# AMAL LIBRARY

## PROJECT REPORT

Submitted by:

| | |
|---|---|
| **NAVARAS P** | **(Reg No.: AZATSCS013)** |
| **ASWIN T** | **(Reg No.: AZATSCS012)** |
| **FARIS RAHMAN PP** | **(Reg No.: AZATSCS035)** |
| **SHEBIN K** | **(Reg No.: AZATSCS043)** |

*For the award of the Degree of*

***Bachelor of Science (B.Sc.)***

***In Computer Science***

*(University of Calicut)*

**AMAL COLLEGE OF ADVANCED STUDIES**

*(Affiliated to the University of Calicut)*

*Santhigramam, Myladi*

*Eranhimangad P.O*

*Nilambur, Kerala 679329*

**APRIL– 2022**

# ACKNOWLEDGMENT

First, I thank the ALMIGHTY GOD for showering me with his immense blessing and giving strength to complete my project work successfully. I wish to express my sincere gratitude to **Dr Zacaria TV**, Principal for providing us with adequate facilities to do this work. I am grateful to **Mrs Niloofer S**, Assistant professor of the Department of Computer Science for bringing this report to successful completion. Last but not the least, I express my sincere thanks to all the staff in the Department of Computer Science, all my friends, and my parents who have patiently extended all sorts of help in accomplishing this undertaking

Date:

NAVARAS P
(Reg No: AZATSCS013)

ASWIN T
(Reg No: AZATSCS012)

FARIS RAHMAN PP
(Reg No: AZATSCS035)

SHEBIN K
(Reg No: AZATSCS043)

# DECLARATION

We, hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person or material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text

| | | |
|---|---|---|
| Place: NILAMBUR | Name: | Signature: |
| Date: | NAVARAS P<br>(Reg No: AZATSCS013) | |
| | ASWIN T<br>(Reg No: AZATSCS012) | |
| | FARIS RAHMAN P P<br>(Reg No: AZATSCS035) | |
| | SHEBIN K<br>(Reg No: AZATSCS043) | |

## CERTIFICATE

# CONTENTS

# ABSTRACT

‘AMAL LIBRARY’ is a web application that is used as the blog for the library of Amal College of Advanced Studies. These are an extension of what we already do: identify, organize, and make information accessible in libraries. They allow us to be more responsive, to reach out to the faculty and students via our library blogs to highlight news, post student/faculty invite comments, announce events, etc. It is introducing a Web Application that creates a Content Management System for libraries. It helps library management update the news and other details easily and comfortably. It has a page that shows the availability of library books and a question bank. Blogs are a simple and efficient way for librarians to stay informed and for libraries to disseminate information promptly. As previously, the library has a blog in Blogspot Content Management System. It is not responsive; the availability of books is not included in the blog. It works under the limited atmosphere of the Blogspot Content Management System. The Question Bank that is uploaded already is not effective. The redirected pages on the blog do not have any content at all. It contains a lack of functionalities, and some functions are not properly working. It is not user-friendly. Like other developments, the Blog is also approaching library science to think about its uses. In question bank, it is a student and faculty-friendly question bank which can easily access model question papers. Also, it shows the availability of books and needed people can block books for their needs. The obvious use of weblogs in libraries is to set one up to deliver news to patrons.

## List of figures

## List of Tables

CHAPTER 1

# INTRODUCTION

‘AMAL LIBRARY’ is a web application that is used as the blog for the library of Amal College of Advanced Studies. These are an extension of what we already do: identify, organize, and make information accessible in libraries. They allow us to be more responsive, to reach out to the faculty and students via our library blogs to highlight news, post student/faculty invite comments, announce events, etc. As previously, the library has a blog in Blogspot Content Management System. It is not responsive, and the availability of books is not included in the blog. It works under the limited atmosphere of the Blogspot Content Management System. The Question Bank that is uploaded already is not effective. The redirected pages on the blog do not have any content at all. It contains a lack of functionalities, and some functions are not properly working. It is not user-friendly. Like other developments, the Blog is also approaching library science to think about its uses. In question bank, it is a student and faculty-friendly question bank which can easily access model question papers. Also, it shows the availability of books and needed people can block books for their needs. It introduces a Web Application that creates a Content Management System for libraries. It helps library management update the news and other details easily and comfortably. It has a page that shows the availability of library books and question banks. Blogs are a simple and efficient way for librarians to stay informed and for libraries to disseminate information promptly. The obvious use of weblogs in libraries is to set one up to deliver news to patrons.

CHAPTER 2

# PROBLEM DEFINITION AND METHODOLOGY

## 2.1 Problem Definition

Currently, the library has a blog in Blogspot Content Management System. It is not responsive; the availability of books is not included in the blog. It works under the limited atmosphere of the Blogspot Content Management System. The Question Bank that is uploaded already is not effective. The redirected pages on the blog do not have any content at all. It contains a lack of functionalities, and some functions are not properly working. It is not user-friendly.

## 2.2 Project Overview

In this project, it is planning to create a Blog Content Management System for the library which works in an owned space and helps library management update the news and other details easily and comfortably. And the viewers and admins can understand, analyze, and administrate easily. It is built more effective and visually comfortable theme. The viewer and admin sides are more interactive and responsive.

## 2.3 Methodology

AGILE methodology is a practice that promotes continuous iteration of development and testing throughout the software development lifecycle of the project. Both development and testing activities are concurrent, unlike the Waterfall model. Agile software development emphasizes four core values.

- Individual and team interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation Responding to change over following a plan.

**Phases of Agile Model:**

1. **Requirements gathering:** In this phase, you must define the requirements. You should explain business opportunities and plan the time and effort needed to build the project. Based on this information, you can evaluate technical and economic feasibility.

2. **Design the requirements:** When you have identified the project, work with stakeholders to define requirements. You can use the user flow diagram or the high-level UML diagram to show the work of new features and show how they will apply to your existing system.

3. **Construction/ iteration:** When the team defines the requirements, the work begins. Designers and developers start working on their project, which aims to deploy a working product. The product will undergo various stages of improvement, so it includes simple, minimal functionality.

4. **Testing:** In this phase, the Quality Assurance team examines the product's performance and looks for the bug.

5. **Deployment:** In this phase, the team issues a product for the user's work environment.

6. **Feedback:** After releasing the product, the last step is feedback. In this, the team receives feedback about the product and works through the feedback.

## 2.4 Purpose

In this project, it is planning to create a Blog Content Management System for the library which works in an owned space and helps library management update the news and other details easily and comfortably. And the viewers and admins can understand, analyze, and administrate easily. It is built more effective and visually comfortable theme. The viewer and admin sides are more interactive and responsive.

## 2.5 Scope

Blogs are a simple and efficient way for librarians to stay informed and for libraries to disseminate information promptly. The obvious use of weblogs in libraries is to set one up to deliver news to patrons. We can pre-own-books and access book details we needed in online. Though it has extremely high efficiency in using it globally and in the future.

CHAPTER 3

# REQUIREMENT ANALYSIS AND SPECIFICATION

## 3.1 Existing System

Currently, the library has a blog in Blogspot Content Management System. It is not responsive; the availability of books is not included in the blog. It works under the limited atmosphere of the Blogspot Content Management System. The Question Bank that is uploaded already is not effective. The redirected pages on the blog do not have any content at all. It contains a lack of functionalities, and some functions are not properly working. It is not user-friendly.

## 3.2 Proposed system

In this project, it is planning to create a Blog Content Management System for the library which works in an owned space and helps library management update the news and other details easily and comfortably. And the viewers and admins can understand, analyze, and administrate easily. It is built more effective and visually comfortable theme. The viewer and admin sides are more interactive and responsive.

## 3.3 Feasibility Study

Nowadays data is the most valued possession in the world, the quality of an entity is assured if enough data is available and the data is not meant to be easily collected therefore, we seek many modern solutions. And much of these solutions never encourage a collaborative approach and centralized access. This web app will be more relevant for users who collect data together and need to access them in real-time since it will be helpful for them in many ways. In this project, feasibility tests include the economical, technical, operational, and behavioural feasibility of the system.

### 3.3.1 Economical feasibility

This is an important aspect to be considered while developing a project. We decided the system to be based on the minimum possible cost factor. We will be developing this in the shortest possible time. We do not need to buy external hardware or other components for development purposes. Also, all the recourses are already available, which indicates whether the system is economically possible for development.

