing knowledge and interacting within the system.
- To get the users feedback along with their compliance to improve system functionality and enhancing marketing strategies.
- To generate a detailed financial, staff and users-related report by staffs in proper format and updating the users through notification.

To wrap up, the SRS of “McGregor Institute of Botanical training System” sets the foundation for an adaptable dynamic platform designed with specified purpose. This system provides various botanical courses which are paid or unpaid along with maintaining a system to purchase varieties of plants and interact within the forum. Moreover, this document guides and serves as a vital reference for developers, stakeholders, and end users updating the development and implementation of an innovative botanical training program.

4. Group Task

4.1. Environmental Model Specification

The environmental model specification for this system is divided into 3 parts. They are context level diagram (level 0 DFD), level 1 DFD and level 2 DFD.

Data Flow Diagram (DFD):

A data flow diagram is a visual representation of the data flow of a system. It gives the viewer an insight on the entities, the flow of data and the processes involved in the system. In this system analysis and design process we follow the Yourdon approach. The four basic elements used in the construction of a DFD through Yourdon approach are:

1. **Process**: A process is a function or operation in the system that changes the data or produces an output. It is represented by a circle.
2. **External Entity**: An external entity is an outside system that sends of receives data with the system. It is represented by a rectangle.
3. **Data store**: A data store is a file or repository that stores data and information for future use. It is represented by open-ended rectangle.
4. **Data flow**: Data flow is a route that shows the path that the data travels in a system. The data flow shows the route of data traveling through various processes, entities, and data stores. It is represented by an arrow.
   (Lucidchart, 2024)

4.1.1. Whole System Context Diagram

A level 0 Data Flow Diagram (DFD), also known as a context level diagram, is a visual representation of the basic overview of the whole system. It is a general and simple representation of the system such that the viewer i.e. stakeholders, business analyst, developers, etc. should be able to understand the core of the system at a single glance. (Lucid Software Inc., 2024)

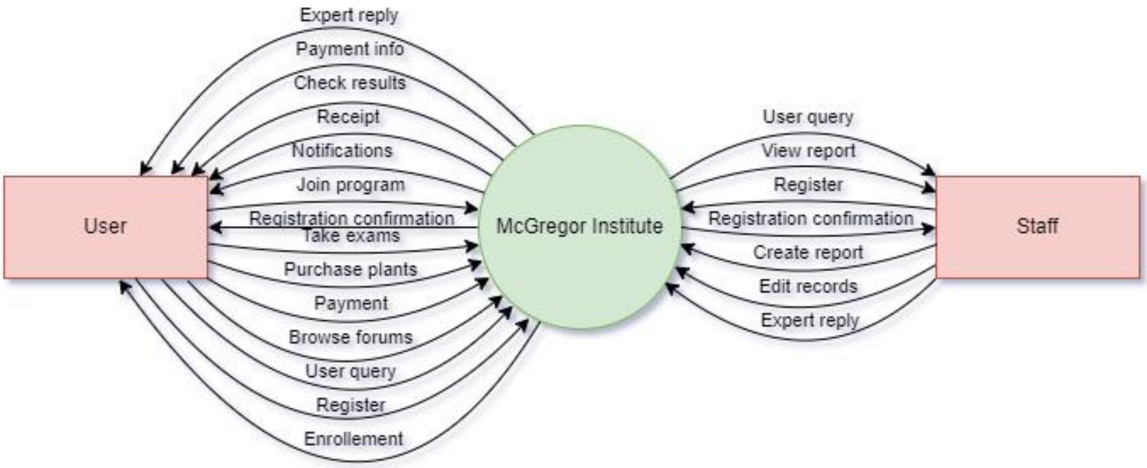


Figure 1: Context Level Diagram(level 0) of the system.

4.1.2. Level 1 Data Flow Diagram

A level 1 Data Flow Diagram breaks down the main process in the level 0 diagram into sub-processes and the data flow and data stores that are associated with each process is also added. Each process in this level are shown as a separate process. (GeekforGeeks, 2024)

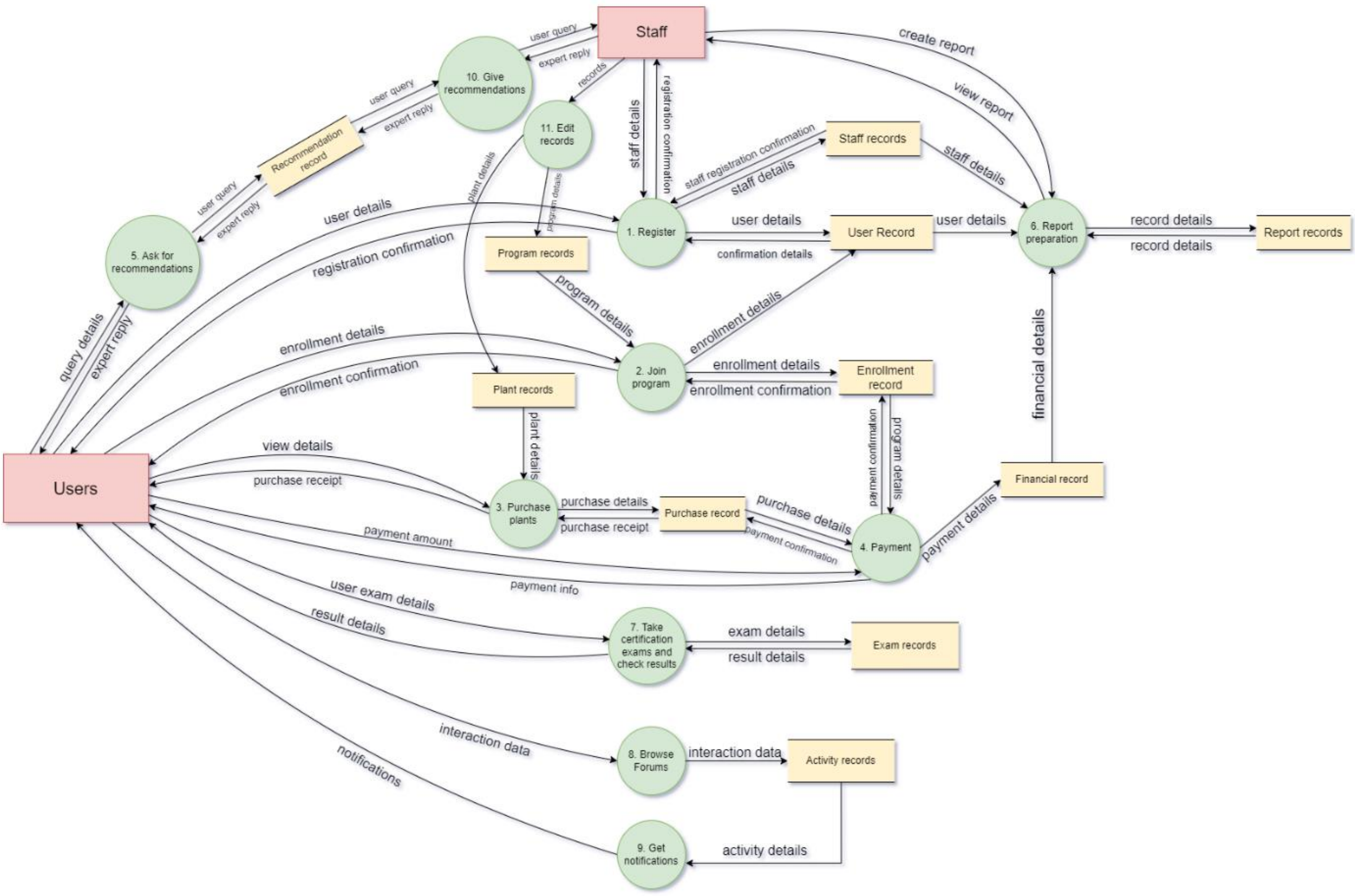


Figure 2: Level 1 DFD of the system.

4.1.3. Level 2 Data Flow Diagram

A level 2 DFD further breaks down the level 1 DFD. The processes are further divided into sub-processes and the data flows and data stores that are associated with each individual sub-process are shown. In a level 2 DFD diagram, each sub-process emerges as a distinct entity and the data flows are data stores are intricately mapped. (Visual Paradigm, 2023)

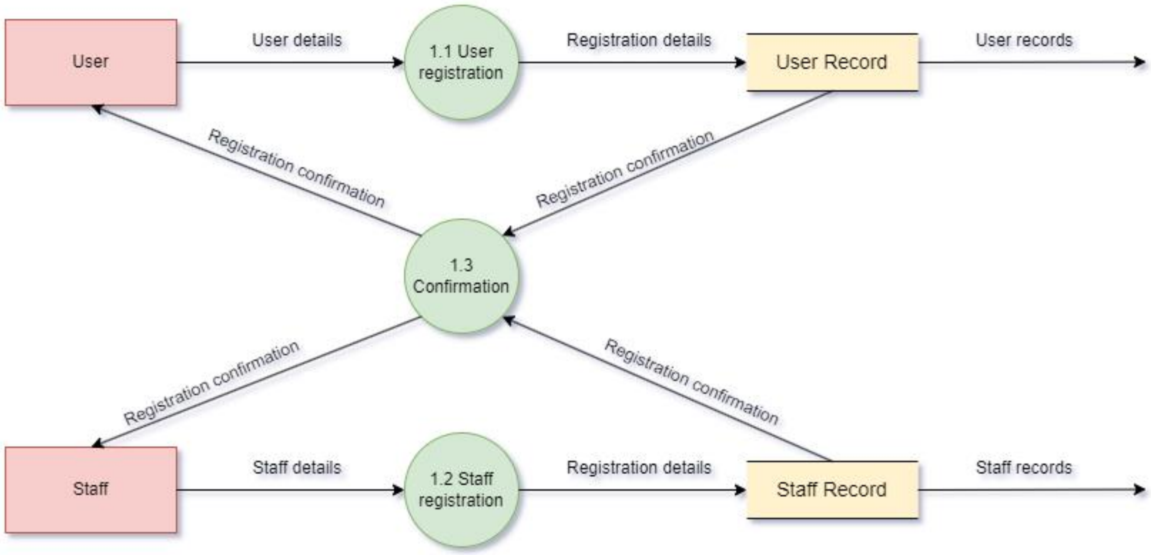


Figure 3: Level 2 DFD of the system (User Registration).

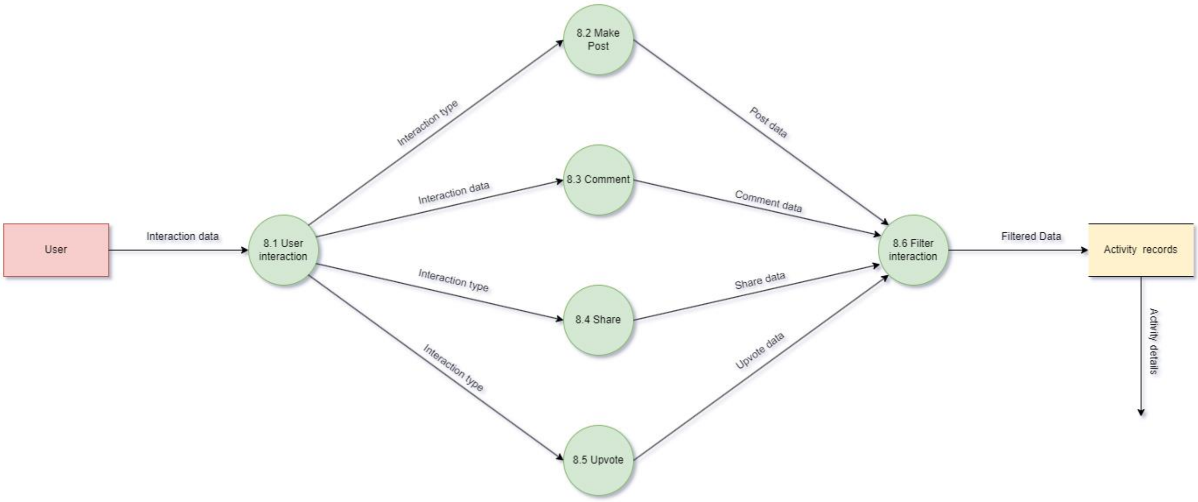


Figure 4: Level 2 DFD of the system (Activity).

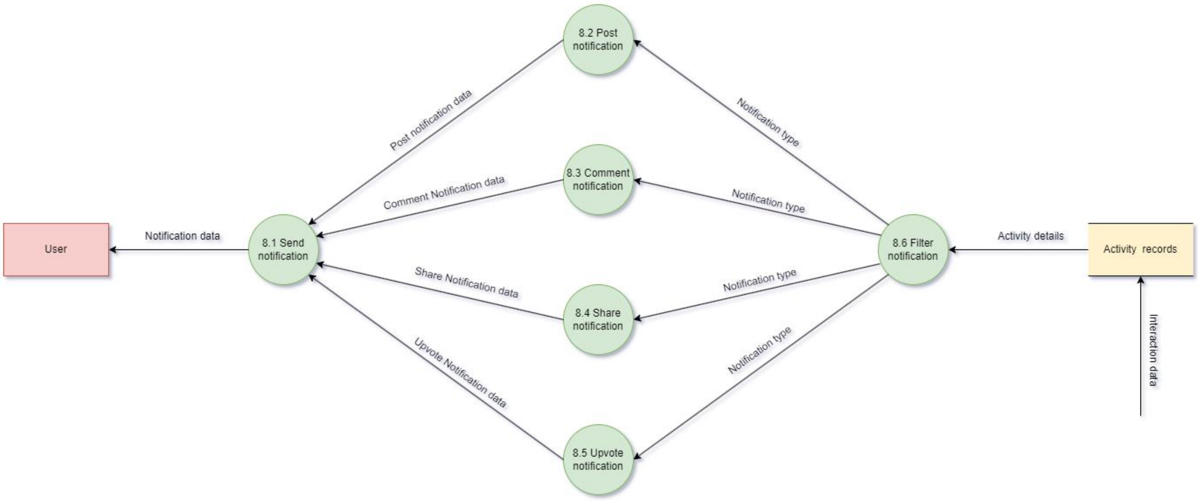


Figure 5: Level 2 DFD of the system (Notifications).

4.2. Internal Model Specification

The Internal Model Specification shows the detailed documentation presenting a comprehensive overview of the system's internal model, outlining its purpose and its significance of the system. Key components including the Entity Relationship diagram, Data dictionary and process specification of the system are detailed within this section.

Together these components contribute to a detailed understanding of the internal model by clarifying its complexities and providing a sophisticated understanding of its functions.

4.2.1. Entity Relationship Diagram

Entity Relationship Diagram (ERD) is a graphical representation of an information system that shows the relationships between people, object, places, concepts, or events within that system. It is a conceptual and representational model of data that illustrates the entity framework structure. For a database design to be successful, an entity-relationship diagram (ERD) is essential hence, a conceptual design for a database can be developed using ERD as a high-level logical data model (Rouse, 2017).

The Elements of ERD are as follows:

- **Entity**

  Entity is a real-world object that has its certain elements called attributes that defines the nature of an entity. It is equivalent to database tables in a relational table in a relational database. An entity is represented by a rectangle with its name at the top in crow's foot notation.

- **Attributes**

  Attributes are the characteristics properties that collectively defines an entity. Attributes are represented inside a box underneath the entity in crow's foot notation.

- **Relationship**

  Relationships is the association that indicates how entities interact with each other. Relationships are represented as straight line in crow's foot notation which has a name expressed as verb on its line to show the relationship between entities. Relationship also consists of two indicators that are cardinality and modality (Nalimov, 2020).

The following ERD is created for the system with the following entity and attributes.

