

**A PROJECT REPORT ON**

# **SUSTAINABLE FINANCE**

**Dissertation Submitted**

In partial fulfillment of the requirement for the award of the degree  
of

**BACHELOR OF COMPUTER APPLICATIONS**

**KERALA UNIVERSITY**

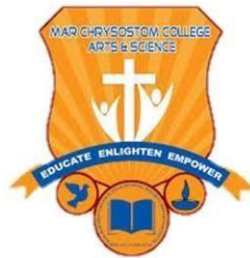
**By:**

**AJIN VARUGHESE (Reg.No. - 33223157007)**

**FAHIM SHAMIM AZIZ (Reg.No. - 33223157017)**

**MIDHUN SUNIL (Reg.No. - 33223157022)**

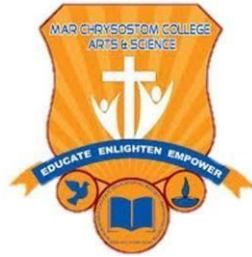
**MUHAMMED ASLAM (Reg.No. - 33223157025)**



**MAR CHRYSOSTOM COLLEGE PARANTHAL - ADOOR**

**July 2025**

**MAR CHRYSOSTOM COLLEGE  
PARANTHAL, ADOOR**



**CERTIFICATE**

This is to Certify that this project report entitled “**SUSTAINABLE FINANCE**” is a Bonafide record of the project work done by,

<b>AJIN VARUGHESE</b>	<b>(33223157007)</b>
<b>FAHIM SHAMIM AZIZ</b>	<b>(33223157017)</b>
<b>MIDHUN SUNIL</b>	<b>(33223157022)</b>
<b>MUHAMMED ASLAM</b>	<b>(33223157025)</b>

Under our supervision and guidance, towards *partial fulfilment of the requirements for the award of the **Degree of BACHELOR OF COMPUTER APPLICATION*** under **University of Kerala** during the year 2023-2026

**Ms. Saramma John**

**Project Guide**

**Ms. Saramma John**

**Head of Department**

**External Examiner**

## **DECLARATION**

We, **AJIN VARUGHESE (REG. No. 33223157007), FAHIM SHAMIM AZIZ (REG. No. 33223157017), MIDHUN SUNIL (REG. No. 33223157022), MUHAMMED ASLAM (REG. No. 33223157025)** declares that the project on “**AUTOMATED GRADING SYSTEM WITH COLLEGE HANDLING**” is the result of original work done by us and to the best of our knowledge, a similar work has not been submitted earlier to the **University of Kerala** or any other Institution, for fulfillment of the requirement of a course of study.

This project report is submitted on partial fulfillment of the requirement for the degree of Bachelor of Computer Applications of University of Kerala

**AJIN VARUGEHSE  
FAHIM SHAMIM AZIZ  
MIDHUN SUNIL  
MUHAMMED ASLAM**

## **ACKNOWLEDGEMENT**

We with our utmost courteousness and gratitude submit our implacable praises to the **Omnipotent Almighty** for having bestowed his grace and blessings over us to accomplish this task of project dissertation.

We extend our sincere thanks to **Prof. ITTY VARGHESE** (Principal, Mar Chrysostom College), **Ms. SARAMMA JOHN** (HOD- BCA Department, Mar Chrysostom College), **Ms. SARAMMA JOHN** (Internal Guide, Mar Chrysostom College) for their valuable advice, inspiration, whole hearted support and guidance at every aspect of this project.

We would like to express our sincere thanks to **SOFTZANE SOLUTIONS**, officials for providing us a chance to do project in their esteemed organization.

**AJIN VARUGHESE**  
**FAHIM SHAMIM AZIZ**  
**MIDHUN SUNIL**  
**MUHAMMED ASLAM**

# CONTENTS

	Page No
<b>1. INTRODUCTION</b>	<b>1</b>
1.1. ABOUT THE PROJECT	2
1.2. PROJECT OVERVIEW	3
1.3. ORGANIZATION PROFILE	5
<b>2. SYSTEM STUDY AND ANALYSIS</b>	<b>7</b>
2.1. REQUIREMENT ANALYSIS	9
2.2. EXISTING SYSTEM	10
2.3. PROPOSED SYSTEM	10
2.4. FEASIBILITY STUDY	11
2.4.1. OPERATIONAL FEASIBILITY	11
2.4.2. TECHNICAL FEASIBILITY	12
2.4.3. ECONOMICAL FEASIBILITY	12
<b>3. REQUIREMENT SPECIFICATION</b>	<b>13</b>
3.1. HARDWARE SPECIFICATION	14
3.2. SOFTWARE REQUIREMENTS	15
3.3. SOFTWARE DESCRIPTION	15
3.4. FUNCTIONAL REQUIREMENTS	19
<b>4. SYSTEM DESIGN</b>	<b>21</b>
4.1. DATABASE DESIGN	23
4.1.1. DATA FLOW DIAGRAM	36
4.1.2. USE CASE DIAGRAM	41
4.1.3. FLOW CHART	45
4.2. PROCEDURAL DESIGN	46
4.3. INTERFACE DESIGN	46
4.4. INPUT DESIGN	46
4.5. OUTPUT DESIGN	47
<b>5. SYSTEM CODING</b>	<b>48</b>
<b>6. SYSTEM TESTING</b>	<b>60</b>
6.1. TESTING	61
6.2. UNIT TESTING	61
6.3. INTEGRATION TESTING	62
6.4. SYSTEM TESTING	63
6.5. USER ACCEPTANCE TESTING	63
<b>7. SYSTEM IMPLEMENTATION AND MAINTENANCE</b>	<b>65</b>
7.1. SYSTEM IMPLEMENTATION	66
7.2. SYSTEM MAINTENANCE	67
7.3. FUTURE ENHANCEMENT	67
<b>8. CONCLUSION</b>	<b>69</b>
<b>9. APPENDIX</b>	<b>71</b>
9.1. GANTT CHART	72
9.2. MEETING MINUTES	73
9.3. SCREEN LAYOUTS AND REPORTS	80
<b>10. BIBLIOGRAPHY</b>	<b>83</b>