### 3.3.2 Technical feasibility

This included the study of function performance and constraints that may affect the ability to achieve an acceptable system. The developing system must be evaluated with the technical capability. The project is feasible within the limits of current technology. The technology we used here is website technology, which is one of the most widely used. We can easily provide the complete services provided by this application with available technology resource constraints. The latest versions of frameworks and IDEs are used for development. So, this system is technically feasible.

### 3.3.3 Operational feasibility

The application provides an effective and reliable way to manage the resources. The website is highly accurate and efficient. The simple user interface provides an easily manageable user experience, flexibility, and accuracy to the user. The server provides a fast experience to all the users all operations on the system are extremely fast. All the inputs taken are self-explanatory even to a layperson.

### 3.3.4 Behavioral Feasibility

Normal human psychology of human beings indicates that people are resistant to change, and computers are known to facilitate change. The users can trust the application and simply look through various information. This application can provide a good user experience. So, we expect that the users will accept the project with an open heart.

## 3.4 Requirement specification

The project aims to create a web app using JavaScript. It is connected to MongoDB Server for database information. Software requirement specification involves the study of the platform being used in detail and in this case the platform being used is HTML. It also involves the detailed study of the various operations performed by the system and their relationship within and outside the system.

The HTML is a standard mark-up language for creating a webpage and web applications, with cascading style sheets (CSS) and JavaScript. Its designers also leveraged may be tried and true approaches that proved to work in the wireless world. Many of these features indeed appear in existing proprietary platforms. The web browser receives HTML documents from a web server or local storage and the document into multimedia web pages. HTML describes the structure of a web page semantically and originally includes cues for the appearance of a document.

### 3.4.1 Functional Requirements

- Super Admin: Has the complete privileges of this web app. Only super admin can add/view/delete users for this system.
- User: Users logged in can add data to the form created. The data added by users will only be shared after the admin marked it as completed.

### 3.4.2 Non-Functional Requirements

- Performance: The website is compatible with major browsers and will work perfectly.
- Usability: The simple user interface, accuracy and flexibility of the website gives ease to use it.
- Efficiency: when a user uses this website, then it will be easier for him to do different activities specified on the website within a minimum amount of time.

## 3.5 Environmental Details

Environmental requirements for the smooth functioning of this product could be configured based on the requirement needed by the component of the operating environment that works as a front-end system. Here we suggest minimum configuration for both hardware and software components.

### 3.5.1 Hardware specification

The following is the hardware used for the development of the application.

| | |
|---|---|
| • Processor | : Intel Core i3 |
| • RAM | : 4 GB |
| • Hard Disk | : 512GB |
| • Input Device | : Standard QWERTY keyboard and mouse |
| • Output Device | : Monitor |

### 3.5.2 Software specification

The following is the software needed for the development of the application

| | |
|---|---|
| • IDE | : Visual Studio code |
| • Front-End | : HTML, JavaScript, CSS |
| • Back-End | : Python |
| • Database | : PostgreSQL |
| • Frameworks | : Django, Tailwind |
| • Operating Systems | : Windows 11, Linux, macOS |

### 3.5.3 Software description

**HTML**

Hypertext Markup Language (HTML) is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript. Web browsers receive HTML documents from a web server or local storage and render the documents into multimedia web pages.

HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML elements are the building blocks of HTML pages. Browsers do not display the HTML tags but use them to interpret the content of the page. HTML can embed programs written in a scripting language such as JavaScript, which affects the behaviour and content of web pages.

**CSS**

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colours, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file and reduce complexity and repetition in the structural content.

## JavaScript

JavaScript often abbreviated as JS, is a programming language that conforms to the ECMA Script specification. JavaScript is high-level, often just-in-time compiled, and multi-paradigm. It has curly-bracket syntax, dynamic typing, prototype-based object orientation, and first-class functions.

Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web.

JavaScript enables interactive web pages and is an essential part of web applications. Most websites use it for client-side page behaviour, and all major web browsers have a dedicated JavaScript engine to execute it. As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM). However, the language itself does not include any input/output (I/O), such as networking, storage, or graphics facilities, as the host environment (usually a web browser) provides those APIs.

Originally used only in web browsers, JavaScript engines are also now embedded in server-side website deployments and non-browser applications.

## PostgreSQL

PostgreSQL is a powerful, open-source object-relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads. The origins of PostgreSQL date back to 1986 as part of the POSTGRES project at the University of California at Berkeley and have more than 30 years of active development on the core platform. PostgreSQL comes with many features aimed to help developers build applications, and administrators protect data integrity, build fault-tolerant environments, and help you manage your data no matter how big or small the dataset. In addition to being free and open-source, PostgreSQL is highly extensible. For example, you can define your data types, build out custom functions, and even write code from different programming languages without recompiling your database. There are many more features that you can discover in the PostgreSQL documentation. Additionally, PostgreSQL is highly extensible: many features, such as indexes, have defined APIs so that you can build out with PostgreSQL to solve your challenges. PostgreSQL has been proven to be highly scalable both in the sheer quantity of data it can manage and in the number of concurrent users it can accommodate.

## **Python**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it extremely attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms and can be freely distributed. When the program does not catch the exception, the interpreter prints a stack trace. Python's features include –

- Easy to learn - Python has few keywords, a simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly
- Easy to read - Python code is more clearly defined and visible to the eyes
- Easy to maintain – Python source code is easy to maintain
- Portable − Python can run on a wide variety of hardware platforms and has the same interface on all platforms
- Extendable − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient
- Databases − Python provides interfaces to all major commercial databases
- GUI Programming − Python supports GUI applications that can be created and ported to many system calls, libraries, and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- Scalable − Python provides a better structure and support for large programs than shell scripting
- It supports functional and structured programming methods as well as OOP

## Django

Django is a Python-based free and open-source web framework that follows the model–template–views architectural pattern. Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to waste a lot of time for no reason. It is free and open-source. It is maintained by the Django Software Foundation (DSF), an independent organization established in the US as a 501(c)(3) non-profit. Django's primary goal is to ease the creation of complex, database-driven websites. The framework emphasizes reusability and "pluggability" of components, less code, low coupling, rapid development, and the principle of do not repeat yourself. Python is used throughout, even for settings, files, and data models. Django also provides an optional administrative create, read, update, and delete interface that is generated dynamically through introspection and configured via admin models. Some well-known sites that use Django include Instagram, Mozilla, Disqus, Bitbucket, Next-door and Clubhouse. Django's features include –

- Ridiculously fast- Django was designed to help developers take applications from concept to completion as quickly as possible.
- Reassuringly secure- Django takes security seriously and helps developers avoid many common security mistakes.
- Exceedingly scalable- Some of the busiest sites on the web leverage Django's ability to quickly and flexibly scale.

## Tailwind

Tailwind CSS is a CSS framework. It is like popular frameworks, like Bootstrap and Materializes, in that you apply classes to elements, and it styles them. But it is also atomic CSS in that one class name does one thing. While Tailwind does have Tailwind UI for pre-built componentry, you customize Tailwind to look how you want it to look.

Tailwind's other features include

- With Tailwind, you get thousands of out-of-the-box CSS classes that you just need to apply to your HTML elements.

- The names are simple, and they do an excellent job of telling you what their functions are. For example, text-sm gives your text a small font size**.** This is a breath of fresh air for people that struggle with naming custom CSS classes

- By utilizing a mobile-first approach, responsiveness is at the heart of Tailwind’s design. Making use of the sm, md, and lg prefixes to specify breakpoints, you can control the way styles are rendered across different screen sizes. For example, if you use the md prefix on a style, that style will only be applied to medium-sized screens and larger ones. Small screens will not be affected.

CHAPTER 4

# SYSTEM DESIGN

## 4.1 User of the system

The main users of the system are:

- Super admin
- Question Pool admin
- User

**Super Admin:**

The super admin will have the complete privilege in the system. The super admin can do the following:

- Add/view/delete/modify Blogpost.
- Add/view/delete/modify Blog Notifications.
- Add/view/delete/modify Blog Comments.
- Add/view/delete/modify Question Pool Admin.
- Add/view/delete/modify Question Pool.
- Add/view/delete/modify Books.
- Add/view/delete/modify Books Transaction.
- Add/view/delete/modify Users.

**Question Pool Admin:**

The question pool admin only has the power to manage the question pool in the system. Their privileges include:

- Add/view/delete/modify Question Pool.