1. **User:** User_ID (PK), First_Name, Last_Name, Contact, Email, Registration_Date
2. **Program:** Program_ID (PK), Program_Name, Program_Type, Program_Fee
3. **UserProgram_Enrollment:** UserProgram_ID(PK),Enrollment_-Date, User_ID(FK), Program_ID(FK)
4. **Plant:** Plant_ID (PK), Plant_Name, Plant_Type, Plant_Price, Availablity_status
5. **UserPlant_Purchase:** UserPlant_ID (PK), Quantity, Purchase_Date, User_ID (FK), Plant_ID (FK)
6. **Exam:** Exam_ID (PK), Exam_Date, Exam_Details
7. **UserExam_Report:** UserExam_ID (PK), Exam_Score, User_ID (FK), Exam_ID (FK)
8. **Activity:** Activity_ID (PK), Activity_type, Date, User_ID (FK)
9. **Staff:** Staff_ID (PK), Staff_Name, Staff_Type, Staff_Email, Staff_Contact
10. **User_Staff_Recommendation:** UserExpert_ID (PK), Recommendation_Date, Recommendation_Detail, User_ID (FK), Staff_ID (FK)

11. **Payment:** <u>Payment_ID (PK)</u>, Payment_Date, <u>UserProgram_ID (FK)</u>, <u>UserPlant_ID (FK)</u>

12. **Report:** <u>Report_ID (PK)</u>, Report_Type, Report_Date, <u>User_ID (FK)</u>, <u>Staff_ID (FK)</u>, <u>Payment_ID (FK)</u>

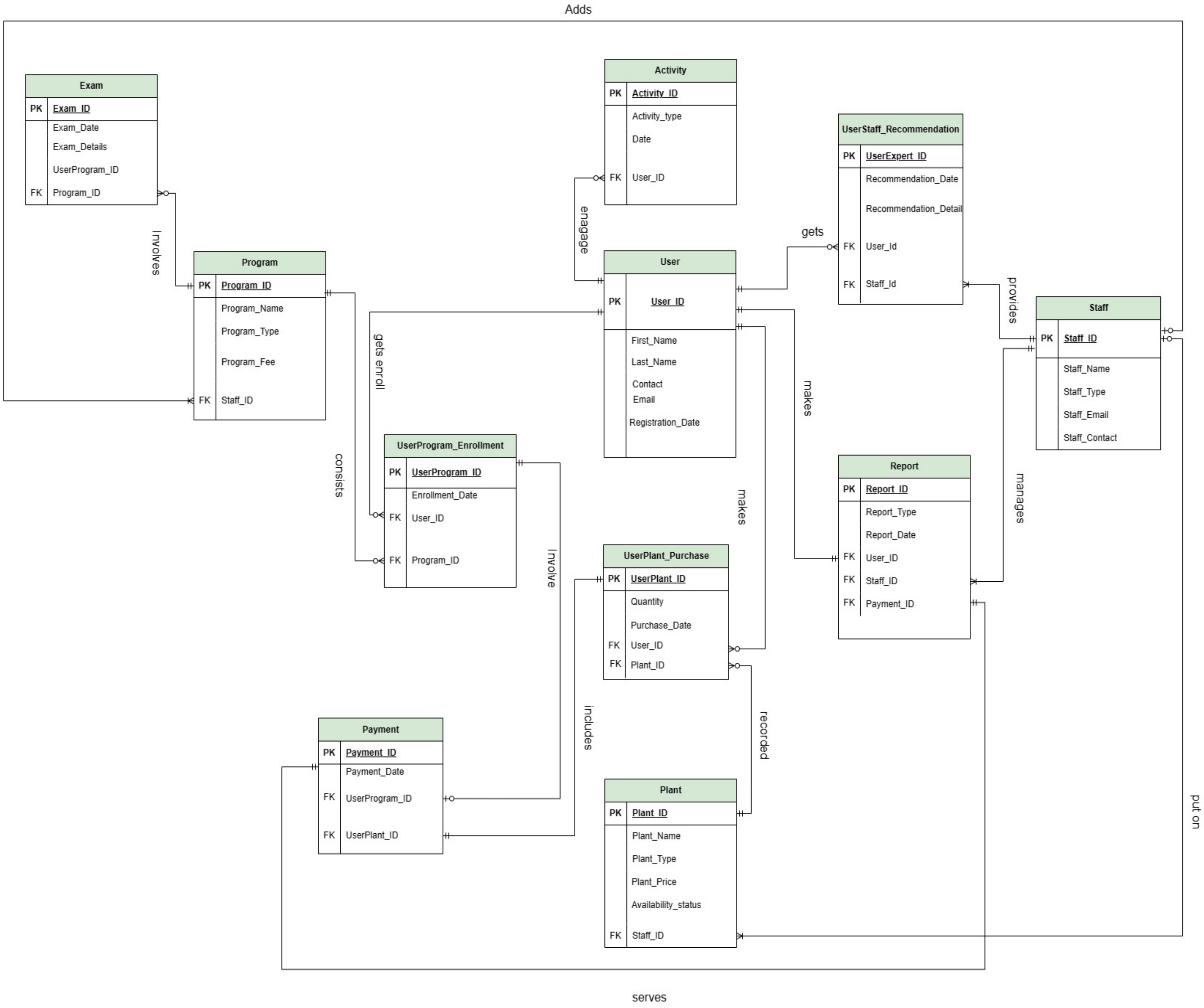


Figure 6: Entity Relationship Diagram of the system.

4.2.2. Data Dictionary

A data dictionary is a collection of details about every sort of data element or records item, including data types and system textual descriptions. It is a file that contains a database's metadata which also holds information about data ownership and the relationship between the data and other objects. A data dictionary is an important part of a relational database as it provides additional information about the tables and its attributes which helps to manage data in a neat and well-organized way (Varma, 2023).

<u>Data Dictionary of the Entire System</u>

User details = input variables

User registration = command

Registration confirmation = Output details

Enrollment details = input variables

Enrollment = command

Join Program = input variables

User record = data store containing records of registered and enrolled users

Program record = data store containing records of courses added by staffs

Plant details = input variables

Purchase = command

Plant record = data store containing records of plants available added by staff

Purchase record = data store containing all the records of plant purchased

Payment = input/output details

Payment = Boolean

Financial record = data store containing records of all the transactions performed

Ask for recommendations = input

Give recommendations = output

Recommendation record = data store containing records of expert advise

Staff details = input

Staff record = data store containing records of experts and admins

Report preparation = command

Report record = data store containing records of all the reports that are generated by staffs

Exams = input/output

Exam records = data store containing records of all the records of exam and result details of user

Interaction data/ activity = input

Browse forums = command

Activity records = data store containing records of all user interaction and also the notifications of the activity

1. **User =** User_ID + First_Name + Last_Name + {Contact}2+ Email + Registration_Date *data store*

   User_ID = Integer
   First_Name = Varchar
   Last_Name = Varchar
   Contact = Varchar

Email = Varchar

Registration_Date = Date

- User record = {new user registration record} *
- Status = [“User registered”,” not registered”]
- Register user = Command

2. **Program =** Program_ID+ Program_Name + Program_Type+ Program_Fee *data store*

   Program_ID = Integer

   Program_Name = Varchar

   Program_Type = Varchar

   Program_Fee = Float

   - Program record = {stores records of the programs} *
   - Status = [“program started” ,“not started”]

3. **UserProgram_Enrollment =** UserProgram_ID + Enrollment_-Date *data store*

   UserProgram_ID = Integer

   Enrollment_Date = Date

   - UserProgram Enrollmant = {stores records of the user enrolled in a program}
   - Status = [“User enrolled”, “not enrolled”]
   - Enroll program = command

4. **Plant =** Plant_ID + Plant_Name + Plant_Type + Plant_Price + Availablity_status *data store*

   Plant_ID = Integer

   Plant_Name = Varchar

Plant_Type = Varchar

Plant_Price = Float

Availablity_status = Varchar

- Plant record = {stores records of the plant} *
- Status = [“plant available”, “not available”]

5. **UserPlant_Purchase=** UserPlant_ID + Quantity + Purchase_Date

   UserPlant_ID = Integer

   Quantity = Integer

   Purchase_Date = Date

   - User Plant Purchase = {stores records of the plant purchased by a user}
   - Status = [“plant purchased”, “not purchased”]
   - User Plant Purchase = command

6. **Exam =** Exam_ID+ Exam_Date + Exam_Details

   Exam_ID = Integer

   Exam_Date = Date

   Exam_Details = Varchar

   - Exam record = {stores record of the exams and results} *
   - Status = [“exam enrolled”,” not enrolled”]
   - Exam enrollment = command

7. **Activity =** Activity_ID + Activity_type + Date

   Activity_ID = Integer

   Activity_type = Varchar

   Date = Varchar

   - Activity record = {stores record of user’s activity and activity notifications} *
   - Status = [“engaged”,” not engaged”]
   - Interact = command

8. **Staff:** Staff_ID + Staff_Name + Staff_Type + Staff_Email + Staff_Contact

   Staff_ID = Integer

   Staff_Name = Varchar

   Staff_Type = Varchar

   Staff_Email = Varchar

   Staff_Contact = Varchar

   - Staff record = {new staff registration record} *
   - Status = [“staff registered”,” not registered”]
   - Register staff = Commands

9. **User_Staff_Recommendation =** UserExpert_ID + Recommendation_Date + Recommendation_Detail

   UserExpert_ID = Integer

   Recommendation_Date = Date

   Recommendation_Detail = Varchar

- recommendation record = {records data store of user query and expert reply} *
- status = [“replied”, “not replied”]
- Ask recommendation = Commands

10. **Payment =** Payment + Payment_Date

    Payment_ID = Integer

    Payment_Date = Date

    - Payment record = {datastore of plant purchase payment and course enrollment payment} *
    - Status = [“paid”, “not paid”]
    - Payment = command

11. **Report =** Report_ID, Report_Date , Report_Type

    Report_ID = Integer

    Report_Date = Date

    Report Type = Varchar

    - Report record = {datastore of reports that is financial record, staff record, user record} *
    - Status = [“report generated “, “not generated”]
    - Generate Report = command

4.2.3. Process Specification

The procedure for recording, examining, and elucidating the formulas and decision-making logic utilized to produce output data from process input data is called a process specification. Its goal is to outline engineering and regulatory requirements and processes in a flow-down manner. Process specifications must be comprehensive and unambiguous in order to produce consistent, high-quality data. A process specification eliminates uncertainty, making it possible for a person or organization to confirm system design—including data flow diagrams and data dictionaries—and get an accurate account of tasks and accomplishments that have been carried out. (Rouse, 2023)

The process specification of this system are as follows:

Level 1 processes:

Number: 1

Title: Register

Description: This process receives user details and staff details for registration confirmation and stores user details in user records datastore and sends back the registration confirmation to user. Similarly, it stores staff details in staff records datastore and sends back the registration confirmation to the staff. It allows the users and staff to be registered into the system.

Input Data Flow: User Details + Staff Details + Staff registration confirmation + Confirmation details+ User Record (datastore) + Staff record (datastore)

Output Data Flow: Registration Confirmation (To User) + Registration Conformation (To Staff) +User details + Staff details + Staff records (datastore) + User record (datastore)

Process Logic:

- Take user details from the user.
- Store it in User Record datastore
- Takes staff details from the staff
- Stores it in the Staff records datastore
- Allows the user and staff to register in the system
- Sends back the registration details to the user.
- Sends back the registration details to the staff

Number: 2

Title: Join Program

Description: This process receives the user enrollment details from the user and stores the enrollment details in the user record datastore and the program details are stored in the enrollment record datastore and the enrollment conformation are sent back to the user. Similarly, it receives program details from the program records datastore. It allows the user to enroll in their selected program.

Input Data Flow: Enrollment Details + Program Details + Enrollment Confirmation + Enrollment Record (Datastore) + Program Records (Datastore)

Output Data Flow: Enrollment Confirmation + Enrollment details (To User Record Datastore) + Enrollment details (To Enrollment Record Datastore) + User Record (Datastore) + Enrollment Record (Datastore)

Process Logic:

- Collects the enrollment details from the user.
- Sends it in the user record datastore.
- Collects program details from the program records datastore
- Enrolls the user into their selected program
- Sends the enrollment details in user record

- Sends the program details and user details in the enrollment record datastore

Number: 3

Title: Purchase Plants

Description: This process sends the plant details and view details of the user and stores the purchase details in the purchase record datastore and the user receives a purchase receipt. It allows the user to purchase plants.

Input Data Flow: View Details + Plant Details + Purchase Receipt + Purchase record (Datastore) + Plant Records (Datastore)

Output Data Flow: Purchase Receipt + Purchase details + Purchase Record (Datastore)

Process Logic:

- Collects Plant Details from the plant records datastore
- Collects view details from the user
- Allows user to purchase the plants
- Stores the purchase details in the purchase record datastore
- Generates a Purchase Receipt to the user

Number 4:

Title: Payment

Description: This process receives the purchase details of the user from the purchase record datastore and returns payment confirmation detail, stores payment confirmation to enrollment record datastore and receives program details, store payment details to financial record datastore. Similarly, it returns the payment info to the user and receives the payment amount. It finally allows the user to pay for the enrolled program or the purchased plants.

Input Data Flow: Payment Amount + Purchase Details + Enrollment record (datastore) + Program Details + Purchase Record (datastore)

Output Data Flow: Payment Info + Program Details + Payment Conformation (To purchase record datastore) + Payment Conformation (To enrollment record datastore) + Financial record (datastore)

Process Logic:

- Receives purchase details from the purchase records datastore.
- Receives program details from enrollment records datastore
- Stores the payment confirmation in the purchase record datastore and enrollment records datastore.
- Stores the payment details in the Financial Record
- Sends the payment info to the user
- Receive payment amount from the user

Number: 5

Title: Ask for recommendation

Description: This process collects query from users and store it in a recommendation record datastore and user receives reply from the expert

Input Data Flow: Query details + Expert reply + Recommendation record (datastore)

Output Data Flow: User query + Expert Reply + Recommendation record (datastore)

Process Logic:

- Collects query details from the user.

- Stores it in the recommendation record datastore as user query.
- Recives a reply from the expert from recommendation record datastore.
- Sends the reply to the user.

Number: 6

Title: Report Preparation

Description: This process collects information about user, financial and staff records from their respective database, allows the admin to make reports and stores in a new database named report records. Similarly, it allows the admin to view the created reports.

Input Data Flow: User Details + User records (datastore) + Financial record (datastore) + Financial Details+ Staff Details + Staff records (datastore) + Record Details + Create report + Report records (datastore)

Output Data Flow: View Report + Records Details + Report records (datastore)

Process Logic:

- Takes user record from user record datastore.
- Takes financial record from financial record datastore.
- Takes staff record from staff record datastore.
- Allows admin to create reports.
- Stores report on report record datastore.
- Allows admin to view reports.

Number: 7

Title: Take certification exams and check results

Description: This process collects user exam details to store in datastore and returns result details. It also allows the user to take mock tests, certification exams and check results.

Input Data Flow: User exam details + Exam details + Exam records (datastore)

Output Data Flow: Result details + Exam records (datastore)

Process Logic:

- Takes user exam details from user.
- Allows user to take mock tests and certification exams.
- Stores exam details in exam records database.
- Access result details from exam records
- Allows user to check results
- Sends result details to the user.

Number 8:

Title: Browse Forum

Description: This process allows the user to make posts, comment, share and upvote the post. It also collects interaction data from user and stores it in activity record database.

Input Data Flow: Interaction data

Output Data Flow: Interaction data + Activity records (datastore)

Process Logic:

- Take interaction data from user.
- Allows the user to make posts, comment, share and upvote the post
- Store interaction data in activity records datastore.

Number 9:

Title: Get Notifications

Description: This process collects activity details from database and sends notification to user.

Input Data Flow: Activity Details + Activity Records (datastore)

Output Data Flow: Notifications

Process Logic:

- Take activity details from the database.
- Send notification to the user.

Number: 10

Title: Give recommendations

Description: This process provides user query to staff from recommendation record datastore and returns expert reply to recommendations datastore from staff.

Input Data Flow: User query + Expert Reply + Recommendation record (datastore)

Output Data Flow: Expert Reply + User query + Recommendation record (datastore)

Process Logic:

- Collects user query from the recommendation datastore.
- Sends a reply from the expert to the recommendation datastore

Number: 11

Title: Edit records

Description: This process allows staff to edit the records and store the edited records in plant records datastore and program record datastore.

Input Flow: Records

Output Flow: Plant details + Program details + Program records (datastore) + Plants records (datastore)

Process Logic:

- Allows staff to edit plant and program records.
- Stores edited records in plant records and program records datastore.

Level 2 processes

Number: 1.1

Title: User Registration

Description: In this process for the registration of the user the user details and the registration details are sent to the user record datastore and the registration confirmation is sent back to the user

Input Data Flow: User Details + Registration Details + User Records (Datastore)

Output Data Flow: Registration Confirmation

Process Logic:

- Store the user and registration details in the datastore
- Send the registration confirmation notification to the user

Number: 1.2

Title: Staff Registration

Description: In this process the details of the staff and the registration details of the staff are sent to staff record datastore and the registration confirmation is sent back to the staff

Input Data Flow: Staff Details + Registration Details + Staff Records (Datastore)

Output Data Flow: Registration Confirmation

Process Logic:

- Store the staff details and registration details in the datastore
- Send the registration confirmation notification to the Staff

Number: 1.3

Title: Confirmation

Description: The motive of this process is to process the registration request that it gets from User Record (Datastore) and Staff Record (Datastore) and send back the registration confirmation to the User and Staff

Input Data Flow: Registration Confirmation + User record (datastore) + Staff record (datastore)

Output Data Flow: Registration Confirmation

Process Logic:

- Process the registration request from staff and user record datastore
- Send the registration confirmation to the user and staff

<u>Number: 8.6</u>

Title: Filter interaction

Description: The process retrieves post data, comment data, share data and upvote data and filters the said data. It then sends the filtered data to the activity records datastore.