**INTRODUCTION**

## 1. INTRODUCTION

### 1.1 ABOUT THE PROJECT

The **Sustainable Finance** platform is developed to evaluate loan applications based on their alignment with environmental sustainability principles. It dynamically analyzes responses to customized questionnaires using predefined eligibility rules and taxonomy-aligned criteria. This system focuses on supporting green loan schemes such as **renewable energy projects, eco-friendly housing, and low-carbon businesses** by integrating automation, sustainability logic, and real-time evaluation.

The platform intelligently assesses questionnaire responses using rule-based logic and environmental contribution tags derived from the **EU Taxonomy for Sustainable Activities**. It ensures consistent and objective eligibility screening, significantly reducing the manual effort required by bank officers and administrators. The system also provides real-time feedback to applicants on their sustainability compliance and basic loan eligibility, ensuring transparency and guiding them toward greener financial decisions.

Additionally, the platform is designed to be adaptive — allowing non-technical administrators to update eligibility rules, add or modify questions, and categorize them based on loan types. By aligning financial decisions with sustainability goals, the system promotes environmentally responsible lending and helps institutions monitor their green loan portfolios.

## 1.2 PROJECT OVERVIEW

### Upload and Submission:

**Function:**

Allows students or users to upload their essays, QP, OMR for grading.

**Features:**

Supports various file formats (e.g., DOCX, PDF, TXT).

Allows students to submit multiple documents for grading.

Provides a user-friendly interface for document submission.

### Natural Language Processing (NLP) for Grammar and Structure Analysis:

**Function:**

Analyzes sentence structure, grammar, and syntax of the written paper.

**Features:**

Sentence Parsing: Breaks down the document into meaningful components.

Error Detection: Identifies and flags grammatical errors (subject-verb agreement, punctuation, etc.).

Structure Analysis: Evaluates clarity, coherence, and readability.

Suggestions: Provides grammar and style improvement tips.

### Text Similarity Algorithms for Plagiarism Detection:

**Function:**

Detects similarities and possible plagiarism within the document.

**Features:**

Cosine Similarity & Jaccard Index: Compare the submitted paper with a database of other information, articles, and online content.

Plagiarism Highlights: Marks copied or paraphrased sections.

Similarity Report: Shows the percentage of similarity with external content.

Rewriting Suggestions: Provides recommendations to improve originality.

### Machine Learning Model for Grading:

**Function:**

Automatically grades the document based on predefined criteria.

**Features:**

Trained Model: Uses a dataset of papers graded by human evaluators.

Assessment Metrics: Evaluates relevance, coherence, argumentation, and content organization.

Continuous Learning: Improves accuracy over time.

Rubric-Based Scoring: Assigns a grade based on written answer quality.



**Feedback Generation:****Function:**

Generates automated feedback for students.

**Features:**

Detailed Feedback: Covers grammar, structure, originality, and overall quality.

Improvement Suggestions: Highlights weak areas like argument strength and sentence clarity.

Personalized Reports: Custom feedback based on document analysis.

Actionable Tips: Provides strategies for writing better answers.

### 1.3 ORGANIZATION PROFILE

Softzane Solutions pvt. ltd is a 7-year-old IT company incorporated on 13-September-2016, having its registered office located at 2nd floor Christuraj Shopping Complex Anchal Road Ayoor, Kollam, Kerala. A company managed by highly professional and experienced team with a group of truly dedicated, hardworking, and sincere young professionals.

The major activity of Softzane Solutions offers a complex spectrum of custom software development services. Since entering the IT industry, we have gained exceptional experience in software development on Smartphone technologies as well as web application development, they make sure that their clients get leading innovative solutions that are both cost-efficient and dependable. They have an effective solution providing service team and an innovative management to organize the client solutions. Since inception, this institute has constantly worked towards raising the technical skills and standards in pursuit to offer the best technical infrastructure to the budding engineers. We have grown technically sound with state-of-the-art technical labs, equipment's and best infrastructure for academic and research activities.

Today's managers and engineers need applicable knowledge and skills to navigate their organization through an ever-changing dynamic environment. We aim to develop the creativity of each student by providing them a motivating and stimulating environment enabling the students to look forwarding to a bright future with meaningful existence. We develop their skills so that they can contribute to the rapidly changing technology requirements making them ready for any competition in the industry. This is achieved through long years of perseverance, persistence and perspiration which make us a cynosure of other organization of similar nature. This has led to our acceptance as an excellent institution providing internship programs in campus training and skill up-gradation training with a view to enhance their employability in electronics as well as information technology. We offer various skill up-gradation programs to students during their semester breaks or as per the time allotted to various institutions at their own campus enabling each student to become job ready.

Softzane Solutions is a dynamic and innovative technology company specializing in providing software solutions tailored to meet the unique needs of its clients. With a focus on cutting-edge technologies and a commitment to customer satisfaction, Softzane Solutions aims to deliver high-quality products and services that drive business growth and efficiency.