**Users:**

Users have the duty of submitting data only. Their privileges include:

- View Books
- View Blog
- View Question Pool.
- Add/View Comments

## 4.3 Logical Design

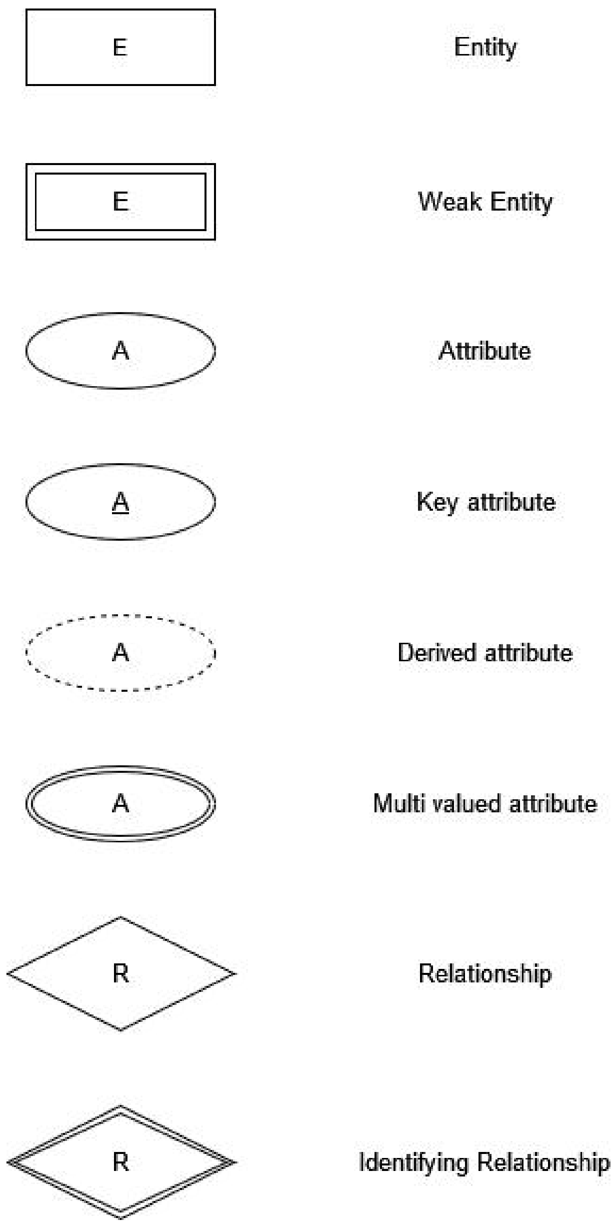
It defines the relationship between major structural elements of the program. This modular framework of a computer program can be derived from the analysis models and the interaction of the subsystems within the analysis model. The primary objective is to develop a modular program structure and represent a relationship between the modules. In addition, the logical design methods of program structure, and data structure, define an interface that enables data to flow throughout the program. The logical design of various modules in the CDAS is described using Data Flow Diagram.

## ER diagram

### 4.3.2 Entity-Relationship Diagram (ER diagram)

An Entity-Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships, and their attributes

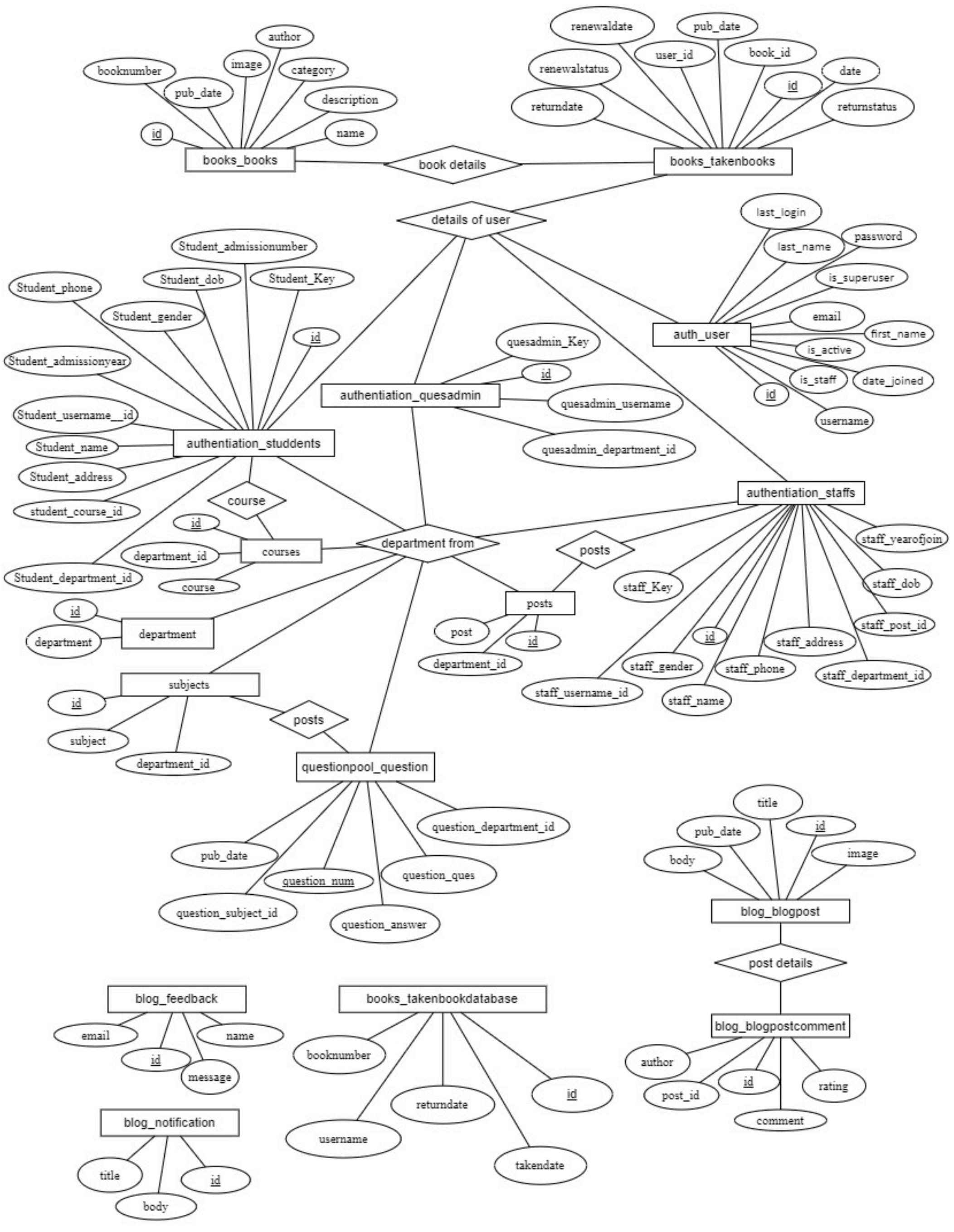
**Entity Relationship Diagram Notation:**



## ER diagram

ER diagram

## ER diagram



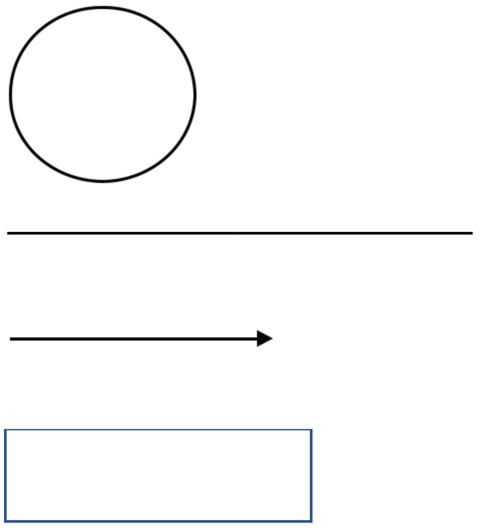
## 4.3.1 Data Flow Diagram (DFD)s

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. A data flow diagram can also be used for the visualization of data processing (structured design). It is widespread practice for a designer to draw a context-level DFD first which shows the interaction between the system and outside entities. This context-level DFD is then "exploded" to show more detail of the system being modelled.