Input Data Flow: Post data + Comment data + Share data + Upvote data

Output Data Flow: Filtered data + Activity records (datastore)

Process Logic:

- Receives various kinds of user interaction data
- Filters the data
- Sends the filtered to be stored in activity records datastore

<u>Number: 9.1</u>

Title: Send notification

Description: This process receives various types of notification data and sends it back to the user

Input Data Flow: Post notification data + Comment notification data + Share notification data + Upvote notification data

Output Data Flow: Notification data

Process Logic:

- Receives various types of notification data
- Send the specific notification data to the user

4.3. Design Specification

The Design Specification for the entire system is a meticulous and comprehensive document. It details the structural aspects of the system, encompassing parameters, properties, specifications, requirements of the entire system. This comprehensive document comprises a structural chart providing an essential blueprint for the development of an efficient system.

4.3.1. Structure Chart

The ultimate form of software architecture is a structure chart, which provides a comprehensive, hierarchical schematic of all system components and how they interact. It is more than just a developer's roadmap; it is a dynamic story that accurately and concisely communicates the complexities of software design. The structure chart, which encapsulates the intellectual architecture envisioned by the development team, becomes a live document through its graphical representation of functions and subroutines. (Holmes, 2023)

Beyond technical documentation, this advanced visualisation technique becomes a common language in the commercial setting. Using shapes and connecting lines, the structure chart goes beyond standard documentation by providing a clear and eye-catching illustration of intricate relationships, processes, and organisational hierarchies. Each stroke represents a thoughtful design choice, directing the development team towards a place where intricacy meets elegance and precision meets creativity. It's a piece of software design art. (Holmes, 2023)

Symbols used in structure chart are:

Module

Module refers to a rectangular shape container which represents components or functions of a software application. It is created to break down the complexity of a software into understandable parts. Each module is labeled with its name and contains a brief description of its functionality. Module is further divided into three types:

- **Control Module:** It is responsible for coordinating the activities of other modules, helps to control the overall program flow.
- **Sub Module:** It is the child module of another module. It contributes to the overall functionality and organization of a larger module or system.
- **Library Module:** It is a reusable and invokable module, saves developer time and effort by providing a set of functions.

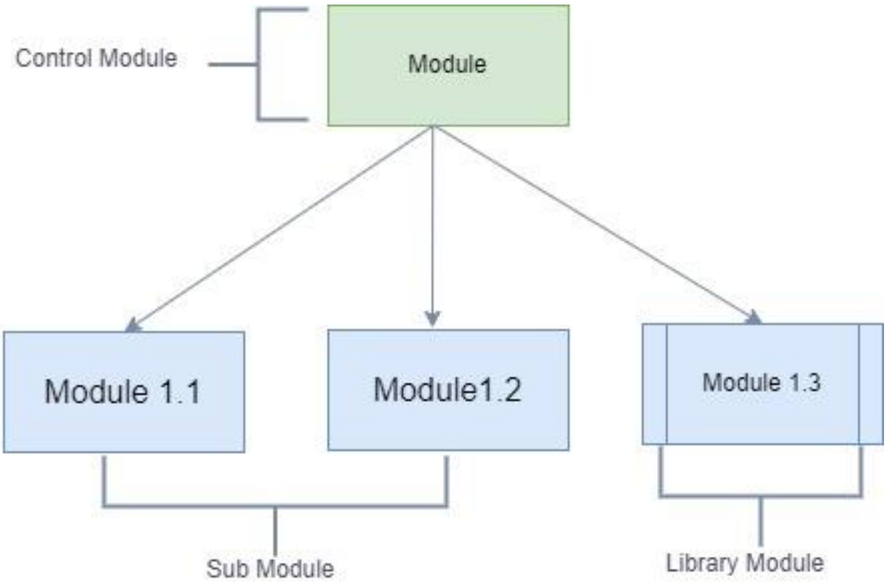


Figure 7: Types of Modules.

Parameter

It represents the movement of data between the modules that is represented through directed arrow with empty circle at the end.

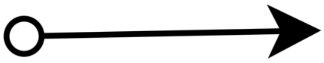


Figure 8: Parameter

Control Parameter

It controls the behavior of the module based on specific conditions, indicates that certain criteria has been met.

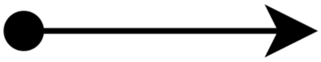


Figure 9: Control Parameter

Repetition

It describes the repetitive execution of module upon the sub module highlighting that a process can occur multiple times.

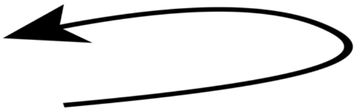


Figure 10: Repetition

Decision

It represents a point in the system where the flow of control can take different paths depending on the outcome of a logical condition. The decision structure is often visualized as a diamond shape in a structure chart.

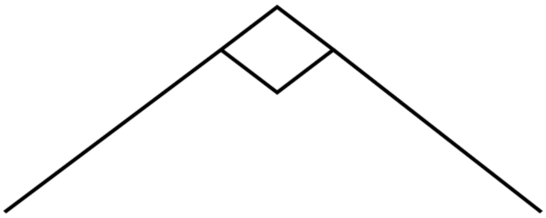


Figure 11: Decision

Call line

It is the representation of the flow of control from one module to another. It indicates that one module calls another module to perform certain task.

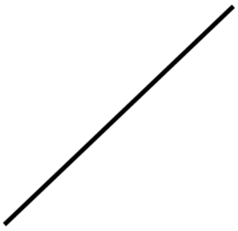


Figure 12: Call Line

The structure chart of the system is given below:

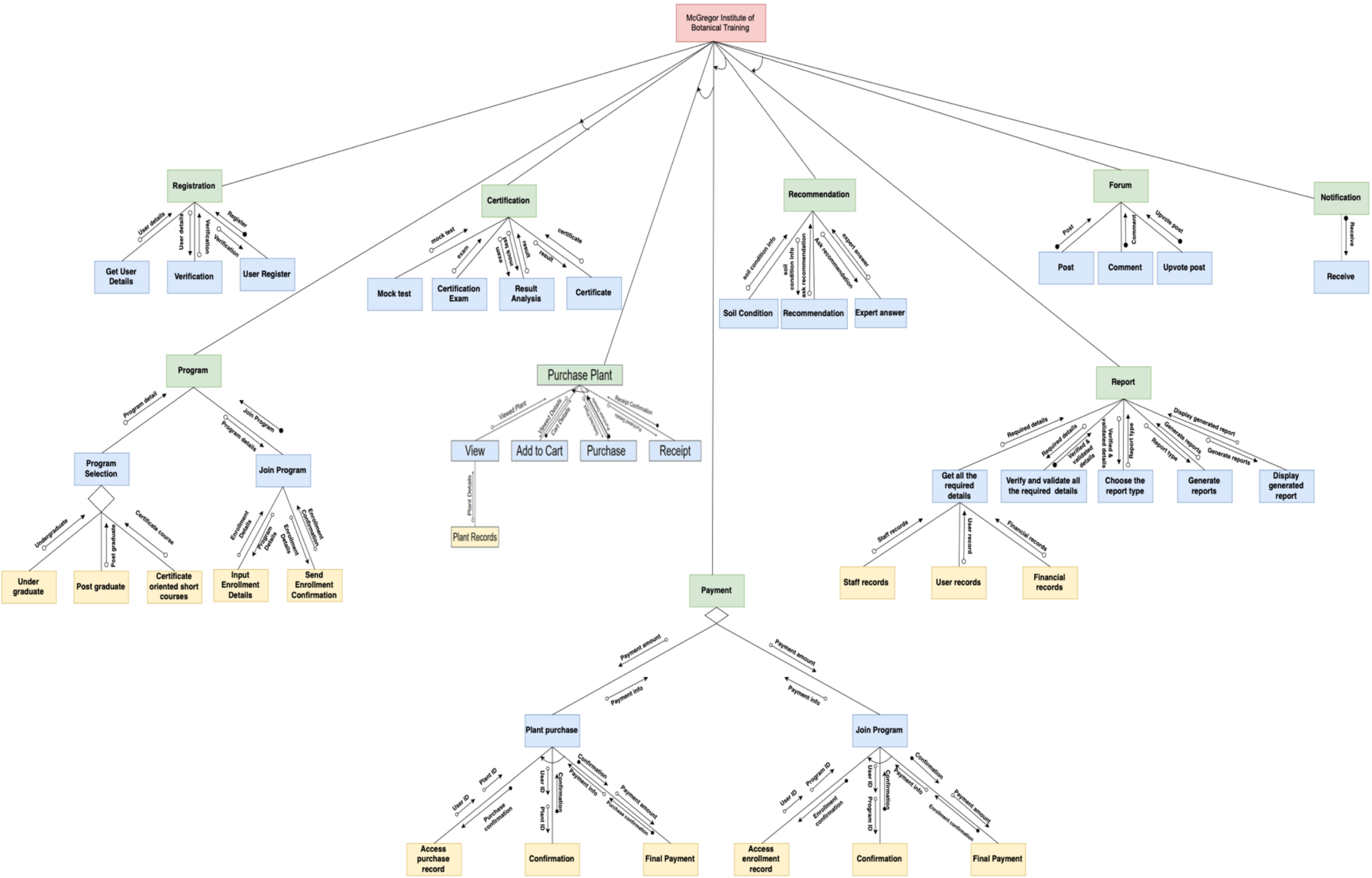


Figure 13: Structure Chart

4.4. Assignment Diary

The assignment Diary includes the following parts which are described below:

4.4.1. Assumptions

The assumptions made in the analysis of this system are as follows:

1. The users should be registered in the system in order to use the services.
2. The staff should also be registered in the system in order to use the services.
3. The staff includes admin, employees, and experts.
4. Experts are hired by the institute.
5. Users may join an undergraduate or postgraduate course or certificate-oriented courses.
6. Courses are both paid and unpaid.
7. Users can just view plants without purchasing.
8. Plants can be kept in cart and bought in bulk.
9. Users can pay for the plant purchased and courses they are enrolled in.
10. Users can ask the experts for recommendations.
11. Experts can reply with suitable answers.
12. Admins are able to prepare a detailed financial report, employees report and user report.
13. User can take mock tests, check results, and also take certification exams.
14. Users can use the forum to chat, share posts, comment on posts and upvote the posts.
15. Users can get notifications according to their activity.
16. Notification activities includes sharing, commenting on, upvoting and making a post.
17. The admins are able to edit the program and plant records.
18. The users and staff get confirmation after they are successfully registered.

19. The users get confirmation after they successfully enrol in a program.
20. The users get a purchase receipt after purchasing a plant.
21. The users get payment info after completing the payment.
22. The admins are able to view the reports created.

4.4.2. Omissions/Inconsistencies

The omissions and inconsistencies found during the analysis of this system are as follows:

1. It is not mentioned if the registration for the user is free or not.
2. The specification does not clarify if the staff need to be registered to use the system.
3. It is not clear if the registration should be renewed.
4. It is not specified if the registration can be cancelled.
5. It is not mentioned if the employees of the institute can join programs and buy plants like the users.
6. It is not clarified if users can join multiple programs in the same level or multiple programs in different levels. For example: a user may or may not be able to join a graduate course, an undergraduate course and a certificate-oriented course at the same time.
7. The specification does not mention if the payment can be refunded if the user is unsatisfied with the product or wants to drop the classes that they might have enrolled in.
8. There is no mention of the users receiving a receipt after purchasing products.
9. It is not clear how the payment, receipt and such are handled if the courses and purchased items are free.
10. It is not clarified whether the data stores can be accessed and edited by the admin.
11. It is not mentioned if the experts can be users themselves or are staff members hired by the institute.

12. The requirements do not mention if the admin can view reports.
13. The prerequisites for giving certification exams are not made clear in the guidelines.
14. It is not mentioned if the users need to pay in order to take certification exams.
15. There is no mention of moderators for the forum.
16. It is not clear what sort of notifications users get.

4.4.3. Group Member Responsibilities

Group Members	Group Task	Individual Task
Aaditya Raj Shrestha	• Assignment Diary • Data Flow Diagram • Documentation	Make Payment
Binita Bhandari	• Introduction • SRS • Documentation	Report Preparation
Bishal Bogati	• Structure Chart • Group Meeting • Documentation	Join Program
Dikshya Sharma	• Entity Relationship Diagram • Data Dictionary • Documentation	Purchase Plant
Siddhant Bhurtel	• Process Specification • Project Charter • Summary • Documentation	Take Certification Exam

Table 2: Group Member Responsibilities.

4.4.4. Group Meeting

Meeting Entry

Meeting No. :1

Date: December 17,2023

Start Time : 10:00 AM

Finish Time: 11:30 AM

Location: London Block

Discussion

- Overall Coursework concept.
- Task division to every group member.
- Discussed Project Charter and SRS
- Queries about coursework.

Achievements

- Understood the overall concept of coursework.
- Successfully solved queries about the coursework.

Task Until Next Meeting

- Complete Project Charter and SRS.
- Review finished topic with tutor.
- Collect queries arising while completing the task.

Team Members	Signature
Aaditya Raj Shrestha	Aaditya
Binita Bhandari	Binita
Bishal Bogati	Bishal
Dikshya Sharma	Dikshya
Siddhant Bhurtel	Siddhant

Table 3: Group Meeting 1.

Meeting Entry

Meeting No. :2

Date: December 22,2023

Start Time: 12:00 PM

Finish Time: 1:00 PM

Location: Skill Block

Discussion

- Progress update on Project Charter and SRS.
- Clarify any queries encountered.
- Concept of DFD and ERD.

Achievements

- Successfully completed the Project Charter and SRS.
- Reviewed the Project Charter and SRS.

Task Until Next Meeting

- Prepare questions or topics for tutor review.
- Complete DFD and ERD.
- Share progress and address any challenges faced.

Team Members	Signature
Aaditya Raj Shrestha	Aaditya
Binita Bhandari	Binita
Bishal Bogati	Bishal
Dikshya Sharma	Dikshya
Siddhant Bhurtel	Siddhant

Table 4: Group Meeting 2.

Meeting Entry

Meeting No. :3

Date: December 24,2023

Start Time: 11:00 AM

Finish Time: 1:00 PM

Location: Alumni Block

Discussion

- ❖ Progress update on DFD and ERD.
- ❖ Clarify any queries encountered.
- ❖ Discuss about data dictionary and process specification.

Achievements

- ❖ Completed DFD and ERD.
- ❖ Reviewed the DFD and ERD.
- ❖ All queries encountered.

Task Until Next Meeting

- ❖ Complete data dictionary and process specification.
- ❖ Prepare questions or topics for tutor review.
- ❖ Share progress and address any challenges faced.

Team Members	**Signature**
Aaditya Raj Shrestha	Aaditya
Binita Bhandari	Binita
Bishal Bogati	Bishal
Dikshya Sharma	Dikshya
Siddhant Bhurtel	Siddhant

Table 5: Group Meeting 3.

Meeting Entry

Meeting No. :4

Date: December 26,2023

Start Time: 11:00 AM

Finish Time: 12:30 PM

Location: Nepal Block

Discussion

- Progress update on Data Structure and Process Specification.
- Clarify any queries encountered.
- Discuss about Structure Chart and Individual task.
- Overall documentation.

Achievements

- Completed Data Structure and Process Specification.
- Reviewed the Data Structure and Process Specification.
- All queries encountered.

Task Until Next Meeting

- Complete Structure Chart and Individual task.
- Completion of the documentation.
- Prepare questions or topics for tutor review.
- Share progress and address any challenges faced.

Team Members	**Signature**
Aaditya Raj Shrestha	Aaditya
Binita Bhandari	Binita
Bishal Bogati	Bishal
Dikshya Sharma	Dikshya
Siddhant Bhurtel	Siddhant

Table 6: Group Meeting 4.

Meeting Entry

Meeting No. :5

Date: December 26,2023

Start Time: 10:00 AM

Finish Time: 12:00 PM

Location: London Block

Discussion

- ❖ Progress update on the Structure Chart, Individual Task and overall documentation.
- ❖ Clarify any queries encountered.

Achievements

- ❖ Completed overall documentation.
- ❖ Reviewed documentation.
- ❖ All queries encountered.

Task Until Next Meeting

- ❖ No need for next meeting as the tasks are over.

Team Members	**Signature**
Aaditya Raj Shrestha	Aaditya
Binita Bhandari	Binita
Bishal Bogati	Bishal
Dikshya Sharma	Dikshya
Siddhant Bhurtel	Siddhant

Table 7: Group Meeting 5.

5. Individual Task

5.1. Make Payment

Student Name: Aaditya Raj Shrestha

London Met ID: 22068760

5.1.1. Environmental Model Specification

The environmental model specification of the make payment function shows the interactions between various environmental factors along with the process of making the payment.