**Mission Statement:**

Our mission at Softzane Solutions is to empower businesses with innovative software solutions that streamline operations, enhance productivity, and drive success. We are dedicated to delivering exceptional value to our clients through continuous innovation, collaboration, and excellence in execution.

**Core Values:**

**Innovation:** We embrace creativity and strive for continuous improvement in everything we do.

**Integrity:** We conduct business with honesty, transparency, and ethical standards.

**Customer Centricity:** We prioritize understanding and exceeding our clients' expectations to build long-term partnerships.

**Collaboration:** We believe in the power of teamwork and collaboration to achieve shared goals.

**Excellence:** We are committed to delivering high-quality solutions and services that exceed industry standards.

**Services:**

Softzane Solutions offers a comprehensive range of services, including:

Custom Software Development

- ❖ Internship Programs
- ❖ Web and Mobile Application Development
- ❖ Enterprise Software Solutions
- ❖ Cloud Computing Services
- ❖ Data Analytics and Business Intelligence
- ❖ IT Consulting and Advisory Services
- ❖ Software Maintenance and Support

## **SYSTEM STUDY AND ANALYSIS**

## 2. SYSTEM ANALYSIS

System analysis is a detailed study of various operations performed by a system and their relationship within and outside of the system. Here the key question is: What must be done to solve the problem? One aspect of analysis is defining the boundaries of a system and determines whether or not the candidate system should consider other related systems. Analysis begins when a user or manager begins a study of the program using the existing system.

During analysis, data is collected on the various files, decision points and transactions handled by the present system. The commonly used tools in system analysis are dataflow diagram, interviews, onsite observation etc. System analysis is application of the system approach to the problem-solving using computers. The ingredients are system elements, process and technology. This means that to do system work, one is to understand the system concepts and how the organizations operate as a system and the design appropriate computer-based system that will meet the organizations requirements. It is actually customized approach to the use of computer problem solving.

Analysis can be defined as the separation of a substance into parts for study an interpretation, detailed examination. System development revolves around a lifecycle that begins with the recognition of user needs. The critical phase of managing system project is planning. To launch a system investigation, we need a master plan detailing the steps taken, the people to be questioned and outcome expected.

**System analysis can be categorized into four parts:**

- System planning and initial investigation.
- Information Gathering.
- Applying analyzing tools for structured analysis.
- Feasibility study.
- Cost/Benefit analysis.

System study or system analysis is the first among the four-life cycle phase of a system. System begins when a user or manager requests a studying of a program in either an existing or a project one. It involves studying the base of the organizations currently operating, retrieving and processing data to produce information with the goal of determining how to make it work better. System analysis itself breaks down into stages-Preliminary and Detailed. During

preliminary analysis the analyst and the user list the objectives of the system. To arrive at a preliminary report, the analyst interviews key personnel in the organization and schedule meetings with the users and the management.

Thus, the objective of analysis phase of the system—analysis and design exercise is the establishment of the requirement for the system to be acquired, developed and installed. Fact-finding or gathering is essential to any analysis of requirements. In brief analysis of the system helps an analyst to make a clear view of the existing system and thereby can give suggestions for the improvement of the new system. Information's about the organization's policies, goals, objectives, and structure explains the kind of environment that promotes the introduction of computer-based system. It is necessary that the analyst must be familiar with the objectives, activities and the functions of the organizations in which the computer system is to be implemented.

## **2.1 REQUIREMENT ANALYSIS**

Requirement Analysis, also known as Requirement Engineering, is the process of defining user expectations for a new. software being built or modified. In software engineering, it is sometimes referred to loosely by names such as requirements gathering or requirements capturing.

### **REQUIREMENT ANALYSIS PROCEDURE**

1. Document analysis. User manuals and process documents about the current system can become helpful in defining software requirements.
2. Interview
3. Observation
4. Workshop
5. Brainstorming
6. Prototyping
7. Collect and Understand Requirements
8. Define Requirements

## 2.2 EXISTING SYSTEM

The current loan evaluation systems used by banks and financial institutions largely rely on manual processes, static eligibility checklists, and paper-based forms. Loan officers assess an applicant's eligibility based on traditional financial parameters, often without structured mechanisms for evaluating sustainability factors. There is minimal use of automation or intelligence in screening loan applications against environmental criteria such as carbon impact or alignment with green project categories. Moreover, there is no centralized system to dynamically manage eligibility rules, categorize loans by type, or generate impact reports. This leads to inconsistencies, delayed decisions, limited transparency, and difficulty in tracking sustainability performance across loan portfolios.

## 2.3 PROPOSED SYSTEM

The proposed Sustainable Finance system transforms the traditional loan evaluation process by introducing a smart, automated, and sustainability-aware platform. It utilizes structured questionnaires, a dynamic rule engine, and EU Taxonomy-based activity mapping to assess whether a loan application supports sustainable development goals. Administrators can easily define and update eligibility criteria, categorize questions based on loan types, and match responses to predefined taxonomy activities. The system automates loan pre-screening, validates environmental compliance, and provides real-time feedback to users. It also supports detailed analytics and reporting on sustainability impact, helping banks monitor green loan portfolios effectively.

## ADVANTAGES

- Automation of Loan Eligibility Evaluation
- Alignment with EU Taxonomy for Sustainability
- Dynamic Questionnaires Based on Loan Categories
- Real-Time Eligibility Feedback for Applicants
- Admin Interface for Rule & Question Management
- Carbon Impact & Green Metrics Tracking
- Transparent, Objective & Scalable Assessment System
- Secure Role-Based Access for Admins, Officers, and Applicants

## **2.4 FEASIBILITY STUDY**

An initial investigation culminates in a proposal that determines whether an ultimate system is feasible. When a proposed system is made and approved it initiates a feasibility study. The purpose of the feasibility study is to identify various candidate systems and evaluate whether they are feasible by considering technical, economical, and operational feasibility and to recommend the best candidate system. The feasibility of such a program is listed in a simulated environment. Once all features are working properly in a simulated environment, we can implement them in a real platform.