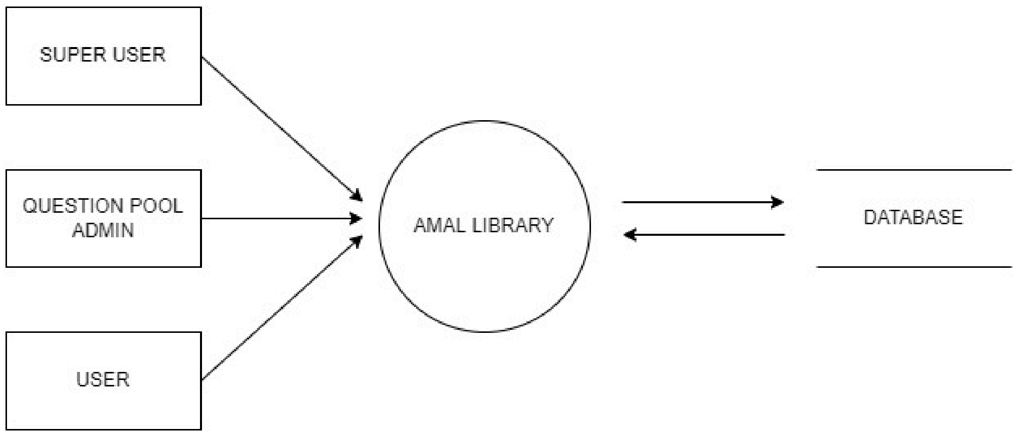
**Data Flow Diagram Notations**

We can use diverse types of notations on Data Flow Diagram. There are four basic symbols in this notation

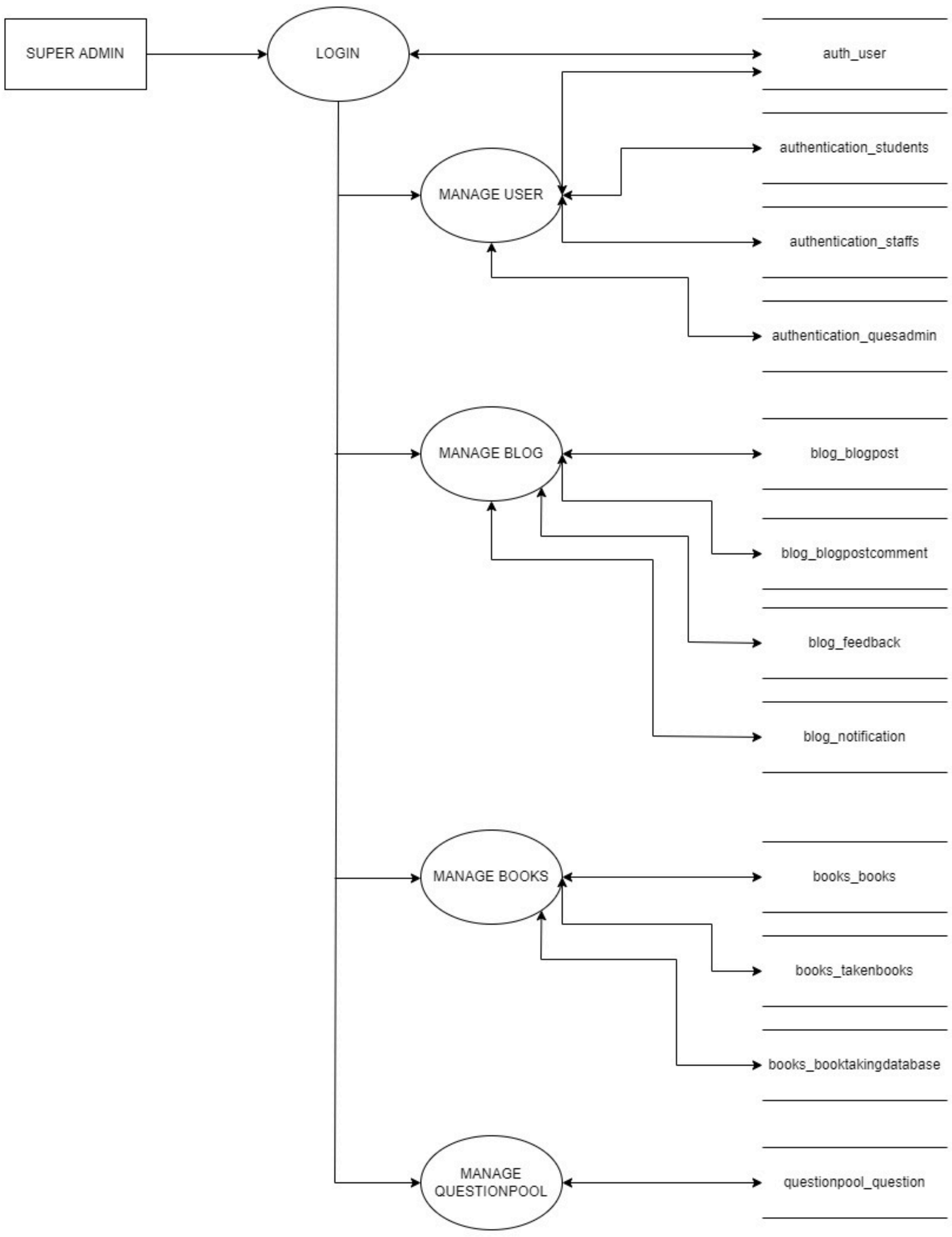
- Process
- Data Store
- Data Flow
- External Entity



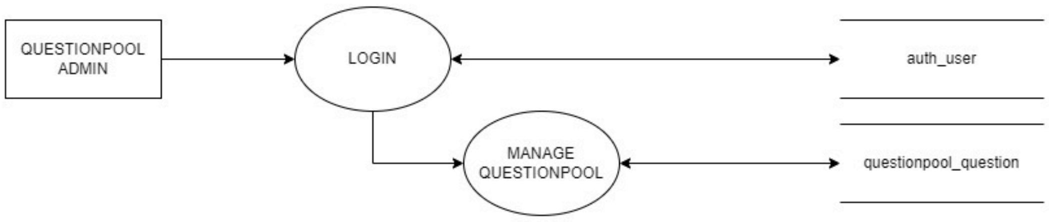
## DFD Level 0



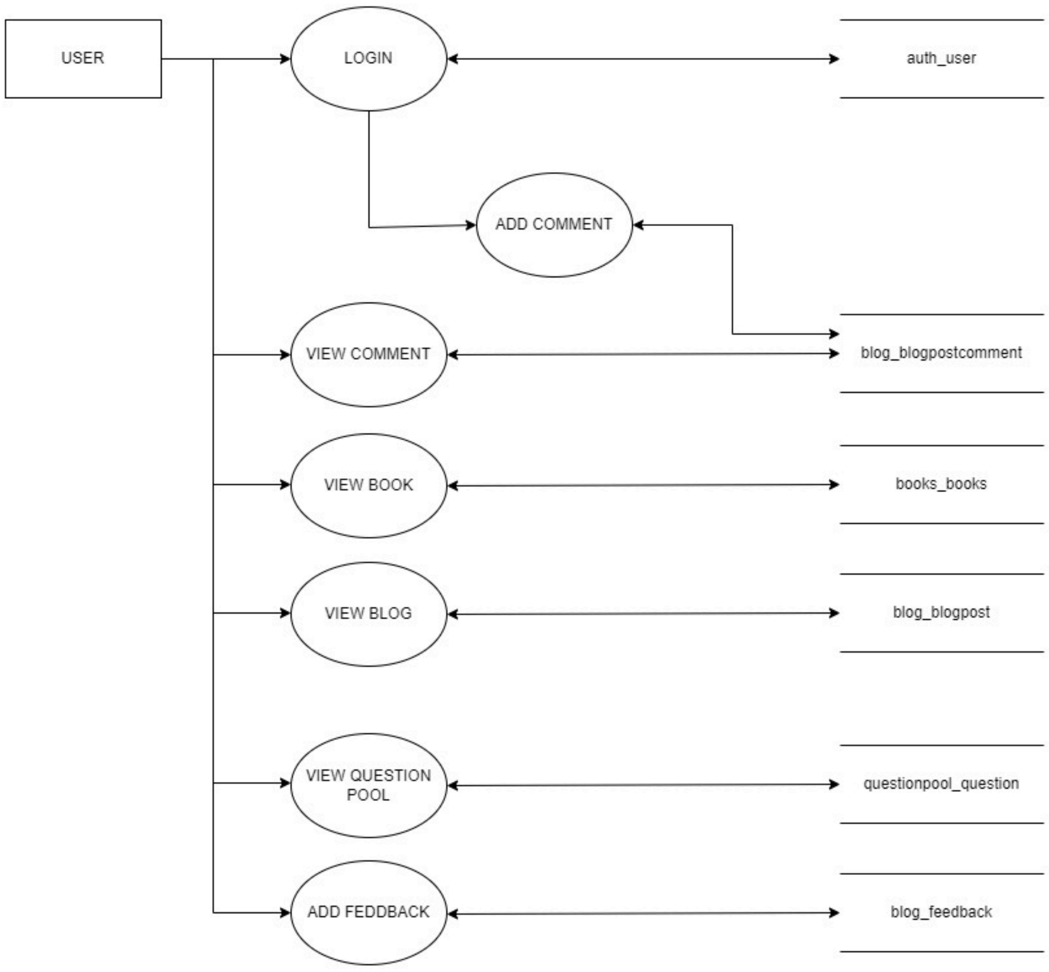
## DFD Level 1.1


SUPER ADMIN
LOGIN
auth_user
authentication_students
MANAGE USER
authentication_staffs
authentication_quesadmin
MANAGE BLOG
blog_blogpost
blog_blogpostcomment
blog_feedback
blog_notification
MANAGE BOOKS
books_books
books_takenbooks
books_booktakingdatabase
MANAGE QUESTIONPOOL
questionpool_question