Context Level Diagram (Level 0)

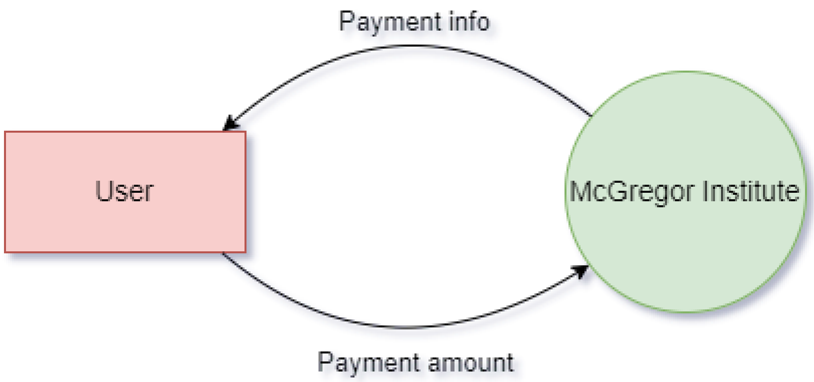


Figure 14: Context Level Diagram (Level 0) of Make Payment.

In the context level diagram of make payment function of this system, the key elements used are user as external entity, make payment as a process and payment info and payment amount as the data flow between them.

Here, the user initiates the payment process by sending the request to the system. The system then processes the command and carries out the necessary steps for the payment and then sends the payment info to the user after which the user can send payment amount to the system.

5.1.2. Internal Model Specification

The Internal Model Specification for the payment process shows the detailed documentation outlining how the internal operations, components, external entities, and process specification of the system collaborate to execute payment function.

Here, the context level diagram is further divided into 2 different levels of DFD. The level 1 DFD and level 2 DFD of payment process are shown below.

Level 1 DFD:

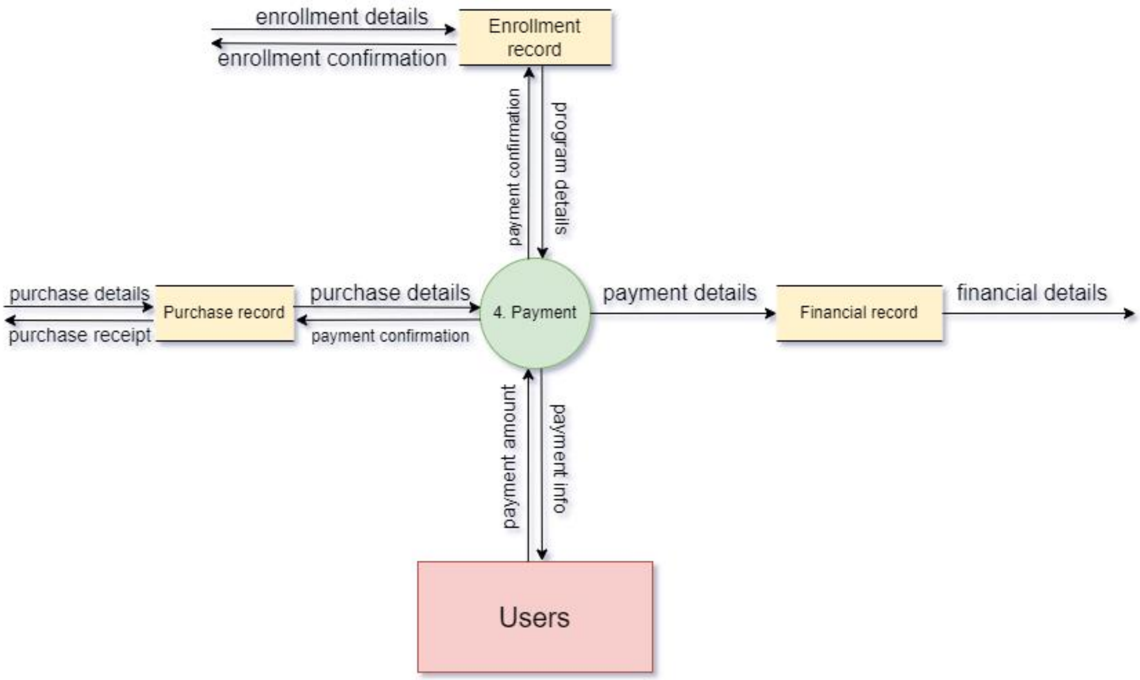


Figure 15: Level 1 DFD of Make Payment.

In the level 1 DFD of payment process, the key elements used are user as an entity, payment as a process, enrolment record, purchase record, financial record as data stores and enrolment details, enrolment confirmation, payment confirmation, program details, purchase details, purchase receipt, purchase details, payment confirmation, payment amount, payment info, payment details, financial details as the data flow in this module of the system.

Here, the user initiates the payment process by sending the request to the system. The payment process then accesses data from various data stores i.e. enrolment record and purchase record. The function sends the payment information to the user and the user sends back the payment amount to the function after which the payment is done. The payment details after completion of the payment process are stored in the financial record.

Level 2 DFD:

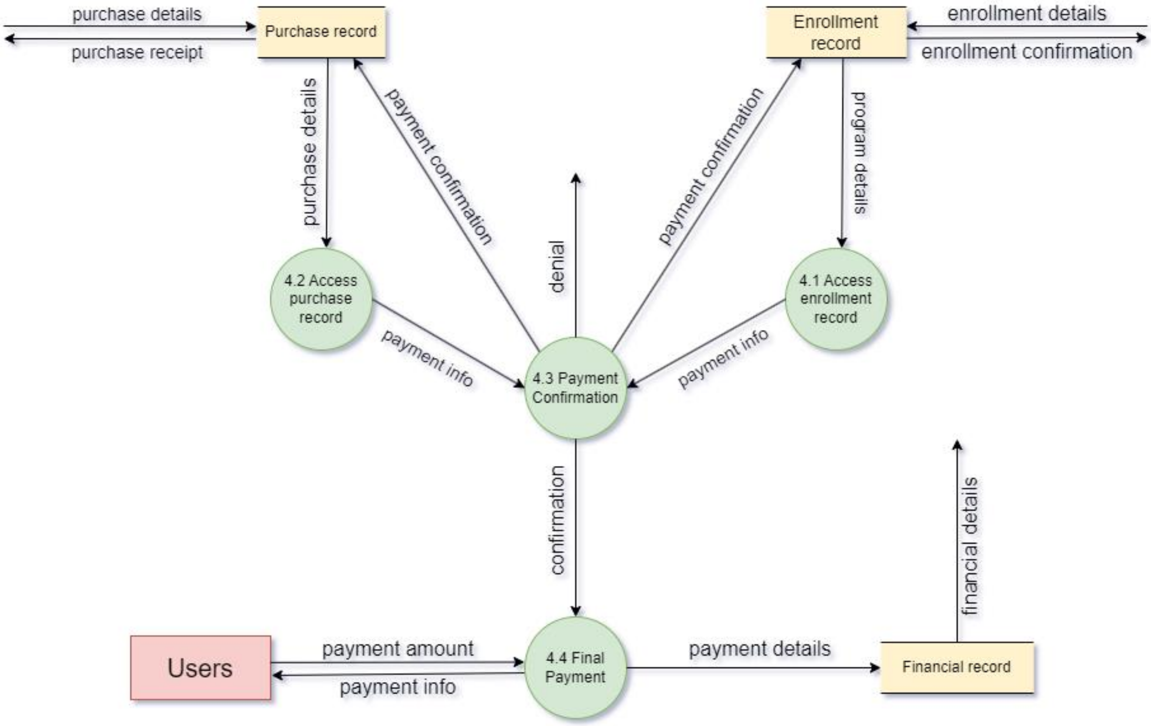


Figure 16: Level 2 DFD of Make Payment.

In the Level 2 DFD of payment function of this system, the key elements used are user as an external entity, access purchase record, access enrolment record, payment confirmation, and final payment as the processes, purchase record, enrolment record, and financial record as the data stores. Similarly, purchase details, purchase receipt, purchase details, payment confirmation, payment info, denial, enrolment details, enrolment confirmation, confirmation, payment amount, payment details, and financial details are the data flow within the level 2 DFD of this system.

In this process, the access purchase record gets the data of the items that the user has purchased. After the details are accessed, the payment information is sent to the payment confirmation function where the user makes the confirmation for the payment. After the confirmation, payment confirmation is sent to the purchase record data store and the final payment function comes into action where payment information is sent to the user and the user sends payment amount to the system to finish the transaction. Similarly, the access enrolment record function gets the data of the program that the user is trying to enrol into. After the details are accessed, the payment information is sent to the payment confirmation function where the user makes the confirmation for the payment. After the confirmation, payment confirmation is sent to the enrolment record data store and the final payment function sends the payment information to the user and the user sends the payment amount to the system to finish the transaction. After the transactions are completed, the payment details are sent to the financial record data store.

5.1.3. Design Specification

The Design Specification of the **“Make Payment”** contains two parts Structure Chart and Module specification. The structure chart displays the hierarchical structure of a system while the module specification contains various general elements of a system like module name, pseudocode, local functions, global functions, etc. for the **“Make Payment”** function.

Structure Chart for Payment

A structure chart is a hierarchical structural representation of modules. It helps to break down the system into functional modules and helps to describe the functions and sub-functions of each module in minute details. The structure chart for the module **“Make Payment”** is as shown below.

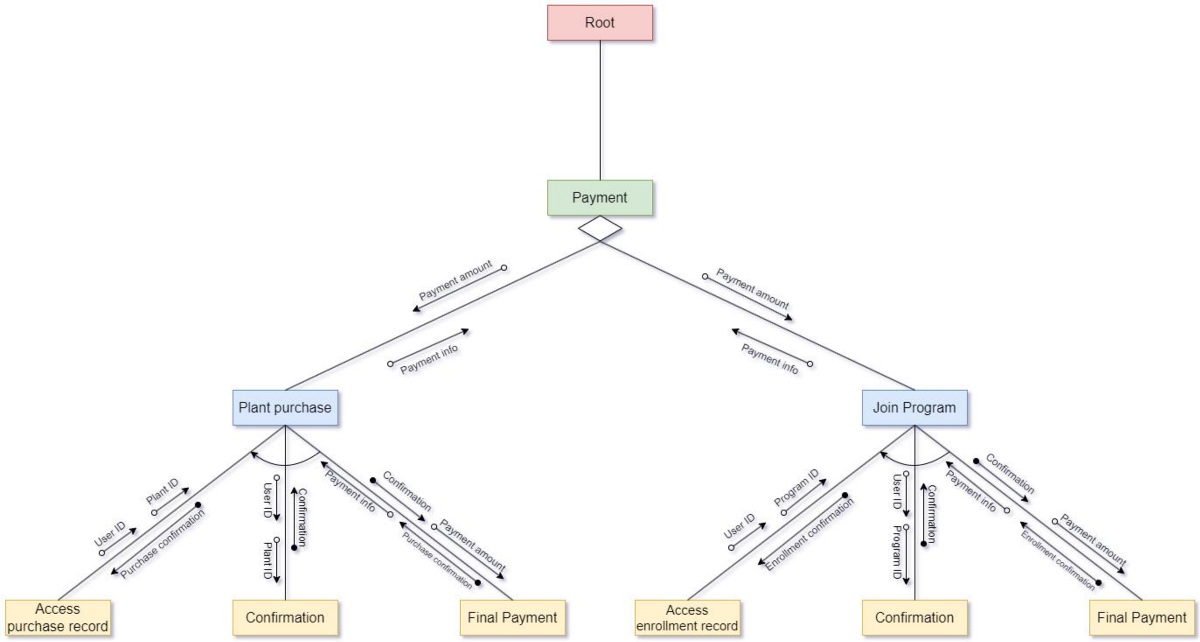


Figure 17: Structure Chart of Make Payment.

The chart shown above gives a visual representation of the hierarchical structure for the function “Make Payment”. In the top-level module, we have Root which represents the main overall system. The second level of the structure chart contains the “Payment” module which is responsible for the overall processing of this particular function. It is further divided into the third level which contains the plant purchase and join program module and the user selects one of the payment options at a time. Finally, in the fourth level, both the plant purchase module and join program module are further separated into access purchase record, confirmation and final payment for plant purchase module and access enrolment record, confirmation, and final payment for join program module.

For the plant purchase module, user id and plant id parameters are accessed by the access purchase record module and sent to the confirmation module. After user confirmation, a confirmation control parameter sent to the final payment which sends the payment information parameter to the purchase plant module. The payment information parameter is sent to the main payment module where the user pays the payment amount. A payment amount parameter is sent to the plant purchase module and forwarded to the final payment module. Finally, a purchase confirmation control parameter is sent from this module to the access purchase record module where the record is modified.

For the join program module, user id and program id parameters are accessed by the access enrolment record module and sent to the confirmation module. After user confirmation, a confirmation control parameter sent to the final payment which sends the payment information parameter to the join program module. The payment information parameter is sent to the main payment module where the user pays the payment amount. A payment amount parameter is sent to the join plant module and forwarded to the final payment module. Finally, an enrolment confirmation control parameter is sent from this module to the access enrolment record module where the record is modified.

Module Specification for Payment

The module specification for the function **“Make Payment”** is a detailed description of the document which includes all the general information of this particular function. It contains the module name, purpose, pseudocode of this module, input parameters, output parameters, global and local variables used in this function, and which module it calls and is called by. The module specification for this particular module is as shown below:

Module Name	Make Payment
Purpose	The purpose of this module is to make a payment system for the plants purchased by the users or the paid courses that they are enrolled in.

Pseudo code	**IMPORT** necessary components
	CREATE class **EXTEND** necessary class
	DO
	
	DECLARE variables
	    CREATE function payment
	    DO
	        SELECT payment type
	        CALL plant or program function
	        ACCESS data store
	
	        INPUT necessary parameters
	        CONFIRM payment
	            IF payment **IS** successful
	                UPDATE records
	                DISPLAY successful message
	            ELSE
	                DISPLAY error message
	            END IF
	    END DO
	END DO

Input Parameters	Payment type, Payment amount, Payment confirmation, User ID
Output Parameters	Payment information
Global Variables	User ID
Local Variables	Payment_type, Payment_amount, Payment_ confirmation, Payment_Date, UserPlant_ID, User Program_ID, Final_payment
Calls	Plant Purchase, Join Program
Called by	Root

Table 8: Module Specification of Make Payment.

5.2. Purchase Plant

Name: Dikshya Sharma

London Met ID: 22067520

5.2.1. Introduction

The **"Purchase Plant"** functionality of **"McGregor Institute of Botanical Training"** enables purchase a plant.

In this particular task, my assigned responsibility is to manage purchase of plant. The first step involves choosing from a wide range of plant species, some of which can be obtained for a minimal cost and others of which are provided without cost. After user makes the choice, they proceed for the purchase and gets information about the plants they have selected which includes important details regarding the name of plant and all other details.

5.2.2. Environmental Model Specification

The environmental model specification of this Purchase Plant shows the interactions between various environmental factors along with the process of purchasing plant in the system. It depicts the **Data Flow Diagram at Context Level i.e. Level 0 DFD.**

Context Level Diagram (Level 0)

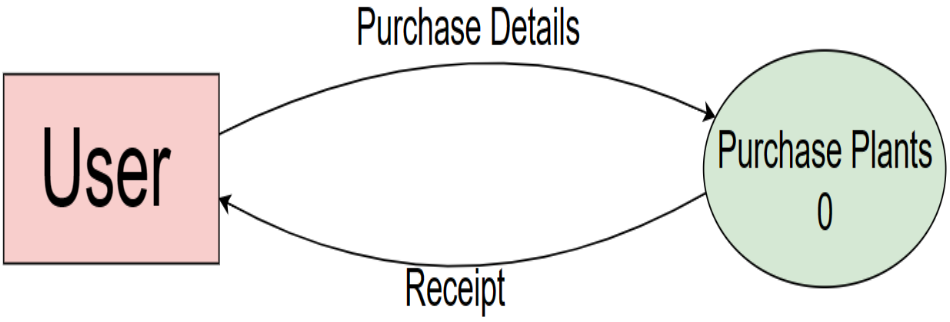


Figure 18: Context Level Diagram (Level 0) of Purchase Plant.

In this context level diagram of purchase plant, the key elements used are User as an external entity, Purchase plants as a process, and Purchase Details and receipt as a data flow between them.

The figure above illustrates the context level diagram for purchasing the plant. In the diagram, “User” signifies the Plant Enthusiasts initiating a request to purchase a plant. The user, as a requester, provides details for purchase including information such plant name, plant type, price, and other relevant details. The user receives a purchase receipt from the system when their request has been processed receipt. In order to give the user a complete record, this receipt includes all relevant information related to the plant purchase transaction.

5.2.3. Internal Model Specification:

The Internal Model Specification for this report preparation shows the detailed documentation outlining how the internal operations, components, external entities, and process specification of the system collaborate to execute purchase plant function. It provides the clarity about purchasing plant in the system.