A feasibility study aims to objectively and rationally uncover the strengths and weaknesses of an existing business or proposed venture, opportunities and threats present in the environment, the resources required to carry through, and ultimately the prospect for the successes. In the simplest terms the two criteria to judge feasibility are cost required and value to be attained. During product engineering, we considered the following types of feasibility. The study is done in six phases.

- Operational feasibility
- Technical feasibility
- Economic feasibility

### **2.4.1 OPERATIONAL FEASIBILITY**

The system is designed with intuitive interfaces and a role-based access model, ensuring ease of use for administrators, loan applicants, and bank officers. Users can seamlessly interact with the platform to create questionnaires, respond to questions, and evaluate loan eligibility. The admin panel allows non-technical staff to manage rules and taxonomy without coding. Automated feedback and analytics reduce manual workload and increase decision-making speed. Because the system follows familiar patterns like form-based input, rule-based validation, and dashboard reporting, the learning curve is minimal, and integration with existing loan processes is smooth.



### **2.4.2 TECHNICAL FEASIBILITY**

The proposed Sustainable Finance system is technically feasible due to the availability of mature, reliable, and well-supported development frameworks and technologies. The backend is developed using Java Spring Boot, ensuring scalability, modularity, and security. The frontend leverages ReactJS, providing a responsive and user-friendly interface. Database management is handled using MySQL, while Spring Data JPA enables efficient ORM-based data handling. The system integrates sustainability taxonomy data, rule engines, and real-time validations — all of which can be implemented using existing open-source libraries and tools. The project is deployable on cloud platforms (e.g., AWS, Azure), making it future-proof and easily scalable.

### **2.4.3 ECONOMICAL FEASIBILITY**

Economically, the system provides a high return on investment (ROI) by significantly reducing manual evaluation costs, minimizing paper-based workflows, and improving turnaround time for loan processing. It reduces the need for additional staff or consultants to perform sustainability assessments. Since it uses open-source technologies, licensing costs are minimal. The long-term cost savings from automation, better decision accuracy, and compliance with green financing norms make the project financially viable. Moreover, institutions adopting this system may qualify for green finance incentives or ESG ratings, further enhancing its value.

**REQUIREMENT SPECIFICATION**

### 3. SOFTWARE SPECIFICATION

A software requirements specification (SRS) is a detailed description of a software system to be developed with its functional and non-functional requirements. The SRS is developed based on the agreement between customer and contractors. It may include the use cases of how the user is going to interact with software system. The software requirement specification document is consistent of all necessary requirements required for project development.

#### 3.1 HARDWARE SPECIFICATION

The hardware for the system is selected considering the factors such as CPU processing speed, memory access speed, peripheral channel speed, printer speed; seek time & relational delay of hard disk and communication speed etc.

The hardware specifications are as follows:

Processor : Intel Pentium Quad Core

Operating Speed : 2.66GHZ

#### **Memory**

RAM : 2GB

#### **Disk Storage**

Hard Disk : 500GB

Data Backup Device : Pen drive

#### **Peripherals**

Keyboards : 106 keys

Mouse : Standard

Monitors : Proper Resolution

Printers : Laser/Inkjet

### 3.2 SOFTWARE REQUIREMENTS

The software requirements for the Sustainable Finance system have been carefully chosen to ensure scalability, security, platform independence, and developer productivity. Modern and widely adopted technologies have been selected to facilitate long-term maintainability, performance, and integration with cloud services.

Component	Specification
-----------	---------------

Operating System	:	Windows 10 / 11, Linux (Ubuntu recommended)
Frontend	:	ReactJS, HTML5, CSS3, JavaScript
Backend	:	Java 17+, Spring Boot (REST API)
Database	:	MySQL
ORM	:	Spring Data JPA with Hibernate
Authentication	:	Spring Security with JWT
IDE / Editor	:	IntelliJ IDEA / Visual Studio Code
Build Tool	:	Maven
Browser	:	Google Chrome, Mozilla Firefox, Edge
API Testing Tool	:	Postman
Version Control	:	Git with GitHub

### 3.3 SOFTWARE DESCRIPTION

#### Windows 10/11

An Operating System may be viewed as an organized collection of software extensions of hardware, consisting of control routines for operating a computer and for providing an environment for execution of programs. Programs rely on facilities provided by the operating system to gain access to computer-system resources such as files and input/output devices, i.e. the operating system acts as interface between users and the hardware of a computer system.

#### Windows 10

It is a major release of Microsoft's Windows NT operating system. It is the direct successor to Windows 8.1, which was released nearly two years earlier. It was released to manufacturing on July 15, 2015, and later to retail on July 29, 2015. Windows 10 is the final version of Windows that supports 32-bit processors (IA-32 and ARMv7-based) and devices with BIOS firmware.

Its successor, Windows 11, requires a device that uses UEFI firmware and a 64-bit processor in any supported architecture (x86-64 for x86 and ARMv8 for ARM). Windows 10 makes its user experience and functionality more consistent between different classes of device and addresses many shortcomings of the user interface introduced in Windows 8. Windows 10 Mobile, the successor to Windows Phone 8.1, shared some user interface elements and apps with its PC counterpart.