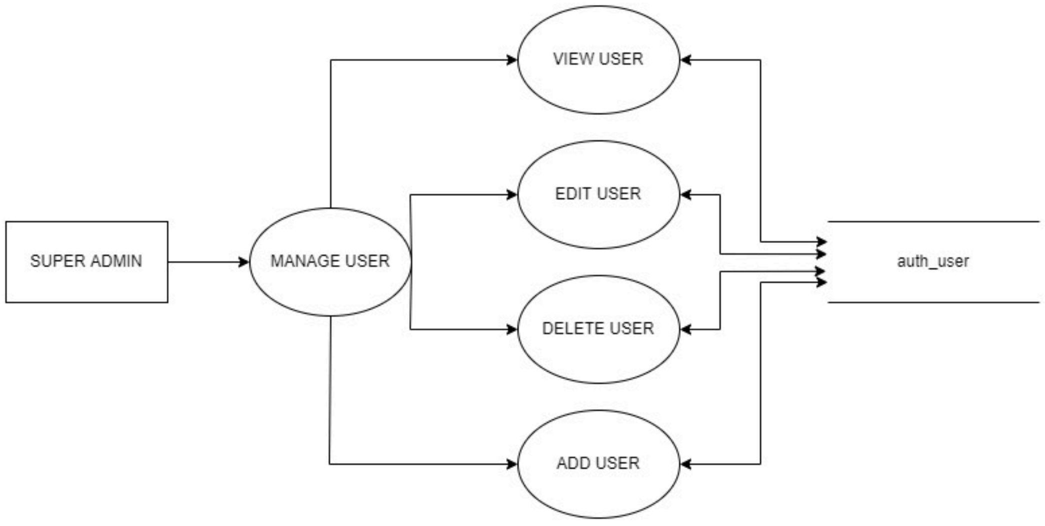
## DFD Level 1.2



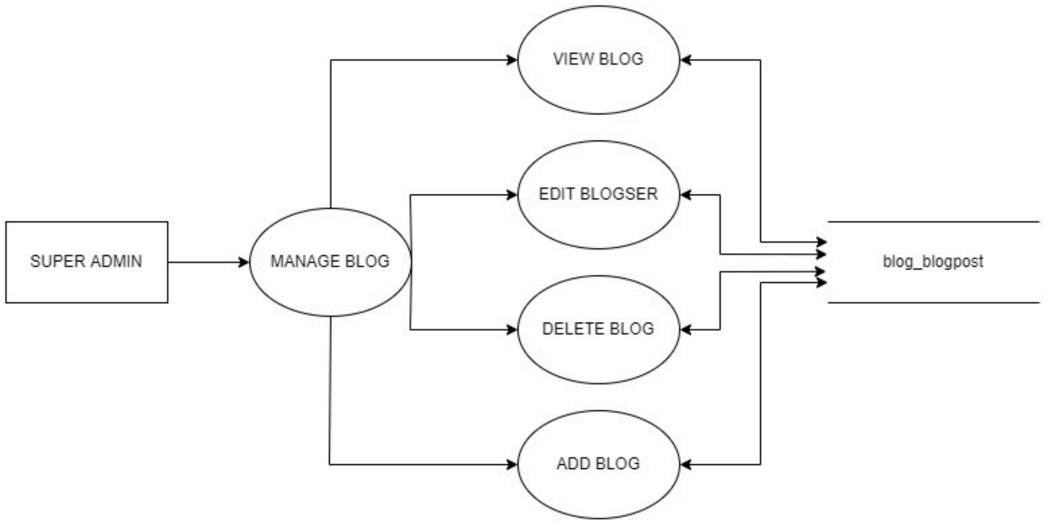
## DFD Level 1.3



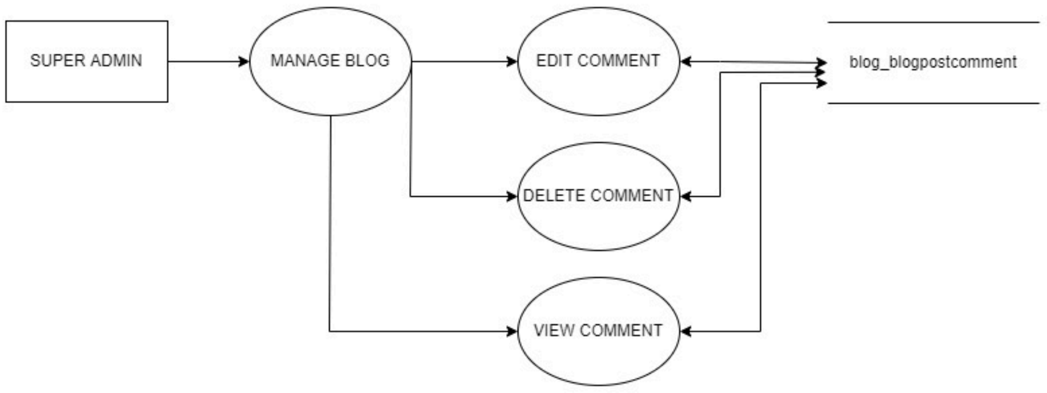
## DFD Level 2.1.1



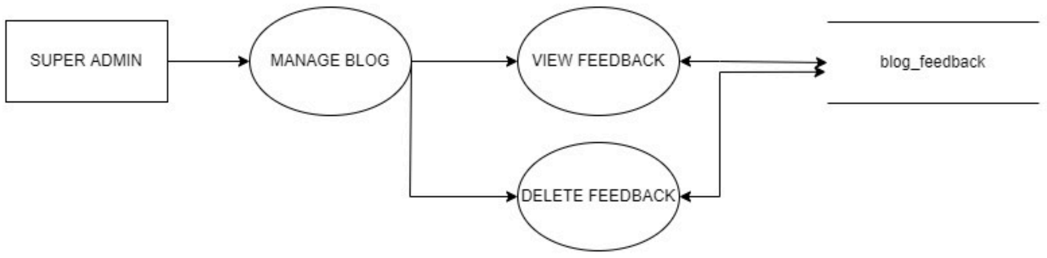
## DFD Level 2.1.2



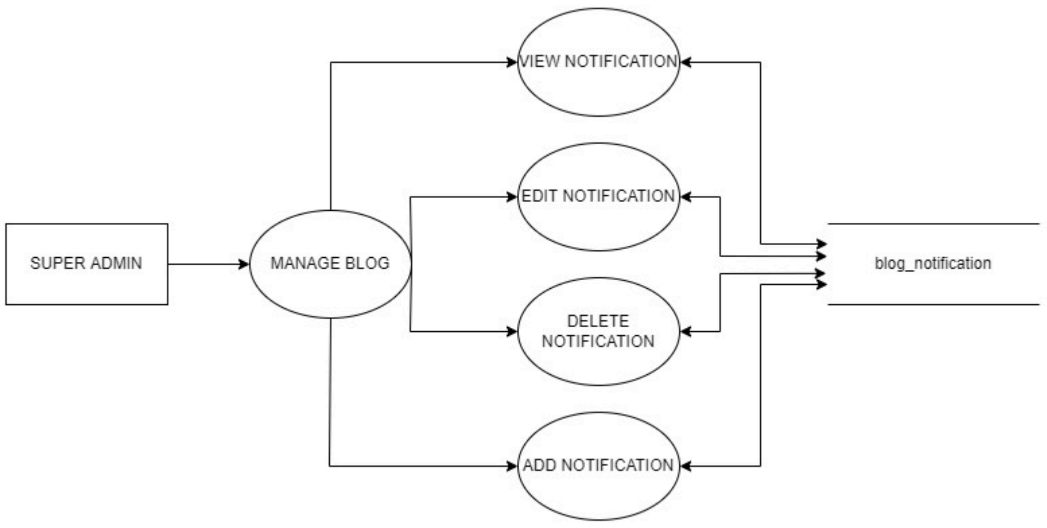
## DFD Level 2.1.3



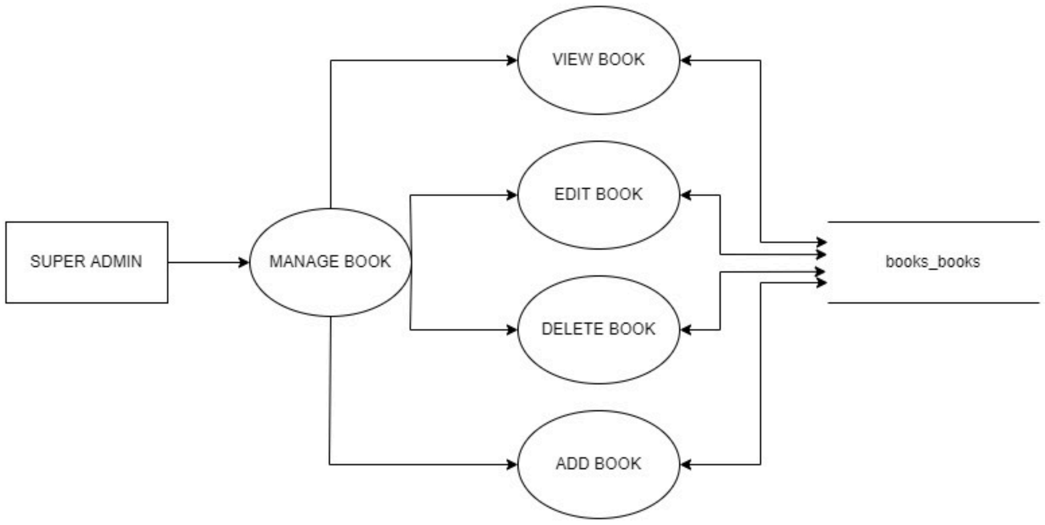
## DFD Level 2.1.4



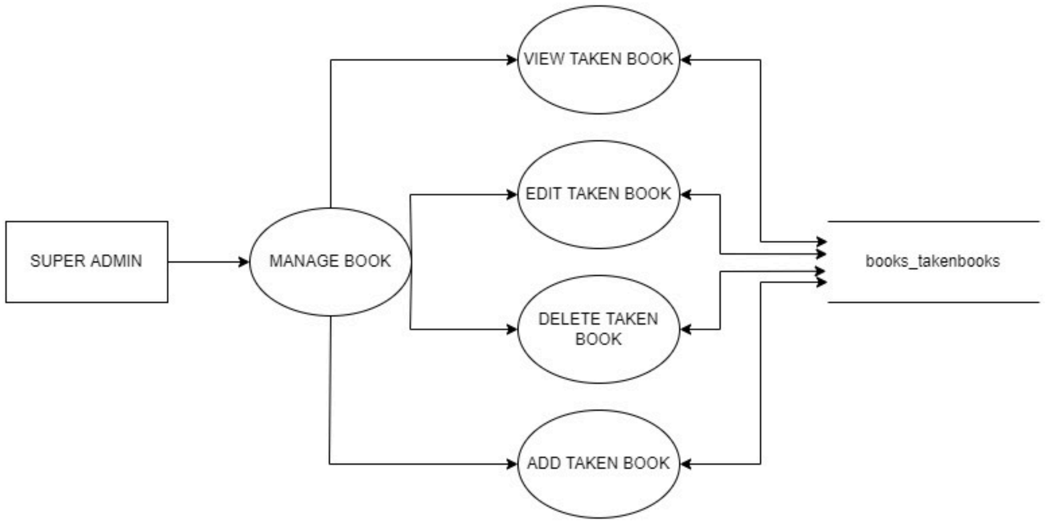
## DFD Level 2.1.5



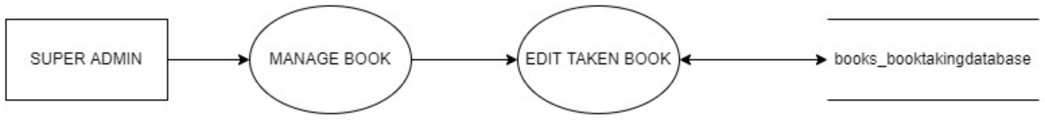
## DFD Level 2.1.6



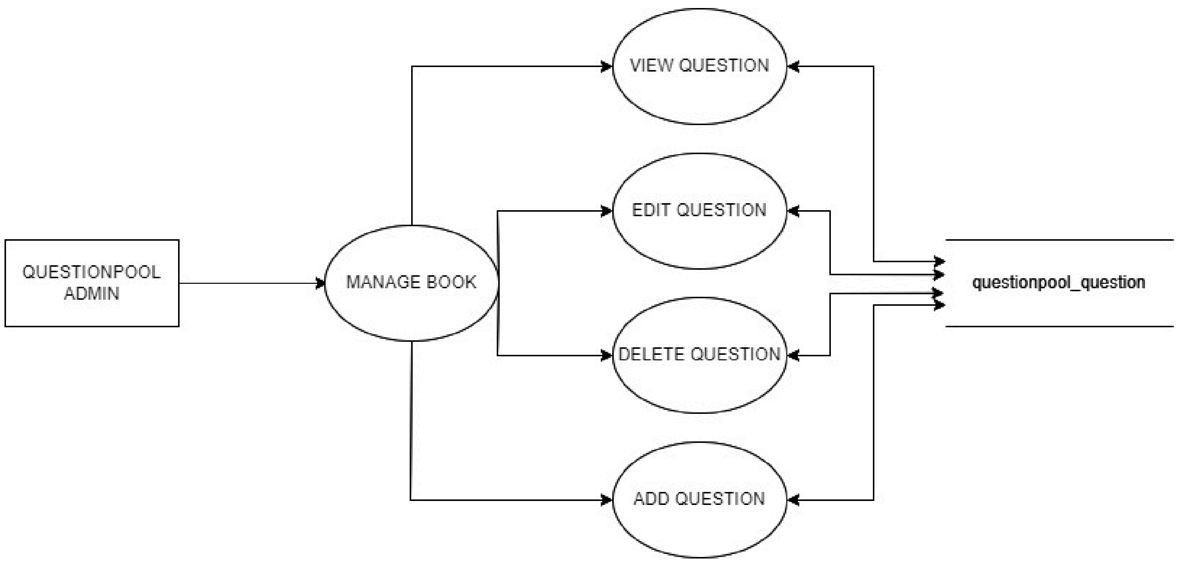
## DFD Level 2.1.7



## DFD Level 2.1.8



## DFD Level 2.2.1



## 4.2 Architectural design

The software architecture of a computing system is the structure of the system, which comprise application components, the externally visible properties of those components, and the relationships between them. The term also refers to documentation of a system's software architecture. Documenting software architecture facilitates communication between stakeholders, documents early decisions about high-level design, and allows the reuse of design components and patterns between projects. Architecture is commonly defined in terms of components and connectors. Components are identified and assigned responsibilities that client components interact with through "contracted" interfaces. Component interconnections specify communication, control mechanisms, and support all component interactions needed to accomplish system behaviour.

### 4.2.1 Database design

Database design is one of the most important parts of the system design phase. In a database environment, common data are available and are used by several users. Instead of each program managing its data, authorized users share data across the application with the database software managing the data as an entity. The primary objective of a database design is fast response time to enquiries, more information at low cost, control of redundancy, clarity and ease of use, date and program independence, accuracy and integrity of the system, fast recovery, and availability of powerful end-user languages. The theme behind a database is to handle information as an integrated whole thus the main objective is to make the information access easy, quick, inexpensive, and flexible for the users. The data directory specifies the major element in the system, and care should be taken while designing, to avoid unnecessary duplication of data. The entire package depends on how the data are maintained in the system. Several tables are maintained in the system to store data that are required for the processing of various data as well as storing intermediate or final processed results. Database design aims at handling large volumes of information, involving the definitions for the structure of storage and provisions for the manipulation of information, providing safety of information despite system crashes due to unauthorized access. Some conditions are satisfied in the database design stage.