It further divides the context level diagram into 2 different level of diagrams i.e. **Level 1 DFD** and **Level 2 DFD**. The Level 1 and Level 2 DFD (Data Flow Diagram) of Purchasing Plant are shown below:

Level 1 DFD

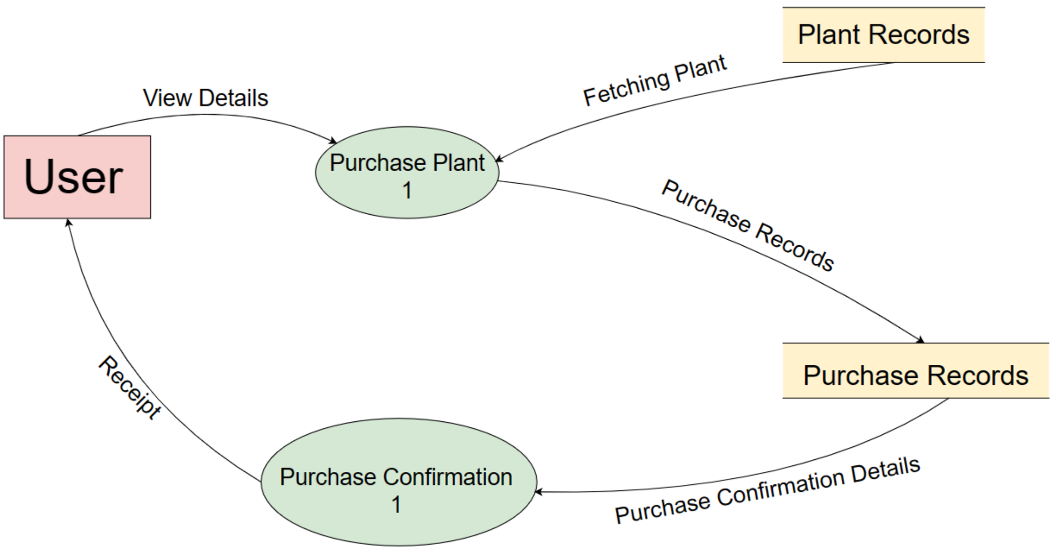


Figure 19: Level 1 Diagram of Purchase Plant.

In this Level 1 DFD of purchase plant, the key elements used are user as an external entity, purchase plant and purchase confirmation as a process, Plant Records and Purchase Records as data store and view details, fetching plant, purchase record and receipt as a data flow within this module of the system.

The figure above illustrates the context level diagram for purchasing the plant. The process of purchasing plant begins with the “User” that is Plant Enthusiasts, who initiates to make a purchase by providing details for purchase which includes plant name, type, price, and other relevant information. The information is then stored in a purchase record database where the user and purchase details are stored. The information is then utilized to make a purchase confirmation and to generate a receipt with every relevant information about the plant purchase transaction, which is given to the user.

Level 2 DFD

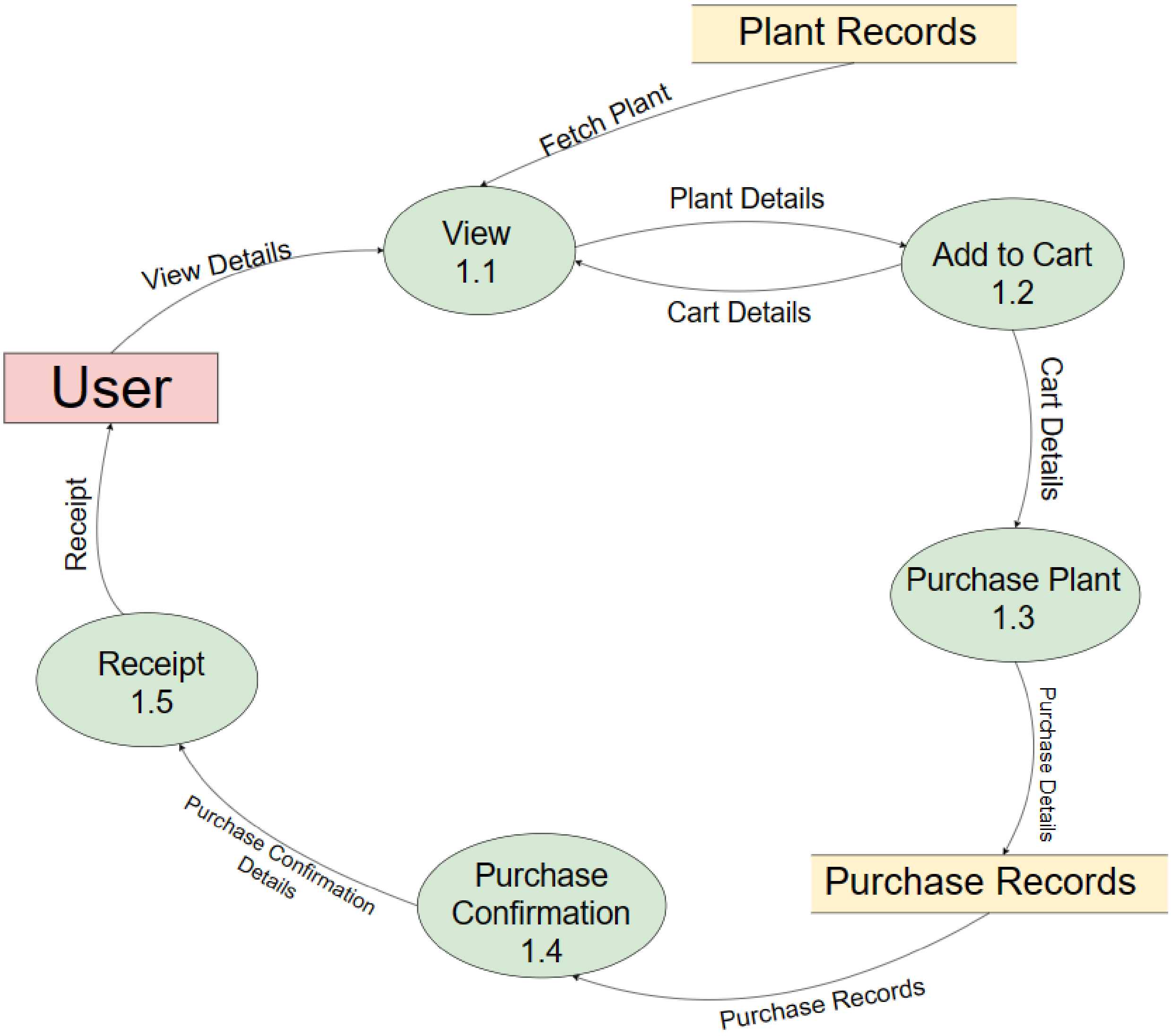


Figure 20: Level 2 Diagram of Purchase Plant.

In this Level 2 DFD of purchase plant, the key elements used are User as an external entity, view, add to cart, purchase plant, purchase confirmation and receipt as a process, Purchase Records and Plant Records as a data store and View Details, Fetch plants, Plant Details, Cart Details, Purchase Details, Purchase Records and Receipt as a data flow within this module of the system.

The figure above illustrates the Lavel 2 level diagram for purchasing the plant. In the diagram User views the plants and adds to cart with required quantity and the added cart items proceed for the purchase. The purchase information is then stored in a purchase record database which is further utilized to confirm a purchase and to generate a receipt with every relevant information required. The generated receipt is then provided to the user for further process.

5.2.4 Design Specification

The Design Specification of purchasing plant involves the detailed structural document listing all the parameters, properties, specifications, requirements, and pseudocode of the Purchase Plant module. It consists of the **structural chart** and **module specifications** serving as a foundation for the development of an effective report in the system.

Structure Chart for Purchase Plant

The structure chart of the purchase plant module represents the hierarchical arrangement of various components, functions along with the different parts of the system. Here, every level of the hierarchy denotes different levels of data abstraction or data details within the system.

The components of the structure chart are read from left-to-right and top-to-bottom. Different symbols are used to denote different process in the hierarchy.

Here, at topmost model there resides the main module i.e. the system and the system branches to different module and those modules to sub-modules and so on.

The structure chart for the module **“Purchase Plant”** is shown below:

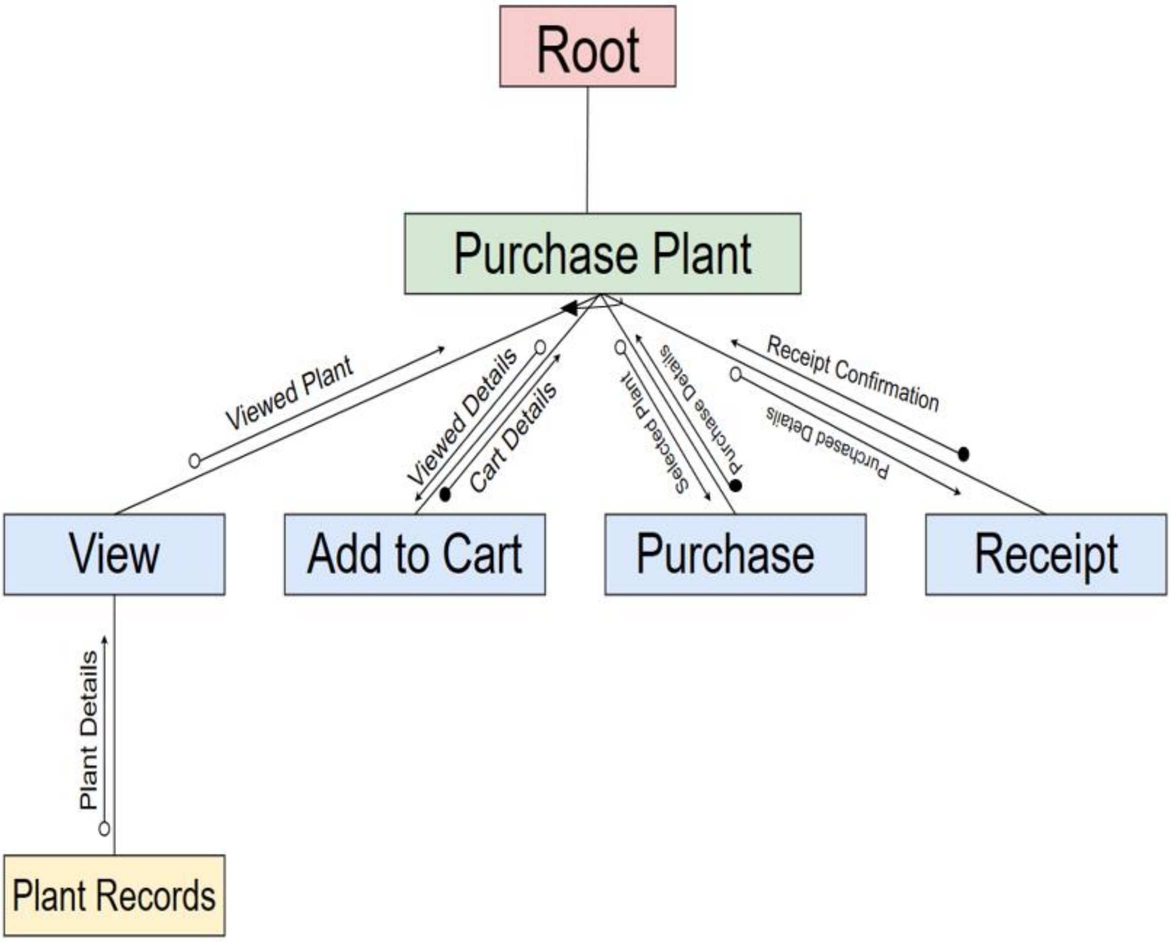


Figure 21: Structure Chart of Purchase Plant.

The above structure chart graphically represents the hierarchical project's structure of the module- "Purchase Plant". The structure chart for this module is simple which makes the project easier to understand.

Here, at the top-most level there is main system denoted by "Root". After it, there is the name of the main module "Purchase Plant" overseeing the entire process and functionality of the system. Similarly, it consists of several sub modules like View, add to cart, purchase, and receipt. These modules also interact with data store Plant Records to get the required plant information.

A project’s structure of purchasing a plant is depicted graphically in the above structure chart. The structure chart for this process is simple and straightforward. After the plant details are viewed user input’s view details and subsequently adds selected plant to the cart. The added cart item status is then communicated to the system. This cart status provides by the user, serves as a crucial component for the plant purchase process. The system then retrieves the cart status and facilitates the selection of desired plants. The purchase details are then sent to the system with all relevant information. After, the user submits the purchase details to system, it responds with a purchase confirmation, and then the receipt is generated to conclude the transaction.

Module Specification for Purchase Plant:

The module specification for the module “**Purchase Plant**” is a detailed description of the document outlining all the specifications, functionalities, and requirements of the system that generates the report for the system.

It consists of name and purpose of the module, pseudocode, input parameters, output parameters, Global Variables, Local Variables, Calls (references to another module) and Called By (name of other module using this module).

The module specification for the **“Purchase Plant”** module is shown below:

Name	Purchase a Plant
Purpose	The module's function is to let the user purchase a plant after they provide viewed plant details, cart details and the selected plant details.
Pseudo Code	**BEGIN** Plant Purchase **DO** **FETCH** Plant Details from Plant Datastore **VIEW** Plant Details **ADD** Plant to Cart **PROVIDE** Cart Details **PURCHASE** Plant from Added Cart **STORE** Purchase Record in Purchase Datastore **GENERATE** Receipt **DISPLAY** Receipt **END DO**
Input Parameters	Plant Details
Output Parameters	Receipt
Global Variables	User Id
Local Variables	Plant ID, Plant Name, Plant Type, Plant Price, Availability Status Cart Details
Calls	Fetch Plant Details, Purchase Details
Called by	Root

Table 9: Module Specification of Purchase Plant.

5.3. Report Preparation:

Student Name: Binita Bhandari

London Met Id: 22067525

5.3.1 Introduction:

The **"Report Preparation"** functionality of **"McGregor Institute of Botanical Training"** enables staffs to generate detailed reports related to users, staffs, financial transactions, and other relevant aspects of the system.

Similarly, the generated reports serve as the important resources for evaluating user's, staff's, and institute's performance. It also helps in tracking the progress of the system. Additionally, it also aids in strategic planning for upcoming projects along with their improvement.

5.3.2 Environmental Model Specification

The environmental model specification of this report preparation shows the interactions between various environmental factors along with the process of generating reports within the system. It depicts the **Data Flow Diagram at Context Level i.e. Level 0 DFD.**

Context Level Diagram(level 0):

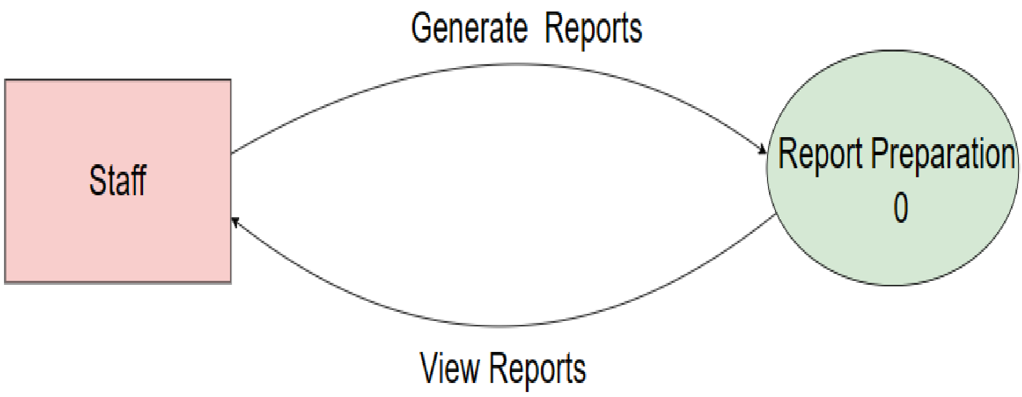


Figure 22: Context Level Diagram (Level 0) of Report Preparation.

In this context level diagram of report preparation of "McGregor Institute of Botanical training", the key elements used are staff as an external entity, report preparation as a process, and generate reports and view reports as a data flow between them.

Here, staff initiates the report preparation by sending the request to the system. Then, a system processes the command and carried out the necessary steps for report preparation, and then delivers report to the staff which allow staffs to access, analyse and review the report along with facilitating informed decision-making and strategic planning for the institute.

5.3.3 Internal Model Specification:

The Internal Model Specification for this report preparation shows the detailed documentation outlining how the internal operations, components, external entities, and process specification of the system collaborate to execute report preparation function. It provides the clarity about the report preparation in the system.