Windows 10 supports universal apps, an expansion of the Metro-style first introduced in Windows 8. Universal apps can be designed to run across multiple Microsoft product families with nearly identical code—including PCs, tablets, smartphones, embedded systems, Xbox One, Surface Hub and Mixed Reality. The Windows user interface was revised to handle transitions between a mouse-oriented interface and a touchscreen-optimized interface based on available input devices—particularly on 2-in-1 PCs. Both interfaces include an updated Start menu which incorporates elements of Windows 7's traditional Start menu with the tiles of Windows 8. Windows 10 also introduced the Microsoft Edge web browser, a virtual desktop system, a window and desktop management feature called Task View, support for fingerprint and face recognition login, new security features for enterprise environments, and DirectX 12.

### **Overview of JAVA SPRINGBOOT**

The Spring Boot is a powerful, open-source, Java-based framework designed to simplify and accelerate the development of standalone, production-grade web applications. It is part of the larger Spring ecosystem and was created by Pivotal Software to reduce the complexity of configuration and deployment in Java enterprise development. Spring Boot follows a microservice-friendly architecture and allows developers to focus on writing business logic without having to worry about boilerplate code or complex setup.

Spring Boot is built on top of the traditional Spring Framework, offering features like auto-configuration, embedded servers (Tomcat/Jetty), and seamless integration with technologies like Spring Data JPA, Spring Security, and RESTful APIs. It supports the MVC (Model-View-Controller) architecture and encourages clean separation between data, business logic, and presentation layers. With support for dependency injection, robust security, easy database connectivity, and cloud-native capabilities, Spring Boot is widely used in enterprise-grade applications and is ideal for building scalable, modular, and maintainable web systems.

**Key Features of Spring Boot:**

- **Auto-Configuration:** Automatically configures application components based on included dependencies, reducing manual setup.
- **Embedded Web Servers:** Comes with built-in Tomcat/Jetty/Undertow servers for running applications independently—no need for external server deployment.
- **Spring MVC Support:** Allows the development of REST APIs and web services using the Model-View-Controller architecture.
- **Spring Data JPA:** Simplifies database operations using object-relational mapping (ORM) and repository abstractions.
- **Security Integration:** Offers seamless integration with Spring Security for implementing authentication and authorization (e.g., with JWT tokens).
- **Dependency Injection (DI):** Encourages loosely coupled code and improved testability through inversion of control.
- **Actuator:** Provides production-ready features like health checks, metrics, and monitoring endpoints.
- **Custom Configuration:** Supports flexible configurations using .properties or .yaml files and @Value or @ConfigurationProperties annotations.
- **Microservice Ready:** Supports cloud deployment, service discovery, and scalability—ideal for distributed systems.

**MySQL**

MySQL is a widely-used, open-source relational database management system (RDBMS) that supports a broad range of applications from small to large-scale web applications. Unlike SQLite, MySQL operates with a client-server architecture, where the database is hosted on a server and accessed via a network connection. MySQL supports a full implementation of SQL, providing a robust, scalable system with extensive features such as stored procedures, triggers, and user-defined functions. It is ACID-compliant, ensuring reliable transactions, and it offers excellent performance for high-concurrency write operations and complex query processing. MySQL is known for its ability to handle high-traffic websites and large volumes of data with ease. While MySQL is typically more complex to set up and maintain than SQLite, it is highly suited for large-scale, high-traffic applications and systems that require advanced database management capabilities.

## **ReactJS**

JavaScript ReactJS is a JavaScript library for building user interfaces, primarily for single-page applications where you need a fast and dynamic response to user interactions. ReactJS simplifies the process of creating interactive UIs by allowing developers to build reusable UI components. Unlike traditional JavaScript, React uses a declarative syntax and a virtual DOM for efficient updates and rendering. ReactJS is maintained by Facebook and has become one of the most popular libraries for front-end development due to its simplicity, performance, and flexibility. ReactJS code can be written using the .jsx extension, which allows you to mix JavaScript and HTML-like syntax (JSX). This makes React code more readable and easier to write compared to traditional JavaScript. ReactJS is client-side, running directly in the browser, but it can also be used in server-side rendering with frameworks like Next.js. React components can manage their own state and can be easily integrated with other libraries or APIs. ReactJS is widely compatible with modern web browsers, and it allows seamless interaction with HTML elements, such as handling form inputs, button clicks, and other user actions. ReactJS is also used in mobile app development through React Native. React's component-based structure enables you to break down complex UIs into smaller, reusable parts, making development faster and easier. It provides powerful features like hooks, lifecycle methods, and state management, which allow you to manage and manipulate data within your components. ReactJS is highly scalable and ideal for dynamic, high-performance web applications.

## **CSS**

CSS (Cascading Style Sheets) is used to control the style and layout of a web document in a simple and efficient way. CSS defines the presentation of HTML elements, handling aspects such as color, fonts, spacing, and positioning. Unlike HTML, which structures the content, CSS enhances the visual appearance of a webpage. CSS allows you to apply styles across an entire website, making it easier to maintain a consistent design. It supports responsive design, enabling web pages to adapt to different screen sizes using media queries. With CSS, you can create layouts using techniques like Flexbox or CSS Grid, allowing for flexible and dynamic designs. CSS is fundamental for making websites visually appealing and ensuring a great user experience across devices.

## **HTML**

HTML stands for Hyper Text Mark-up Language, which is the most widely used language on Web to develop web pages. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML5 version which is an extension to HTML 4.01, and this version was published in 2012. Originally, HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, and so forth to facilitate the sharing of scientific information between researchers. Now, HTML is being widely used to format web pages with the help of different tags available in HTML language.