- Control redundancy
- Ease of use
- Data independency
- Accuracy and integrity
- Recovery from failures
- Security and privacy
- Performance

### 4.2.2 Database Tables

Database name: amallibrary

1. Table name: auth_user

| **Field Name** | **Data types** | **Constraints** |
|---|---|---|
| id | integer | Primary Key |
| username | character varying | Not Null |
| email | character varying | Not Null |
| password | character varying | Not Null |
| last_login | timestamp with timezone | Not Null |
| first_name | character varying | Not Null |
| last_name | character varying | Not Null |
| is_staff | boolean | Not Null |

| | | |
|---|---|---|
| is_active | boolean | Not Null |
| date_joined | timestamp with timezone | Not Null |
| is_superuser | boolean | Not Null |

2. Table Name: authentication_quesadmin

| **Field Name** | **Data types** | **Constraints** |
|---|---|---|
| id | bigint | Primary Key |
| quesadmin_Key | character varying | Not Null |
| quesadmin_department_id | bigint | Foreign Key |
| quesadmin_username | character varying | |

3. Table Name: authentication_students

| **Field Name** | **Data types** | **Constraints** |
|---|---|---|
| id | bigint | Primary Key, Not Null |
| student_Key | character varying | Not Null |
| student_name | character varying | Not Null |
| student_gender | character varying | Not Null |
| student_dob | character varying | Not Null |
| student_address | character varying | Not Null |
| student_phone | character varying | Not Null |
| student_admissionyear | character varying | Not Null |
| student_admissionumber | character varying | Not Null |
| student_course_id | bigint | Not Null |
| student_department_id | bigint | Not Null |
| student_username__id | integer | Not Null |

4. Table Name: authentication_staffs

| **Field Name** | **Data types** | **Constraints** |
|---|---|---|
| id | bigint | Primary Key |
| staff Key | character varying | Not Null |

|  |  |  |
|---|---|---|
| staff_name | character varying | Not Null |
| staff gender | character varying | Not Null |
| staff_dob | character varying | Not Null |
| staff address | character varying | Not Null |
| staff_phone | character varying | Not Null |
| staff_yearofjoin | character varying | Not Null |
| staff department_id | bigint | Not Null |
| staff_post_id | bigint | Not Null |
| staff_username_id | integer | Not Null |

5. Table Name: authentication_department

| **Field Name** | **Data types** | **Constraints** |
|---|---|---|
| id | bigint | Primary Key |
| department | character varying | Not Null |

6. Table Name: authentication_posts

| **Field Name** | **Data types** | **Constraints** |
|---|---|---|
| id | bigint | Primary Key |
| post | character varying | Not Null |
| department_id | bigint | Not Null |

7. Table Name: authentication_courses

| **Field Name** | **Data types** | **Constraints** |
|---|---|---|
| id | bigint | Primary Key |
| course | character varying | Not Null |
| department_id | bigint | Not Null |

8. Table Name: authentication_subjects

| **Field Name** | **Data types** | **Constraints** |
|---|---|---|
| id | bigint | Primary Key |

| | | |
|---|---|---|
| subject | character varying | Not Null |
| department_id | bigint | Not Null |

9. Table Name: blog_blogpost

| Field Name | Data types | Constraints |
|---|---|---|
| id | bigint | Primary Key |
| title | character varying | Not Null |
| pub_date | timestamp with time zone | Not Null |
| body | text | Not Null |
| image | character varying | Not Null |

10. Table Name: blog_blogpostcomment

| Field Name | Data types | Constraints |
|---|---|---|
| id | bigint | Primary Key |
| comment | text | Not Null |
| author | character varying | Not Null |
| rating | integer | Not Null |
| post_id | bigint | Not Null |

11. Table Name: blog_feedback

| Field Name | Data types | Constraints |
|---|---|---|
| id | bigint | Primary Key |
| name | character varying | Not Null |
| email | character varying | Not Null |
| message | text | Not Null |

12. Table Name: blog_notice

| Field Name | Data types | Constraints |
|---|---|---|
| id | bigint | Primary Key |
| title | character varying | Not Null |
| body | character varying | Not Null |

13. Table Name: books_books

| **Field Name** | **Data types** | **Constraints** |
|---|---|---|
| id | bigint | Primary Key |
| name | character varying | Not Null |
| author | character varying | Not Null |
| pub_date | timestamp with timezone | Not Null |
| category | character varying | Not Null |
| description | text | Not Null |
| booknumber | integer | Not Null |
| image | character varying | Not Null |

14. Table Name: books_takenbooks

| **Field Name** | **Data types** | **Constraints** |
|---|---|---|
| id | bigint | Primary Key |
| date | timestamp with timezone | Not Null |
| book_id | bigint | Not Null |
| user_id | integer | Not Null |
| renewaldate | timestamp with timezone | Not Null |
| renewalstatus | boolean | Not Null |
| returndate | timestamp with timezone | Not Null |
| returnstatus | boolean | Not Null |

15. Table Name: books_takenbooksdatabase

| **Field Name** | **Data types** | **Constraints** |
|---|---|---|
| id | bigint | Primary Key |
| booknumber | integer | Not Null |

| | | |
|---|---|---|
| username | character varying | Not Null |
| takendate | timestamp with timezone | Not Null |
| returndate | timestamp with timezone | Not Null |

16. Table Name: questionpool_question

| **Field Name** | **Data types** | **Constraints** |
|---|---|---|
| question_num | integer | Primary Key |
| question_ques | text | Not Null |
| question_answer | text | Not Null |
| pub_date | timestamp with timezone | Not Null |
| question_department_id | bigint | Not Null |
| question_subject_id | bigint | Not Null |

CHAPTER 5

# IMPLEMENTATION

The implementation is an important phase. It involves user training, system testing and successful running of the developed proposed system. The user tests the developed systems and changes are made according to their needs.

The implementation phase of software development involves translating design specifications into source code, and debugging, documenting and unit testing the source code.

Implementation is the process of converting a new or revised system design into an operational one. Conversion means changing from one system to another. The objective is to put the system into operation while holding costs, risks, and personnel irritation to a minimum. It involves

- Creating computer compatible files.
- Training operational staff.
- Installing terminals and hardware.

Form Builder is a web-based application system, so it requires some space in the server. To implement the new system, it must be deployed in the localhost. It is also possible to run the system in the local network. First upbuild the website, then publish the website. After publishing the project, the files will be integrated into a package. These files and databases will be published on the online server. Then it is possible to access the functionalities just by using the URL. Only a web browser is required to operate. The project is developed in a framework, but the client system does not require any framework to be installed. This method also offers the greatest security.

## 5.1 SYSTEM DEVELOPMENT

The development phase of the software design consists of different tasks to be done sequentially for obtaining the desired results. The distinct phases are:

- Creating a system environment:

For the intended project to work on, we need to implement its required hardware and software requirements. This system is built using Visual studio code and is based on Windows Operating System. Memory and Hard disk should confirm according to the hardware mentioned.

- Creating graphical user interface:

A graphical user interface is created in Visual studio code for a user-friendly interface. It is intended for two purposes. The first is to create a user-friendly interface for the software. Having a good user interface makes it easier for the user to use and understand the different functionalities of the website. Secondly, the user interface hides the end-users from the complexities in the working of the software. So, the user is made unaware of how a task is performed when he chooses to perform it

- Creating the database:

A database is created, in which all the tables are defined which are required to do the different operations such as storage and retrieval of information. The database is designed in such a way it can handle the different database queries. Users and admins can retrieve required details from the system by clicking on the links and buttons.

.

CHAPTER 6

# SYSTEM TESTING

Testing is the process of examining the application to compare the actual behaviour with that of the excepted behaviour. The major goal of software testing is to demonstrate that faults are not present. To achieve this goal, the tester executes the program with the intent of offending errors. Though testing cannot show the absence of errors by not showing their presence it is considered that these are not present. The following method of testing is carried out to assure the correctness and reliability of Amal Library.

## 6.1Test Plan

A test plan is a systematic approach to testing a system. The plan typically contains a detailed understanding of what the eventual workflow will be. Normally, testing of any large system will be in two parts.

- The functional verification and validation against the requirement specification.
- Performance evaluation against the indicated requirements.

Testing activity is involved right from the beginning of the project. At the very first stage of testing, the goals and objectives are set. This simplifies the limits or borders of the testing process. Before testing, the tester should plan what kind of data he is giving for the test. Give data inputs as functional, boundary, stress, performance, usability values etc.