It further divides the context level diagram into 2 different level of diagrams i.e. **Level 1 DFD** and **Level 2 DFD**. The Level 1 and Level 2 DFD (Data Flow Diagram) of Report Preparation are shown below:

Level 1 DFD:

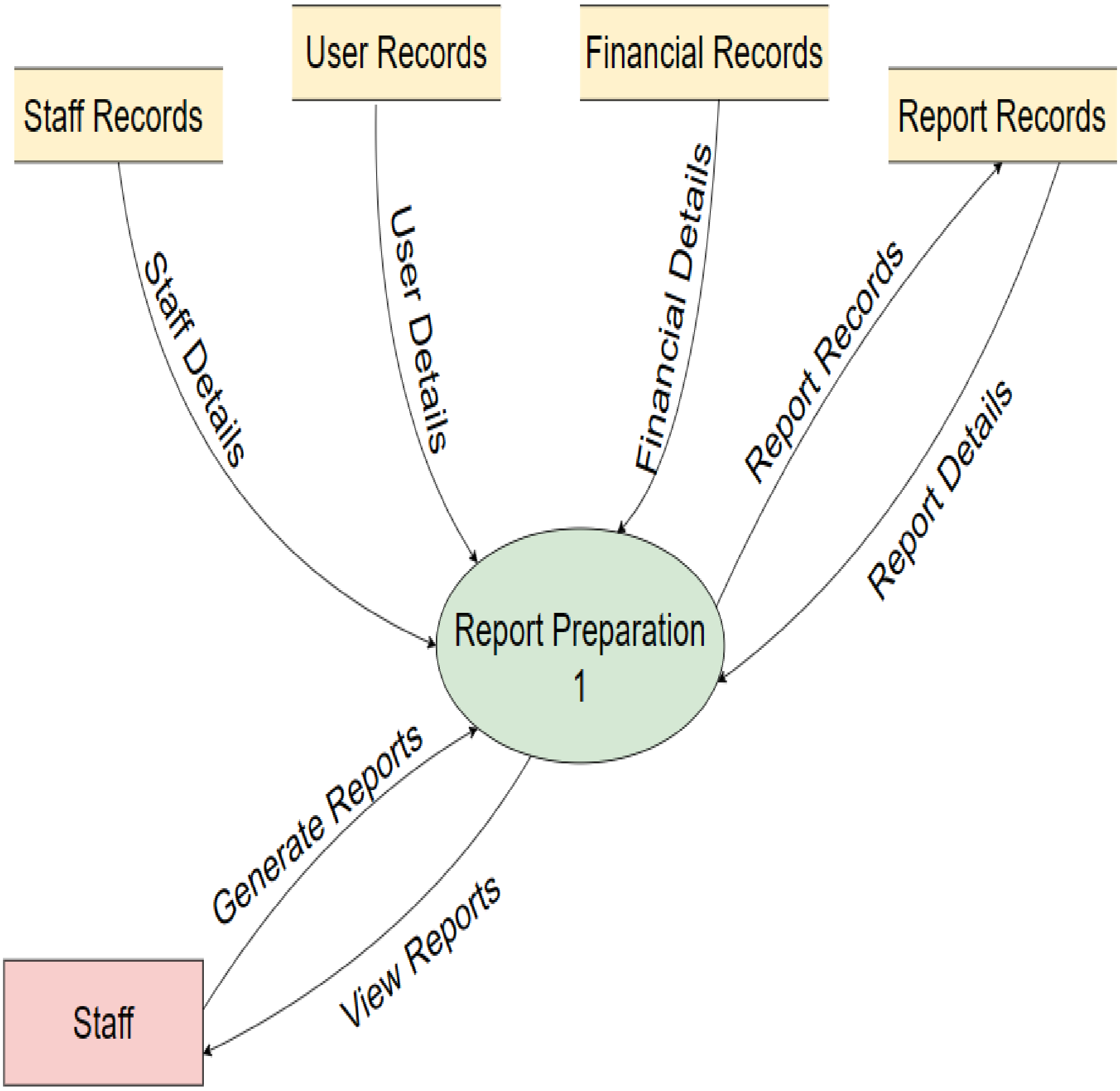


Figure 23: Level 1 DFD of Report Preparation.

In this Level 1 DFD of report preparation of “McGregor Institute of Botanical training”, the key elements used are staff as an external entity, report preparation as a process, Staff Records, User Records, Financial Records, Report Records as data store and Staff Details, User Details, Financial Details, Report Records, Report Details, Generate Reports and View reports as a data flow within this module of the system.

Here, staff initiates the report preparation by sending the request to the system. Then a Report Preparation process obtained the stored data from various data store like user, staff, financial records, and prepared a report accordingly. Then, the resulting records of generated reports are stored in a report record data store and at the same time, the report details of generated reports stored in report record datastore is accessed by the staff through report preparation process and finally the report is exported to the staff to view which facilitates the staff to access, review, and analyse the report aiding in the institute’s strategic planning and well-informed decision-making.

Level 2 DFD:

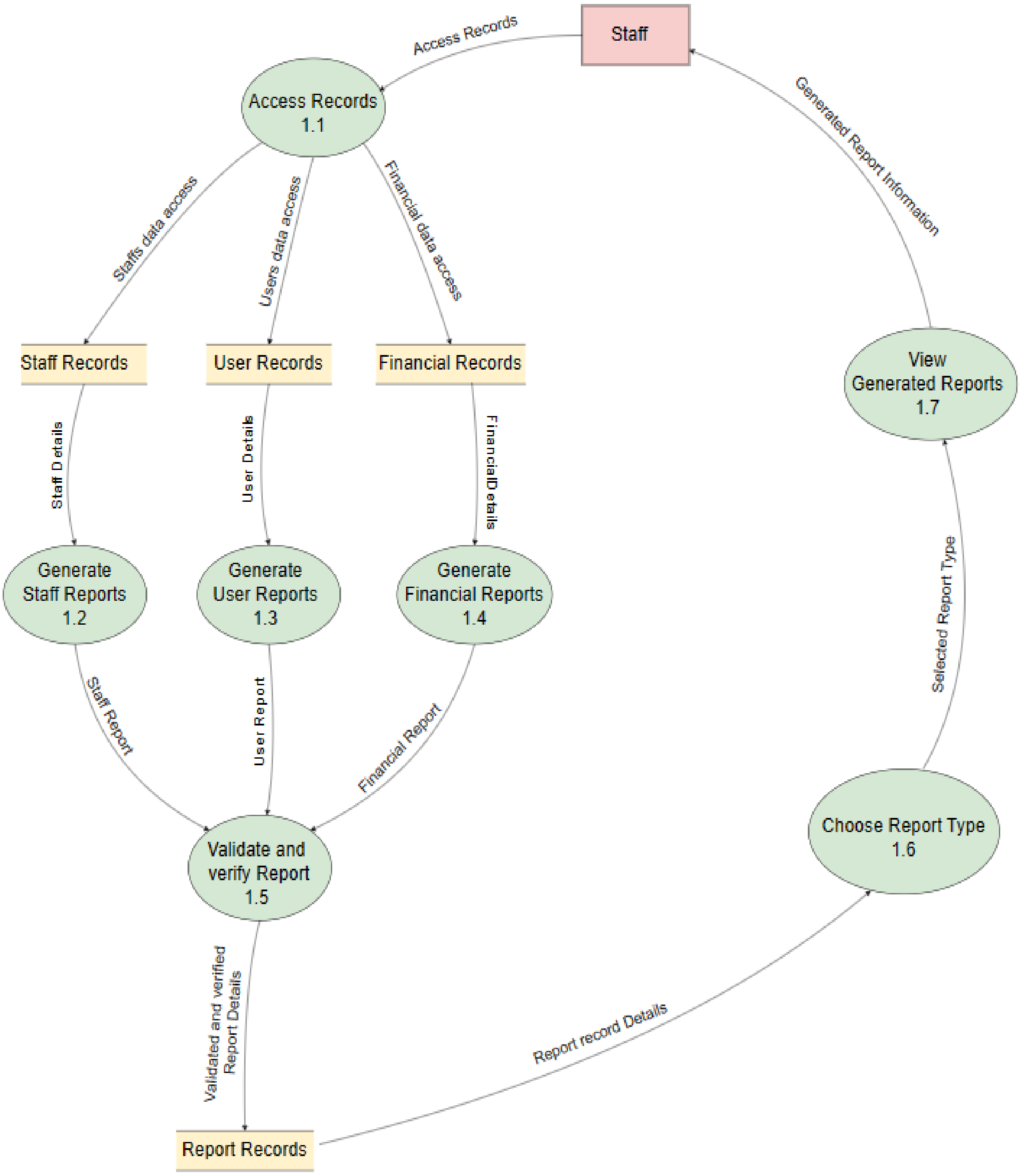


Figure 24: Level 2 Diagram of Report Preparation.

In this Level 2 DFD of report preparation of “McGregor Institute of Botanical training”, the key elements used are staff as an external entity, access records, generate staff reports, generate User Reports, Generate Financial Reports, Validate and Verify Report, Choose Report Type, and View Generated Reports as a process, Staff Records, User Records, Financial Records, Report Records as data store and Access Records, Staff data access, User data access, Financial data access, Staff Details, User Details, Financial Details, Staff Report, User Report, Financial Report, Validated and Verified Report Details, Report record Details, Selected Report Type, Generated Report Information as a data flow within this module of the system.

Here, the user initiate the process by accessing the records from Staff Records, User Records, and Financial Records datastore which then generate each report for staff, user and financial records. Then, Those generated reports are validated and verified over a Validate and verify Report process. After Validating and Verifying the generated report, it is stored over a Report Records datastore. Likewise, the stored data are accessed and a report type is choose in order to generate report. After selecting a report type, the report type selected is sent to the view generated reports process and is finally exported to the staff where staff can access, analyse, and review the report contributing for strategic planning and informed-decision making of the institute.

5.3.4 Design Specification:

The Design Specification of this report preparation involved the detailed structural document listing all the parameters, properties, specifications, requirements, and pseudocode of the Report Preparation module for a development of the proper report for the user. It consists of the **structural chart** and **module specifications** serving as a foundation for the development of an effective report in the system.

Structure Chart of Report Preparation:

The structure chart of the report preparation module represents the hierarchical arrangement of various components, functions along with the different parts of the system. Here, every level of the hierarchy denotes different levels of data abstraction or data details within the system.

The components of the structure chart are read from left-to-right and top-to-bottom. Different symbols are used to denote different process in the hierarchy.

Here, at topmost model there resides the main module i.e. the system and the system branches to different module and those modules to sub-modules and so on.

The structure chart for the module **"Report Preparation"** is shown below:

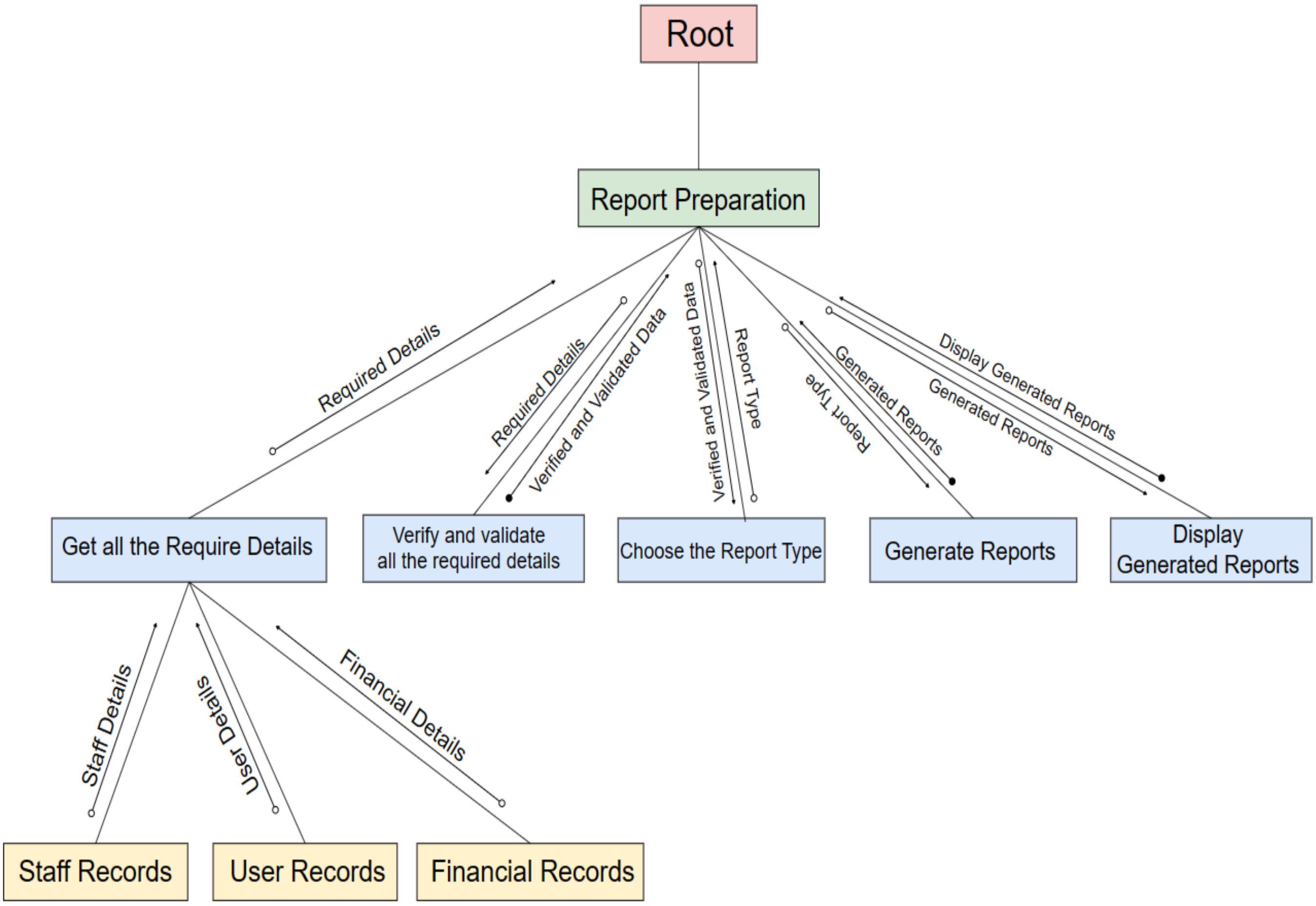


Figure 25: Structure Chart of Report Preparation.

The above structure chart graphically and visually represents the hierarchical project's structure of the module- "Report Preparation". The structure chart for this module is simple which makes the project easier to understand.

Here, at the top-most level there is main system denoted by "Root". After it, there is the name of the main module "Report Preparation" overseeing the entire process and functionality of the system. Similarly, it consists of several sub modules like "Getting all the required details, Verifying and validating required details, choosing report type, generating reports and displaying generated report to the system i.e. Staff". These modules interact with various data store like "Staff Records, User Records, Financial Records" to get the required details.

Now, when the staff request to generate report at first “Get all the Require Details Submodule” get all the required details from the staff, users, financial records datastore, process the data and sent it to the “Verify and Validate Submodule” which verified the required details and sent it to the “Choose Report Type Submodule” then this submodule select the desired report type on the basis of verified and validated details of those components and generate report accordingly. Similarly, the generated report is delivered to “Display Generated Submodule” which finally exports the generated report to the staff to conclude the report generation module process.

Thus, the generated report allows staffs to evaluate the user performance, along with financial revenue of the institute. Also, it contributes in reviewing in report allowing staff to contribute for strategic planning and decision making process of the institute.

Module Specification of Report Preparation:

The module specification for the module “Report Preparation” is a detailed description of the document outlining all the specifications, functionalities, and requirements of the system that generates the report for the system.

It consists of name and purpose of the module, pseudocode, input parameters, output parameters, Global Variables, Local Variables, Calls (references to another module) and Called By (name of other module using this module).

The module specification for the **“Report Preparation”** module is shown below:

Name	Report Preparation
Purpose	The purpose of this module is to accurately generate reports for users, staffs, and financial records by managing input, verifying, and validating the data, choosing report type and allowing user to view the report.
Pseudo Code	**BEGIN** Report Preparation **DO** **ACCESS** Staff Data **ACCESS** User Data **ACCESS** Financial Data **STORE** all the access staff, user, financial data in Staff, User, and Financial Records Datastore **GET** Staff Details from Staff Records Datastore **GET** User Details from User Records Datastore **GET** Financial Details from Financial Records Datastore **PROCESS** Staff Details **PROCESS** User Details **PROCESS** Financial Details **GENERATE** Staff Reports **GENERATE** User Reports

	GENERATE Financial Reports **VALIDATE** AND **VERIFY** reports details **STORE** validated and verified details of Reports in Report Records Datastore **CHOOSE** Report Type **DISPLAY** Prepared Reports **END DO**
Input Parameters	Staff Details, User Details, Financial Details
Output Parameters	Prepared Reports
Global Variables	N/A
Local Variables	Staff Records, User Records, Financial Records, Staff Reports, User Reports, Financial Reports, Report ID, Report Type
Calls	Get Staffs Details from Staff Records Datastore, Get User Details from User Records Datastore, Get Financial Details from Financial Records Datastore
Called by	Root

Table 10: Module Specification of Report Preparation.

5.4 Join the Program

Student Name: Bishal Bogati

London Met ID: 22068134

Introduction

Individuals who complete the user registration process have access to a user-friendly portal designed to assist their seamless admission into institute programs. The platform displays a clear and extensive list of various courses, each supported by full descriptions, allowing consumers to simply select their desired program. Once a course is selected, the system walks consumers through a safe and simple payment procedure, offering clear information regarding course pricing and issuing a confirmation receipt following successful payment.

Users who enrolls in a program can benefit from a rich learning experience, including the ability to participate in mock assessments and official certification exams. The platform provides a simulated testing environment, allowing users to track their progress and receive instant feedback on their exam results. Candidates who are successful can receive their certificate, which includes details like course name and completion date.