## **VISUAL STUDIO CODE**

Visual Studio Code (VS Code) is a lightweight, open-source source code editor developed by Microsoft, designed to work with various programming languages, including JavaScript and React. Built on the Electron framework, VS Code is available for Windows, Linux, and macOS. It offers essential features such as syntax highlighting, intelligent code completion, debugging support, code snippets, and Git integration. The editor is highly customizable, allowing users to change themes, keyboard shortcuts, and install extensions to enhance functionality. VS Code is an ideal tool for React development, providing a streamlined and efficient environment for building web applications.

## **INTELLIJ**

IntelliJ IDEA is a powerful IDE that is widely used for Java development, particularly for building Spring Boot applications. It provides features such as advanced code refactoring, smart code completion, real-time debugging, and integration with build tools like Maven and Gradle. IntelliJ IDEA offers robust support for Spring Boot, making it a preferred choice for developers working on enterprise-level Java applications. With a focus on productivity and user experience, IntelliJ ensures that developers can quickly build, test, and deploy Spring Boot applications with ease.



### 3.4 FUNCTIONAL REQUIREMENTS

The **Sustainable Finance** is an advanced banking software solution designed to assist individuals, enterprises, and governments in securing loans for sustainable development projects. Built with **Java Spring Boot** for the backend and **ReactJS** for the frontend, this system supports a cloud-based deployment model, ensuring scalability, performance, and accessibility. The software focuses on providing green loans at reduced interest rates, encouraging investments in environmentally sustainable initiatives.

Automated Grading System with College Handling provides the modules like

- ❖ Applicant Module
- ❖ Bank Finance Module
- ❖ Admin Module

#### **Applicant Module**

- ❖ Registration
- ❖ Loan category viewing
- ❖ Questionnaire answering
- ❖ Eligibility checking
- ❖ Complaints adding

#### **Bank Module**

- ❖ Registration
- ❖ Questionnaire adding
- ❖ Rules managing
- ❖ Response viewing
- ❖ Loan approving

#### **Admin Module**

- ❖ Complaints handling
- ❖

**SYSTEM DESIGN**

#### 4. SYSTEM DESIGN

Design of a system can be defined as the process of applying various techniques and principles for the purpose of defining a device, a process or system is sufficient details to permit its physical realization. Thus, system design is a solution a “how to” approach to the creation of a new system. This important phase provides the understanding and procedural details necessary for implementing the system recommended in the feasibility study.

The data design transforms the information domain model created during analysis into the data structure that will be required to implement the software. The architectural design defines the relationship among major structural components into a procedural detail necessary for implementing the system recommended in the feasibility study.

The data design transforms the information domain model created during analysis into the data structure that will be required to implement the software. The architectural design defines the relationship among major structural components into a procedural description of the software. Source code is generated and testing is conducted to integrate and validate the software. Project management point of view software design is conducted in two into data and software architecture. There are two levels of the system design:

- Logical design
- Physical design

In the logical design, the designer produces a specification of the major features of the system which meets the objectives. The delivered product of logical design includes current requirements of the following system components:

- Input design.
- Output design.
- Database design.

Physical design takes this logical design blueprint and produces the program software, files and a working system. Design specifications instruct programmers about what the system should do. The programmers in turn write the programs that accept input from users, process data, produce reports, and store data in files. Structured design is a data flow-based methodology that partitions a program into a hierarchy of modules organized in a top-down manner with details.

## 4.1 DATABASE DESIGN

A database is a collection of interrelated data stored with minimum redundancy to serve users more quickly and efficiently. James Martin defines database as "A collection of data designed to be used by different programs". The general objective of a database is to make information access easy, quick, inexpensive, integrated, and shared by different applications and users.

Database design is an important yet sometimes overlooked part of the application development lifecycle. An accurate and up-to-date data model can serve as an important reference tool for Database Administrators, developers, and other members of joint application development team.

A good database design does the following:

- Provide minimum search time when locating specific records.
- Stored data in the most efficient manner possible to keep the database from
- Growing too large
- Make data update as easy as possible.
- Is flexible enough to allow inclusion of new functions required of the programs.

The database design is a two-level process. In the first step user requirements are gathered together and a database is designed which will meet these requirements as cleanly as possible. This step is called Information Level Design, and it is taken independent of any individual DBMS.

In the second step this information level design is transferred into a design for the specific DBMS that will be used to implement the system in question. This step is called Physical Level Design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve two major objectives, the two objectives are:

- Data Integrity
- Data independence

The data base design is made up of three levels:

- Conceptual level (High level)
- Physical level (Low level)
- View level (Representation level)

### Conceptual Level

Conceptual level describes the essential features of the system data. It uses symbols and is called entity-relationship analysis. An entity is a conceptual representation of an object.

### Physical Level

In this level data is stored physically. That is an internal schema that describes the physical storage structure of the database.

### View Level

This level is used to describe how the user views the records or objects in the database. Data structuring is refined through a process called normalization. Data is grouped into the simplest way possible, so that later changes can be made with a minimum impact on data structures. Based on the requirements determined during the definition phase of project life cycle, the data elements describing the entity were determined. They are later submitted to normalization to remove redundancies and to optimize them.

The concepts and techniques used when designing an effective database includes:

An **entity** is a logical collection of things that are relevant to a database. The physical counterpart of an entity is a database table.

An **attribute** is a descriptive or quantitative characteristic of an entity. The physical counterpart of an attribute is a database column (or field).