## 6.2Test Cases

The specific set of steps and data along with expected results for a particular test objective. A test case should only test one limited subset of a feature or functionality. Test case documents for each functionality/testing area will be written, reviewed, and maintained separately in Excel Sheet. In system testing, test data should cover the possible values of each parameter based on the requirements. Since testing every value is impractical, a few values should be chosen from each equivalence class. An equivalence class is a set of values that should all be treated the same. Ideally, test cases that check error conditions are written separately from the functional test cases and should have steps to verify the error messages and logs. Realistically, if error test cases are not yet written, it is OK for testers to check for error conditions when performing normal functional test cases. It should be clear which test data, if any, is expected to trigger errors.

**Unit Testing:**

| Test Cases | Test Conditions | Test Data | Execution Setup | Expected Result | Pass/Fail |
|---|---|---|---|---|---|
| 1 | Admin Registration | Username, Password and Other Details | Click on Registration Button | Redirects to Admin Login Page | Pass |
| 2 | Admin Login | Username and Password | Click on Login Button | Redirects to Admin Dashboard | Pass |
| 3 | User Login | Username and Password | Click on Login Button | Redirects to Home Page | Pass |

**System Testing:**

The entire system is tested as per the requirements. Black-box type testing that is based on overall requirements specifications, covers all combined parts of a system.

| Test Cases | Test Conditions | Test Case Description | Expected Result | Observed Result | Pass/Fail |
|---|---|---|---|---|---|
| 1 | Testing all the modules in the system | It tests all the modules in the system whether it is executing in the system or not | By testing all the modules, it is accepted by the system and should execute the result | It tests all the modules, and the system accepts all the modules and produces an expected result | Pass |

**Validation Testing:**

The objective of this testing is to tell the user about the validity and reliability of the system.

| Test Cases | Test Conditions | Test Case Description | Expected Result | Observed Result | Pass/Fail |
|---|---|---|---|---|---|
| 1 | Testing all the modules in the system | It tests all the modules based on the input from the user to the system and gets a valid output from the system | Valid input from the user to the system and getting a valid output from the system | Getting a valid and reliable output from all the modules in the system | Pass |

## 6.3 Unit testing

Unit testing is carried out in parallel to coding. The functionality of each of the individual modules we have developed is tested using test cases. Unit testing is done according to the test plans prepared for each module. The test plan prepared has covered all functional areas, actual input and expected outputs. Modules are tested using different test cases. Actual outputs are compared with expected outputs. The results are also recorded.

## 6.4 Integration Testing

The major concerns of integration testing are developing incremental strategies that will the complexity of entire actions among components as they are added to the system. Though each program works individually; they should work after linking them together. This is also referred to as interfacing. Data may be lost across the interface and one module can hurt another. Integration testing is a systematic technique for constructing program structure while at the same time, conducting a test to uncover errors associated with the interface. In the testing, the programs are constructed and tested in small segments.

## 6.5 Validation Testing

Validation testing provides the final assurance that software meets all functions, behavioural and performance requirements. Here victims submit their applications and then enter all details for assets.

## 6.6 Output Testing

After performing the validation testing, the next step is the output testing of the enhanced system. No system could be useful if it does not produce the required output in the required format. The outputs generated or displayed by the "ADS" are defined during the system analysis. During output testing, the developer required some new report layout changes. These changes were done and tested before the final delivery of the system.

## 6.7 Performance Testing

Performance Testing is designed to test the runtime performance of the integrated system. CDAS can process a large amount of data with minimum memory and time is tested here.

## 6.8 System Security Measures

System security is a branch of technology known as information security as applied to computers and networks. The objective of system security includes fifty-six protection of information and property from theft, corruption, or natural disaster while allowing the information and property to remain accessible and productive to its intended users. The terms system security means the collective processes and mechanisms by which sensitive and valuable information and services are protected from publication, tampering or collapse by unauthorized activities or untrustworthy individuals and unplanned events, respectively. The technologies of system security are based on logic. As security is not necessarily the primary goal of most computer applications, designing a program with security in mind often imposes restrictions on that program's behaviour.

### 6.8.1Checks and Controls

Some of the important checks and controls used for system security are given below, they are:

1. **Confidentiality**: A security measure that protects against the disclosure of information to parties other than the intended users that are by no means the only way of ensuring.

2. **Integrity**: A measure intended to allow the receiver to determine that the information which it receives has not been altered in transit or by other than the originator of the information. Integrity schemes often use some of the same underlying technologies as confidentiality schemes, but they usually involve adding additional information to communication to form the basis of an algorithmic check rather than encoding all the communication.
3. **Authentication**: A measure designed to establish the validity of a transmission, message, or originator. Allows a receiver to have confidence that information it receives originated from a specific known source.
4. **Authorization**: The process of determining that a requester is allowed to receive a service or perform an operation. Access control is an example of authorization.
5. **Availability**: Assuring information and communications services will be ready for use when expected. The information must be kept available to authorized persons when they need it.
6. **Non-repudiation**: A measure intended to prevent the later denial that an action happened, or communication that took place etc. In communication terms, this often involves the interchange of authentication information combined with some form of the provable time stamp.

### 6.8.2 Database Security

Data security is the practice of keeping data protected from corruption and unauthorized access. The focus behind data security is to ensure privacy while protecting personal or corporate data. Data is the raw form of information stored as columns and rows in our databases, network servers and personal computers. This may be a wide range of information from personal files and intellectual property to market analytics and details intended to be top secret. Data could be anything of interest that can be read or otherwise interpreted in human form.

Encryption has become a critical security feature for thriving networks and active home users alike. This security mechanism uses mathematical schemes and algorithms to scramble data into unreadable text. It can only be decoded or decrypted by the party that possesses the associated key.

Full-disk encryption offers some of the best protection available. This technology enables you to encrypt every piece of data on a disk or hard disk drive. Full disk encryption is even more powerful when hardware solutions are used in conjunction with software components. This combination is often referred to as end-based or endpoint full disk.

Authentication is another part of data security that we encounter with everyday computer usage. Just think about when you log into your email or blog account. That single sign-on process is a form of authentication that allows you to log into applications, files, folders and even an entire computer system. Once logged in, you have various given privileges until logging out. Some systems will cancel a session if your machine has been idle for a certain amount of time, requiring that you prove authentication once again to reenter. The single sign-on scheme is also implemented into strong user authentication systems. However, it requires individuals to log in using multiple factors of authentication.

Data security would not be complete without a solution to back up your critical information. Though it may appear secure while confined away in a machine, there is always a chance that your data can be compromised. You could suddenly be hit with a malware infection where a virus destroys all your files.

Someone could enter your computer and thieve data by sliding through a security hole in the operating system. It was an inside job that caused your business to lose those sensitive reports. If all else fails, a reliable backup solution will allow you to restore your data instead of starting completely from scratch.

### 6.8.3 Creation of User Profiles & Access Rights

User security lets your application use security rules to determine what it displays. It has two elements:

**Authentication**: Authentication is another part of user security that we encounter with everyday computer usage. Ensures that a valid user is logged-in, is based on an ID and password provided by the user. ColdFusion (or, in some cases, if you use web server authentication, the webserver) maintains the user ID information while the user is logged in.

**Authorization**: Ensures that the logged-in user is allowed to use a page or perform an operation. Authorization is typically based on one or more roles (sometimes called groups) to which the user belongs. For example, when the user signs as a buyer, he/she can only see the property details for selling and searching for properties. But they cannot modify the property details. You can also use the user ID for authorization.

CHAPTER 7

# FUTURE ENHANCEMENT

We have some future visions for the enhancement of the web app, this is just a prototype. Now, this system is only available as a browser app. In future, we will be developing the system for android and apple applications. We will be implementing more features such as a block book, automated question paper generator, Search book, and book review soon. The design of this software is in such a way that the addition of any new module if necessary is possible without affecting the integrity of the present system. For further enhancement, we can develop this web application to deliver to all teams. The design of the system has done foreseeing future enhancements so it will not affect the base system inversely.

CHAPTER 8

# CONCLUSION

AMAL LIBRARY, the web app is very essential for college educational growth in the current digital scenario. It allows users to See Library updating, get the Question pool, and view the Availability of Books In the library. The students can register for an event at anytime from anywhere. By using this system students can save a lot of time and effort. The student can easily get the information from anywhere. The program for caring out the various activities has been run and evaluated successfully to ensure that the software development will meet the needs satisfactory. We could conclude that this website has met all its objectives of being user friendly.

# BIBLIOGRAPHY

**Websites:**

- https://medium.com
- https://stackoverflow.com
- https://www.w3schools.com
- https://www.djangoproject.com
- https://www.youtube.com
- https://realpython.com
- https://tailwindcss.com

## APPENDIX