5.4.1 Environmental Model Specification

The Join Program's environmental model definition describes the relationships between various environmental components and clarifies how a program is joined inside the system. The Context Level Data Flow Diagram (Level 0 DFD) shown in this standard shows how many environmental parameters interact with the join program process.

Level 0 DFD:

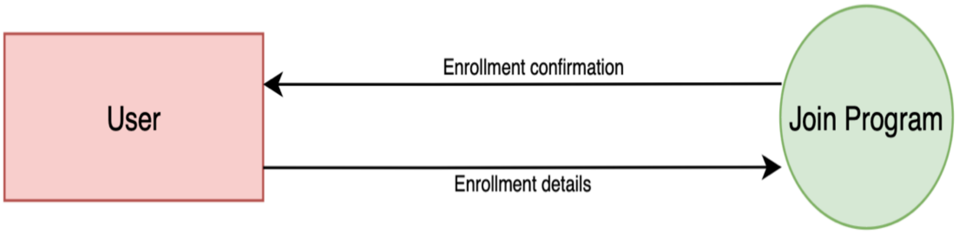


Figure 26: Context Level Diagram (Level 0) of Join Program.

Users enter enrollment information, which is necessary for programme enrollment, in this Level 0 Data Flow Diagram (DFD). This information flows from the "Users" entity to the "Join Programme" process. After the system has processed this data, users receive an "Enrollment Confirmation" that details the status of their enrollment. Without going into depth about how the "Join Programme" process operates internally, this high-level map offers a clear summary of the important interactions in the enrollment system.

5.4.2 Internal Model Specification

The joining program's internal model definition is a body of detailed documentation that describes how internal operations, external entities, and process requirements work together inside the system to perform the join program function. This documentation provides a comprehensive understanding of the join program handling by the system.

To give a more thorough depiction of the report creation procedure, it also divides the context level diagram into two separate levels, Level 1 DFD and Level 2 DFD (Data Flow Diagram). Below are the Level 1 and Level 2 DFDs for Join Program.

Level 1 DFD:

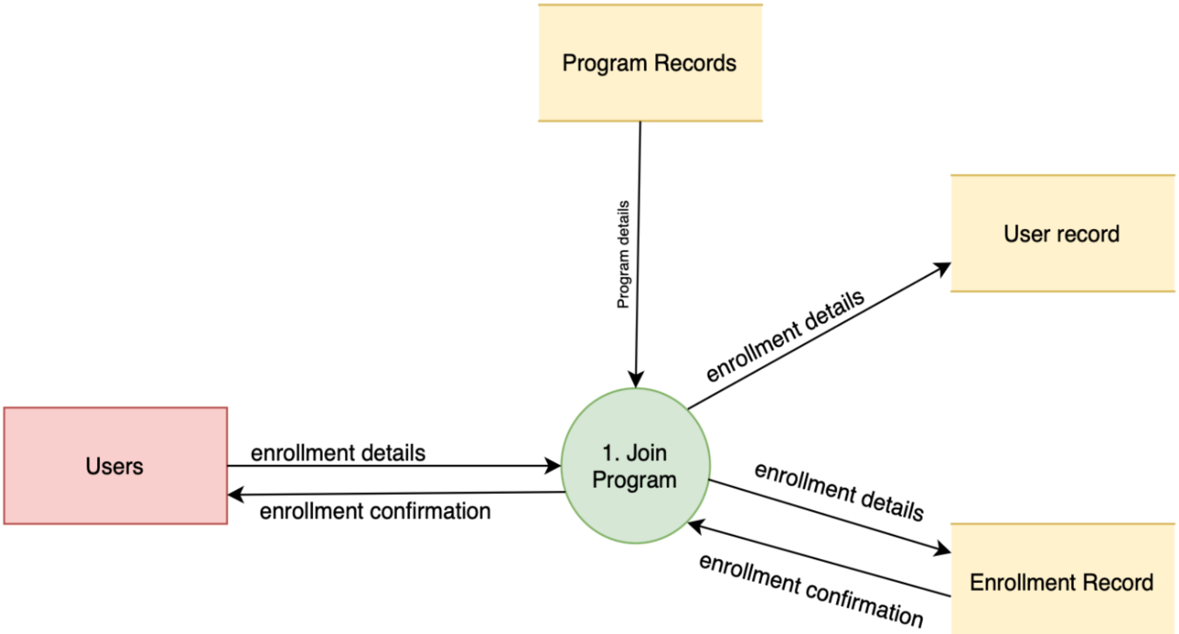


Figure 27: Level 1 DFD of Join Program.

Users send "Enrollment Details" to the Level 1 Data Flow Diagram's "Join Programme" method (DFD). This data is contained in the "User Record" and "Enrollment Record" data stores. Users receive an "Enrollment Confirmation" from the method, which also simultaneously gathers "Programme Details" from the "ProgramRecords" data store. This DFD captures the important conversations and information flows that occur during the user enrollment process.

Level 2 DFD:

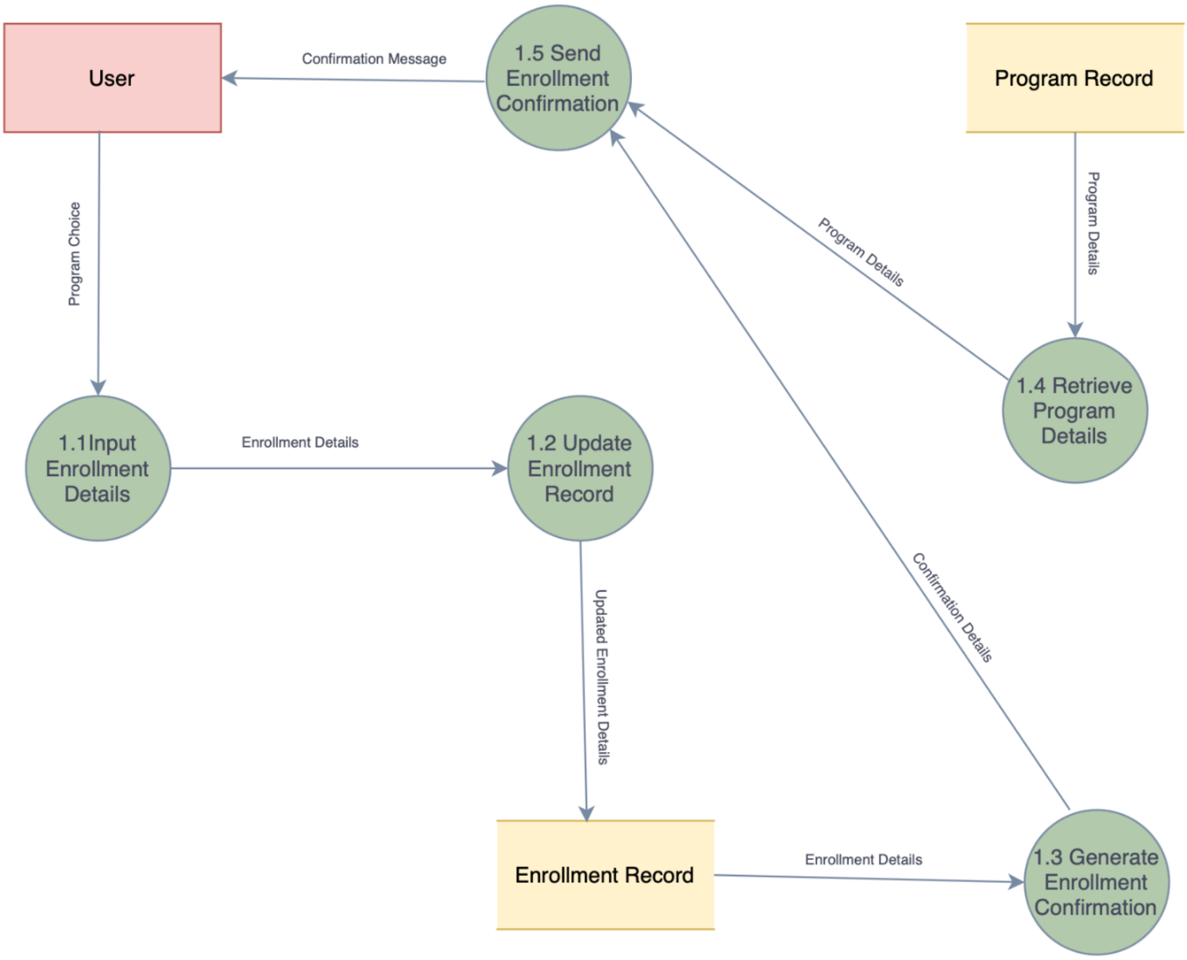

User
Confirmation Message
1.5 Send Enrollment Confirmation
Program Record
Program Choice
Program Details
Program Details
1.4 Retrieve Program Details
1.1Input Enrollment Details
Enrollment Details
1.2 Update Enrollment Record
Confirmation Details
Updated Enrollment Details
Enrollment Record
Enrollment Details
1.3 Generate Enrollment Confirmation

Figure 28: Level 2 DFD of Join Program.

The Data Flow Diagram (DFD) outlines a system for managing user enrollments and generating confirmation messages. Users input program choices, which flow to the "Input Enrollment Details" process. The chosen program is updated in the "Enrollment Record," and program details are retrieved from "Program Records." The system generates a confirmation message in the "Generate Enrollment Confirmation" process, which is then sent back to users through the "Send Enrollment Confirmation to Users" process. The key data stores are "Program Records" and "Enrollment Record." This DFD offers a concise overview of how user data moves through processes to generate enrollment confirmations for joining the program.

5.4.3 Design Specification

A comprehensive structural document that includes parameters, attributes, requirements, specifications, and pseudocode is the Design Specification for the Join Program module. This document serves as the basis for creating an efficient report within the system by combining module requirements with a structural chart.

Structure Chart for Joining the Program

The report preparation module's structural chart describes each component's specific role and shows how they are arranged hierarchically inside the system. This hierarchy's tiers correlate to different systemic levels of specificity or abstraction of data. The structural chart is interpreted from top to bottom and left to right, with different symbols denoting the hierarchy's many processes. At the top of the model is the main module, or the system itself. Subsequently, the system branches out into other modules, which in turn leads to more levels and sub-levels. Below is the structure chart for the "Join Program" module:

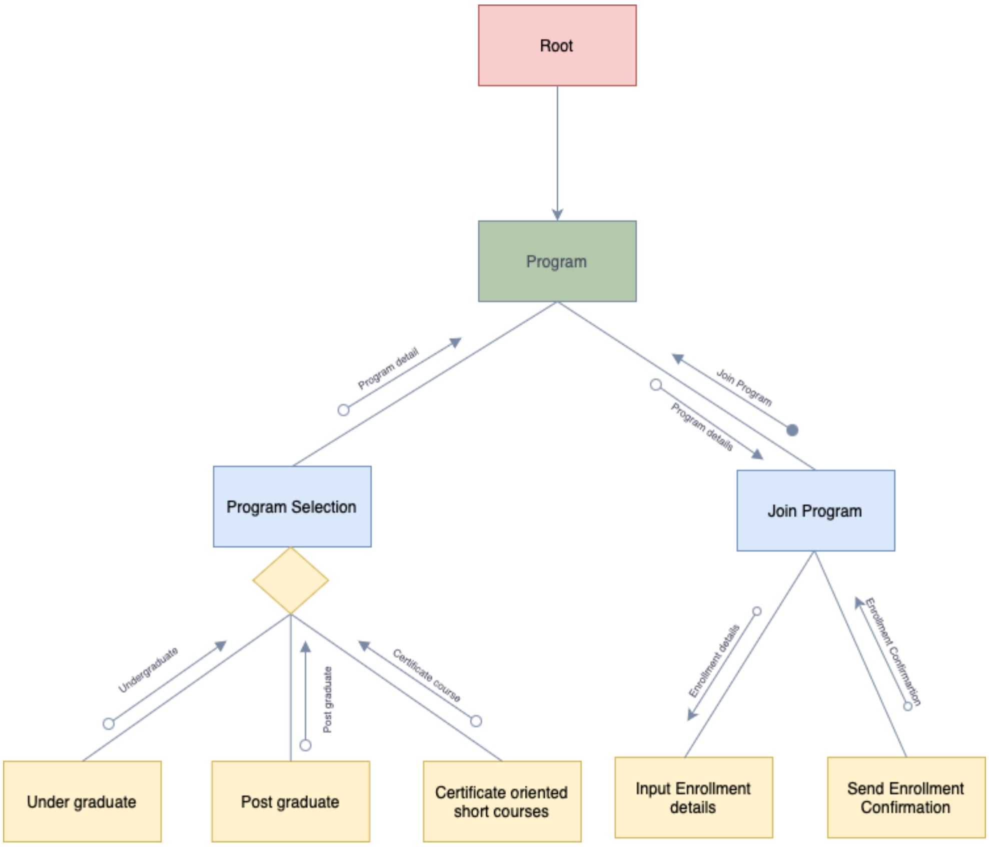


Figure 29: Structure Chart of Join Program.

The "Program" module is the main emphasis of the structure chart, which illustrates how a software system is organised. "Program" is not identified as the primary module, although it seems to be a key element, branching into two essential functions: "Join Program" and "Program Selection." The former includes subsections such as "Postgraduate," "Undergraduate," and "Certificate Oriented Short Courses," which are all focused on different facets of choosing a program. For the latter, there are modules like "Input Enrollment Details" and "Send Enrollment Confirmation," which handle enrolling in a programme of choice and verify enrollments. The aspects of the system are represented in a systematic and modular manner by this hierarchical organisation, which also clarifies the data flow and interactions between various software modules.

Module Specification for Joining the Program.

The characteristics, features, and prerequisites of the system that are needed to join program are covered in full in the module specification for the "Join Program” module. Calls (references to other modules), called by (the name of another module that uses this module), input and output parameters, global and local variables, pseudocode, and the module's name and function are all covered in detail in this extensive page.

Module Name	Join Program
Purpose	This module helps user to join program by providing the enrolment confirmation to users.
Pseudocode	**BEGIN** Join Program **FUNCTION** Enroll Enrollment Details **BECOMES CALL** UpdateEnrollment Confirmation Message **BECOMES** **CALL** Generate Confirmation **CALL** SendConfirmation **END FUNCTION** **FUNCTION** UpdateEnrollment **RETURN** Updated EnrollmentDetails **END FUNCTION** **FUNCTION** GenerateConfirmation **RETURN** ConfirmationMessage **END FUNCTION** **FUNCTION** SendConfirmation **END** FUNCTION **END** Join Program
Input Parameters	Program Choice, Enrollment Details , Program Details
Output Parameters	Confirmation Message
Global Variables	User_ID
Local Variables	Program_ID,Program_Name,Program_Type, Program_Fee
Calls	Generate Confirmation, Send Confirmation
Called By	Root

Table 11: Module Specification of Join Program.

5.5. Take Certification Exam

Student Name: Siddhant Bhurtel

London Met ID: 22068753

Introduction

Members who have joined the program are required to Take a certification exam in order to get a certificate. The “Take Certificate Exam” feature permits horticultural enthusiasts to pass specific exams to verify their knowledge. The exam interfaces are user-friendly, allowing users to browse a variety of courses and receive instant feedback after finishing them.

The goal of the McGregor Institute's certification process integration within its educational offerings is to create a thriving community of plant enthusiasts. The "Take Certification Exam" function will be explained in detail with Environmental Model Specification like DFD and Design Specification like Structure Chart

5.5.1. Environmental Model Specification

The interactions between several environmental conditions and the system’s certification exam process are depicted in the environmental model specification of this plant purchase. It illustrates the **Level 0 DFD, or Context Level, Data Flow Diagram.**

Context Level Diagram (Level 0)

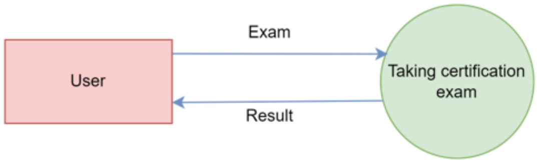


Figure 30: Context Level Diagram (Level 0) of Take Certification Exam.

This context level diagram of Taking certification exam, the key elements used are, User as an external entity, Taking certification exam as a process, and Exam and result as a data flow between them.

In the Level 0 DFD, the “User” submits an “Exam” to the “Taking a certificate” process. This process then returns a “Result” back to the “User”. Essentially, the user provides an exam, it’s processed, and the result is returned to the user.

5.5.2. Internal Model Specification

The Internal Model Specification for this report preparation provides a thorough description that explains how the system's internal operations, external entities, and process specification work together to carry out the taking the certification exam. It makes the process in the system clear.