A **primary key** is an attribute (or combination of attributes) that uniquely identifies each instance of an entity. A primary key cannot be null, and the value assigned to a primary key should not change over time. A primary key also needs to be efficient. For example, a primary key that is associated with an INTEGER data type will be more efficient than one that is associated with a CHAR data type.

A **relationship** is a logical link between two entities. A relationship represents a business rule and can be expressed as a verb phrase.

A **foreign key** exists when the primary key of a parent entity exists in a child entity. A foreign key requires that values must be present in the parent entity before like values may be inserted in the child entity.

## **Normalization**

Normalization is the process of efficiently organizing data in a database. There are two goals of the normalization process: eliminate redundant data (for example, storing the same data in more than one table) and ensure data dependencies make sense (only storing related data in a table). Both if these are worthily goals as they reduce the amount of space a database consumes and ensure that data is logically stored.

The process of refining the data model and creating it with a database is called normalization. Database designers can implement several levels of normalization. Each level builds on the previous level by reducing the amount of redundancy between tables. This typically increases performance and avoids problems with data consistency.

### **1. First Normal Form**

- First normal form is the most basic level of database normalization. The keys to creating tables in the 1NF.
- Eliminate repeating groups in individual tables.
- Create a separate table for each set of related data.
- Identify each set of related data with a primary key.

### **2. Second Normal Form**

The first normal form is not good database design, but it is a start. The keys to creating tables in the 2NF create separate tables for set of values that apply to multiple records.

### **3. Third Normal Form**

The third normal form is typically the last form that most database designers will normalize to. After 3NF, the number of relationships between tables becomes very large, and performance begins to decrease on the database server. The goal of the third normal form is to remove all data that does not depend on the primary key of table.

**Advantages of normalization are:**

- Help in reduction in the complexity of maintaining data integrity by removing the redundant data.
- It reduces inconsistency of data.
- Eliminate the repeating fields

## TABLES

Table name: EduVance\_studentreg

Purpose: To store student details.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store student id
photo	varchar	100	Not Null	To store student photo
admno	varchar	10	Not Null	To store student admission number
name	varchar	50	Not Null	To store student full name
address	varchar	40	Not Null	To store student address
gender	varchar	10	Not Null	To store student gender
dob	date	20	Not Null	To store student date of birth
department	varchar	40	Not Null	To store student course
semester	int	11	Not Null	To store student semester
contactno	varchar	10	Not Null	To store student contact number
login_id_id	bigint		Foreign key	To store student login id



Table name: EduVance\_teacherreg

Purpose: To store teacher details.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store teacher id
teacherid	varchar	40	Not Null	To store automated teacher college id
tphoto	varchar	100	Not Null	To store teacher photo
tname	varchar	50	Not Null	To store teacher full name
tgender	varchar	10	Not Null	To store student gender
age	varchar	20	Not Null	To store teacher age
tdepartment	varchar	40	Not Null	To store teacher course
tcontactno	varchar	10	Not Null	To store teacher contact number
tcertificate	varchar	100	Not Null	To store teacher certificate
tqualification	varchar	40	Not Null	To store teacher qualification
texp	varchar	40	Not Null	To store teacher experience
referenceletter	varchar	100	Not Null	To store teacher reference letter
login_id_id	bigint		Foreign key	To store teacher login id

Table name: EduVance\_login

Purpose: To store login details.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store login id
email	varchar	254	Not null	To store email
password	varchar	50	Not null	To store password
usertype	int	11	Not null	To store usertype

Table name: EduVance\_essay

Purpose: To store the uploaded essay.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store essay id
essay	varchar	100	Not Null	To store essay file
current_date	datetime		Not Null	To store current date
login_id_id	bigint		Foreign key	To store login id
tea_id_id	bigint		Foreign key	To store teacher id

Table name: EduVance\_answer

Purpose: To store the uploaded answer sheet.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store answer sheet id
answer	varchar	100	Not null	To store answer sheet file
current_date	datetime		Not null	To store current date
login_id_id	bigint		Foreign key	To store login id
t_id_id	bigint		Foreign key	To store teacher id

Table name: EduVance\_omr

Purpose: To store the uploaded OMR sheet.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store OMR sheet id
omr	varchar	100	Not null	To store OMR sheet file
current_date	datetime		Not null	To store current date
login_id_id	bigint		Foreign key	To store login id
tc_id_id	bigint		Foreign key	To store teacher id

Table name: EduVance\_assignment

Purpose: To store the uploaded assignment.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store assignment file id
assignment	varchar	100	Not null	To store assignment file
current_date	datetime		Not null	To store current date
login_id_id	bigint		Foreign key	To store login id
ta_id_id	bigint		Foreign key	To store teacher id

Table name: EduVance\_attendance

Purpose: To store the attendance of students.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store attendance id
current_date	date		Not null	To store current date
present	int	11	Not null	To store as present
absent	int	11	Not null	To store as absent
login_id_id	bigint		Foreign key	To store login id
t_id_id	bigint		Foreign key	To store teacher id

Table name: EduVance\_subject

Purpose: To store the subjects of departments and semesters.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store subject id
dept	varchar	40	Not null	To store department
sem	varchar	40	Not null	To store semester

Table name: EduVance\_internalmarks

Purpose: To store the internal marks of students.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store internal mark id
marks	int	11	Not null	To store marks
stud_id_id	bigint		Foreign key	To store student id
login_id_id	bigint		Foreign key	To store login id
subject_id_id	bigint		Foreign key	To store subject id

Table name: EduVance\_complaints

Purpose: To store the complaints and replies.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store complaint id
current_date	datetime		Not null	To store current date
complaint	varchar	100	Not null	To store complaints
reply	varchar	100	Not null	To store replies
stud_id_id	bigint		Foreign key	To store student id