It further divides the context level diagram into 2 different level of diagrams i.e. Level 1 DFD and Level 2 DFD. The Level 1 and Level 2 DFD (Data Flow Diagram) of Taking Certification Exam are shown below:

Level 1 DFD

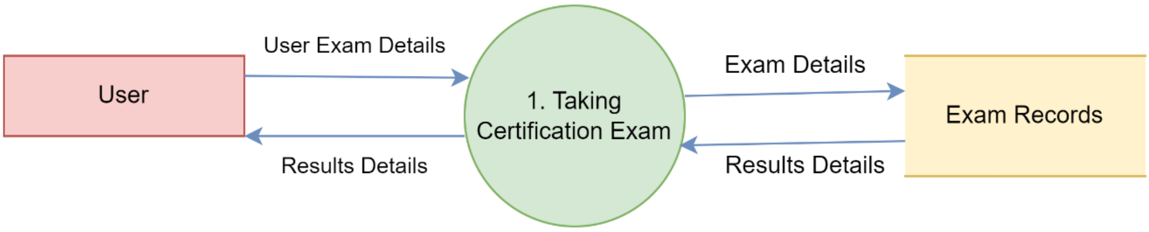


Figure 31: Level 1 DFD of Take Certification Exam.

In this Level 1 DFD of Taking Certification Exam the key elements used are user as an external entity, Taking Certification Exam as a process, Exam Records as data store and user exam details and result details as a data flow within this module of the system.

The figure illustrates the process of a user taking a certification exam. The user provides the exam details to the "Taking the certification exam" process and the exam details of the user are sent to "Exam record database which stores the information for future reference. The result details are sent back to the user, thereby successfully completing the process.

Level 2 DFD

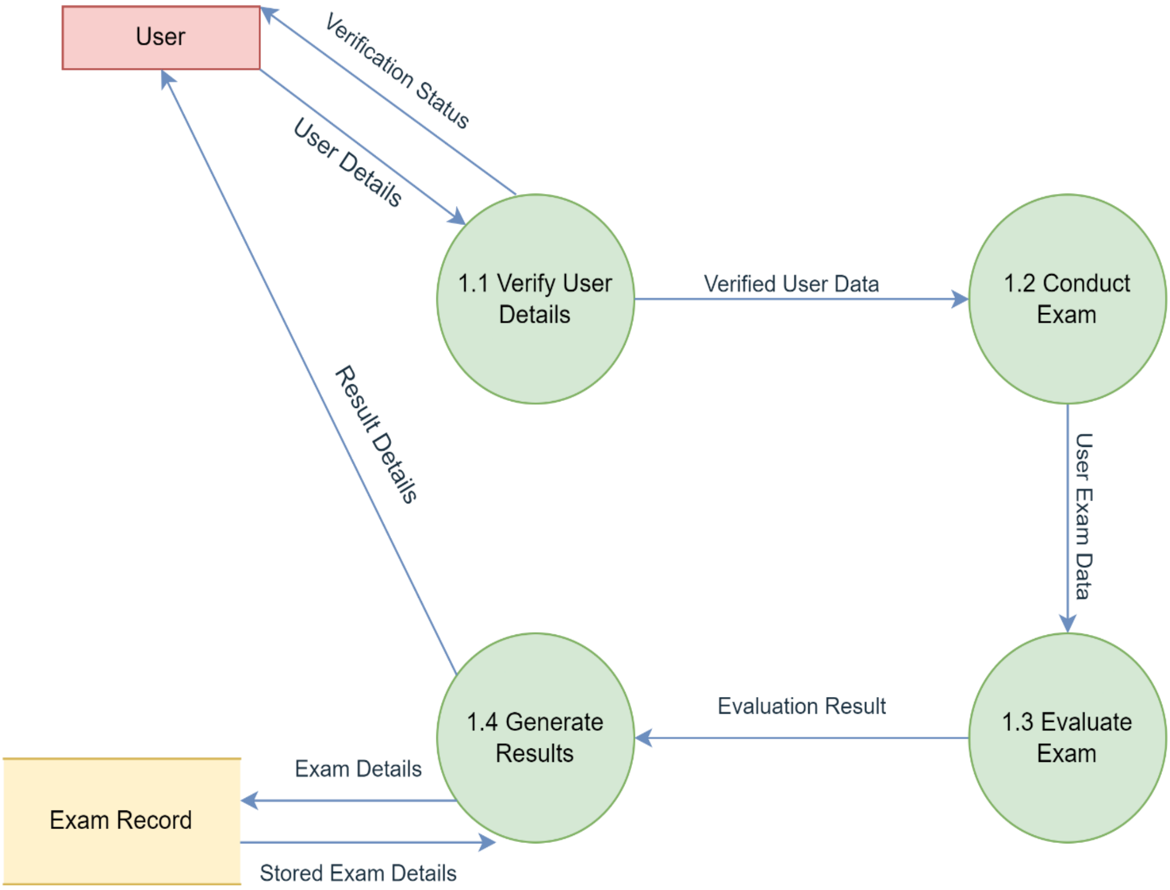


Figure 32: Level 2 DFD of Take Certification Exam.

In this Level 2 DFD of Taking Certification Exam the key elements used are user as an external entity, Verifying User Details, Conduct Exam, Evaluate Exam and Generate Results as a process, Exam Records as data store and user details, verification status, verified user data, Evaluation result, exam details and result details as a data flow within this module of the system.

The Level 2 Data Flow Diagram (DFD) outlines a system for conducting and evaluating exams. The process begins with a "User" providing their details to the "Verify User Details" process. This process checks the user's information and sends back a verification status. If the user's details are verified, the "Verified

User Data" is sent to the "Conduct Exam" process. Here, the exam takes place and the user's exam data is collected. This "User Exam Data" is then sent to the "Evaluate Exam" process, which processes the data to generate an "Evaluation Result". This result is sent to the "Generate Results" process, which also receives "Stored Exam Details" from an "Exam Record". The "Generate Results" process combines the evaluation result and the stored exam details to generate the final results. These result details are sent back to the user completing the flow of the DFD.

5.5.3. Design Specification

The Design Specification of Taking Certification Exam includes a full list of all the parameters, properties, specifications, needs, and pseudocode of the Taking Certification Exam module. It is composed of the module requirements and structural chart, which act as a basis for creating an efficient report within the system.

Structure Chart

The taking certification exam module's structure chart shows how various pieces of the system are arranged hierarchically, along with their respective functions. Here, each tier of the hierarchy represents a distinct systemic level of data abstraction or data detail.

The structural chart's elements are interpreted top to bottom and left to right. Various symbols are employed to signify distinct processes within the hierarchy.

The primary module, or the system, is located at the top of the model. From there, the system branches out to other modules, which in turn branch out to sub-modules, and so on.

Below is the module "Taking certification exam" structure chart

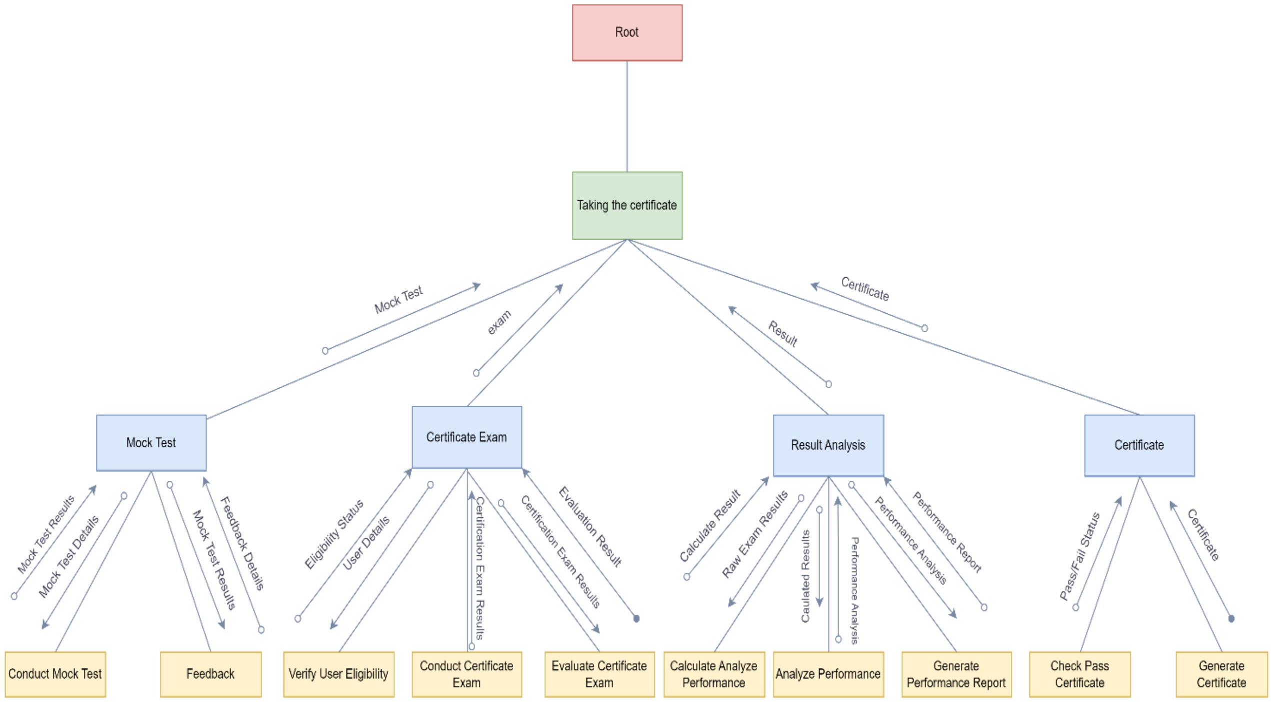


Figure 33: Structure Chart of Take Certification Exam.

This structure chart outlines a process for obtaining a certificate, starting with a “Main” process that branches into “Mock Test,” “Certificate Exam,” “Taking the certificate,” and “Result Analysis.” The “Mock Test” involves conducting a test, providing feedback, and verifying user eligibility. The “Certificate Exam” includes conducting and evaluating the exam, calculating and analysing performance, generating a performance report, and confirming the scenario. The “Taking the certificate” process confirms exam results, checks prior certificates, compares with prior certificates, and issues the certificate. Lastly, the “Result Analysis” module likely analyses the performance report for more detailed insights. This chart provides a comprehensive overview of the process to obtain a certificate.

Module Specification of Take Certification Exam

The document describing all the specifications, features, and needs of the system that creates the system report is contained in the module specification for the "Taking certification exam" module.

It includes the module's name and purpose, pseudocode, input and output parameters, global and local variables, calls (which are references to other modules), and called by (which is the name of the other module that uses this module).

The following is the module specification for the "Taking certification exam" module:

Name	Taking certification exam
Purpose	The purpose of this module is to assess and validate an individual's knowledge, skills, and proficiency in the specific subject or field and providing them a certification exam, analysing their result and providing them their progress certificate
Pseudocode	**BEGIN** TakingCertificationExam: **DO** **ACCESS** Mock Test **ACCESS** Certificate Exam **ACCESS** Result Analysis **ACCESS** Certificate **FETCH** MockTest Details **FETCH** CertificateExam Details **FETCH** ResultAnalysis Details **FETCH** Certificate Details

	PROCESS MockTest Details **PROCESS** CertificateExam Details **PROCESS** ResultAnalysis Details **PROCESS** Certificate Details **GENERATE** MockTest Reports **GENERATE** CertificateExam Reports **GENERATE** ResultAnalysis Reports **GENERATE** Certificate Reports **DISPLAY** Certificate **END DO**
Input Parameters	Mock Test Details, Certificate Exam Details, Result Analysis Details, Certificate Details
Output Parameters	Mock Test Reports, Certificate Exam Reports, Result Analysis Report, Certificate Reports
Global Variables	User Id
Local Variables	Mock Test Details, Certificate Exam Details, Result Analysis Details
Calls	Access, Fetch, Process, Generate, Display
Called by	Root

Table 12: Module Specification of Take Certification Exam.

6. Summary

According to our scenario's needs we have successfully completed the design of our system, which has been implemented for McGregor Institute of Botanical Training. We worked hard to understand the complexities of "Structured Software Engineering" and related topics as we carefully worked through this group assignment. We used tools such as the Data Flow Diagram, Structure Chart, Data Dictionary, Module Specification, and ER Diagram, to suit the project requirements. Using these fundamental components ensured a strong and organized software engineering solution by acting as the framework for our methodical approach. Each member from our group worked on the design specifications, internal model specifications, and environmental model specifications for distinct functions in the section dedicated to individual activities.

The goal of this McGregor Institute of Botanical Training coursework is to demonstrate practical knowledge of "Structured Software Engineering" (Yourdon) in a small group of 5 members setting within a given time limit. Our project's objective is to improve the institute's operations in order to meet the growing demand for horticulture and agriculture. The features that the system include are user registration, program enrolment, plant purchases, payments, expert suggestions, report preparation, certification tests, and a forum for plant enthusiasts, and notifications are all included in the system design.

In general, the course aims to provide a well-organized software solution that was customized to meet the changing needs of McGregor Institute while exhibiting high levels of technical expertise and productive teamwork. Since we were somewhat new to this whole concept, we encountered some difficulties with the coursework. It was a little intimidating to deal with DFD, Structure Chart, and SRS. As a first phase, we extracted the fundamental rules from the context diagram that will be applied in the next steps. We then created Level 1 and Level 2 DFD to provide more complexity to our system. We took care to record every important detail and documented the entire procedure. While developing the DFD, Structure Chart, and Data Dictionary we had numerous conversations regarding the queries. We had multiple meetings to choose the best course of action for the project, allocate our time and resources, and address any possible problems.

We sincerely thank Mr. Rubin Thapa, our module leader, and Mr. Dipesh Raj Adhikari, our module teacher, for their outstanding assistance during the course. Their advice was really helpful to us as we navigated the project's difficulties and intricacies. In addition to helping us solve problems, Mr. Thapa and Mr. Adhikari gave us the assurance and confidence we needed to handle the coursework successfully. Throughout our many conversations and meetings, their knowledge, warmth, and dedication to our comprehension were clear. Their support created an environment that was conducive to learning and gave us the confidence to tackle the challenges at hand. Their commitment to our educational journey and their crucial role in creating a fulfilling and enriching coursework experience are deeply appreciated.

This course really helped us learn a lot and be ready for different challenges. In summary, we find that Structure Software Engineering delivers significant advantages at every stage of the creation process and can be applied to a wide range of systems. It is comparable to having a good set of tools that you can use to build anything from the ground up. Our comprehension of the various aspects of software development has improved as a result of this course, which covered everything from fundamental concepts like context diagrams and DFDs to more complex concepts like structure charts and data dictionaries the course work gave us a good understanding of how to handle different parts of making software.

7. References

GeekforGeeks, 2024. *Levels in Data Flow Diagrams (DFD) - GeeksforGeeks.* [Online] Available at: https://www.geeksforgeeks.org/levels-in-data-flow-diagrams-dfd/ [Accessed 2 January 2024].

GeeksforGeeks, 2023. *Non-functional Requirements in Software Engineering - GeeksforGeeks.* [Online]
Available at: Non-functional Requirements in Software Engineering - GeeksforGeeks
[Accessed 20 December 2023].

Holmes, T., 2023. *What is a structure chart? (with picture).* [Online] Available at: https://www.smartcapitalmind.com/what-is-a-structure-chart.htm [Accessed 27 December 2023].

Jafari, L., 2020. *What Are Functional Requirements? Types and Examples - WINaTALENT Blog.* [Online]
Available at: https://winatalent.com/blog/what-are-functional-requirements-types-and-examples/
[Accessed 18 December 2023].

Lucid Software Inc., 2024. *What is a Data Flow Diagram | Lucidchart.* [Online] Available at: https://www.lucidchart.com/pages/data-flow-diagram#:~:text=DFD%20Level%200%20is%20also,its%20relationship%20to%20external%20entities.
[Accessed 2 January 2024].

Lucidchart, 2024. *What is a Data Flow Diagram | Lucidchart.* [Online] Available at: https://www.lucidchart.com/pages/data-flow-diagram#:~:text=DFD%20Level%200%20is%20also,its%20relationship%20to%20external%20entities.
[Accessed 2 January 2024].

Nalimov, C., 2020. *Crow’s foot notation in entity-relationship diagrams.* [Online] Available at: https://www.gleek.io/blog/crows-foot-notation [Accessed 21 December 2023].

Rouse, M., 2017. *Entity-Relationship Diagram.* [Online]
Available at: https://www.techopedia.com/definition/1200/entity-relationship-diagram-

erd
[Accessed 21 December 2023].

Rouse, M., 2023. *What Does Process Specification Mean?.* [Online] Available at: https://www.techopedia.com/definition/28636/process-specification

S.Gills, A., 2023. *What is a project charter?.* [Online] Available at: https://www.techtarget.com/searchcio/definition/project-charter

Varma, S., 2023. *Data Dictionary in DBMS.* [Online] Available at: https://www.scaler.com/topics/data-dictionary-in-dbms/
[Accessed 21 December 2023].

Visual Paradigm, 2023. *Navigating System Complexity: A Comprehensive Guide to Data Flow Diagram Levels - Visual Paradigm Guides.* [Online] Available at: https://guides.visual-paradigm.com/navigating-system-complexity-a-comprehensive-guide-to-data-flow-diagram-levels/
[Accessed 2 January 2023].