Table name: EduVance\_exam

Purpose: To store exam announcements.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store announcement id
date	date		Not null	To store current date
remark	varchar	60	Not null	To store the announcements

Table name: EduVance\_subjectdetail

Purpose: To store subject details.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store subject id
major1	text		Not null	To store major 1
major2	text		Not null	To store major 2
major3	text		Not null	To store major 3
minorsone	text		Not null	To store minor one
minortwo	text		Not null	To store minor two
aeca	text		Not null	To store aec A
aecb	text		Not null	To store aec B
mdc	varchar	60	Not null	To store mdc
vac1	text		Not null	To store vac 1
vac2	text		Not null	To store vac 2
sec	varchar	60	Not null	To store sec
elective1	text		Not null	To store elective 1
elective2	text		Not null	To store elective 2
subject_id_id	bigint		Foreign key	To store subject id

Table name: EduVance\_studentsubjectselection

Purpose: To store the subject selections of students.

Column	Type	Size	Constraints	Description
id	int	11	Primary key	To store subject selection id
student_id_id	bigint		Foreign key	To store student id
subject	bigint		Foreign key	To store subject detail id
minorsone	varchar	100	Not null	To store minor one
minortwo	varchar	100	Not null	To store minor two
aeca	varchar	100	Not null	To store aec A
aecb	varchar	100	Not null	To store aec B
mdc	varchar	100	Not null	To store mdc
vac1	varchar	100	Not null	To store vac 1
vac2	varchar	100	Not null	To store vac 2
sec	varchar	100	Not null	To store sec
elective1	varchar	100	Not null	To store elective 1
elective2	varchar	100	Not null	To store elective 2
created_at	datetime		Not null	To store the time of creation



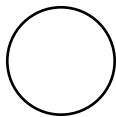
### 4.1.1 DATA FLOW DIAGRAM(DFD)

The Data Flow Diagram (DFD) represents a system at any level of detail with a graphic network of symbols showing data flows, data stores, data processes, and data sources. The purpose of DFD is to provide a semantic bridge between users and system developers. The diagram is the basis of structured system analysis. A level 0 DFD, also called a fundamental system model or a context model represents the entire software elements as a single bubble with input and output indicated by incoming and outgoing arrows respectively. Additional process and information flow parts are represented in the next level i.e., Level 1 DFD. Each of the processes represented at Level 1 are sub functions of the overall system depicted in the context model. Any processes which are complex in Level 1, will be further represented into sub functions in the next level, i.e., in level 2. Data flow diagrams illustrate how data is processed by a system in terms of inputs, and output represented by major components or functions with Circles.

#### Components of Data Flow Diagrams

There are only four symbols that are used in the drawing of data flow diagrams. These are explained below together with the rules that apply to them.

- Process: A process that represents some amount of work being performed on data.



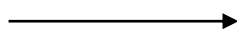
Circle or Bubble

- External Entity: This represents any outside agency, which interact with the system. It represents the source or destination of data for the system under consideration.



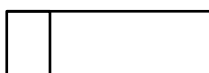
Rectangle

- Data Flow: The data flow portrays an interface among different components in a DFD. It represents flow of data between a process and an external entity or between a process and data store.



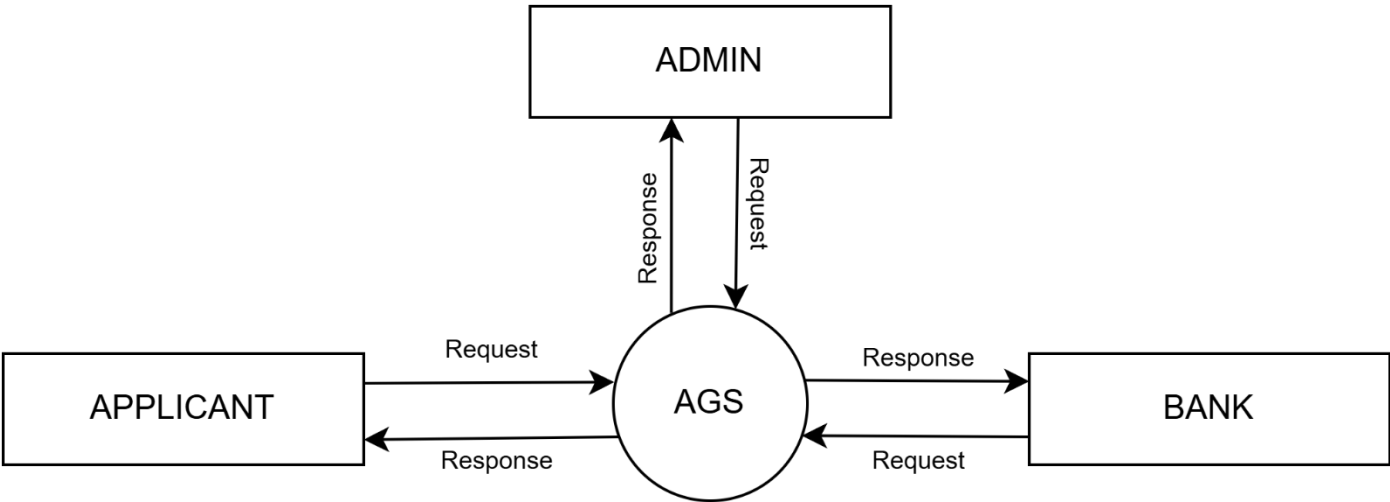
Arrow

- Data Stores: A data store is a place for holding information within the system.

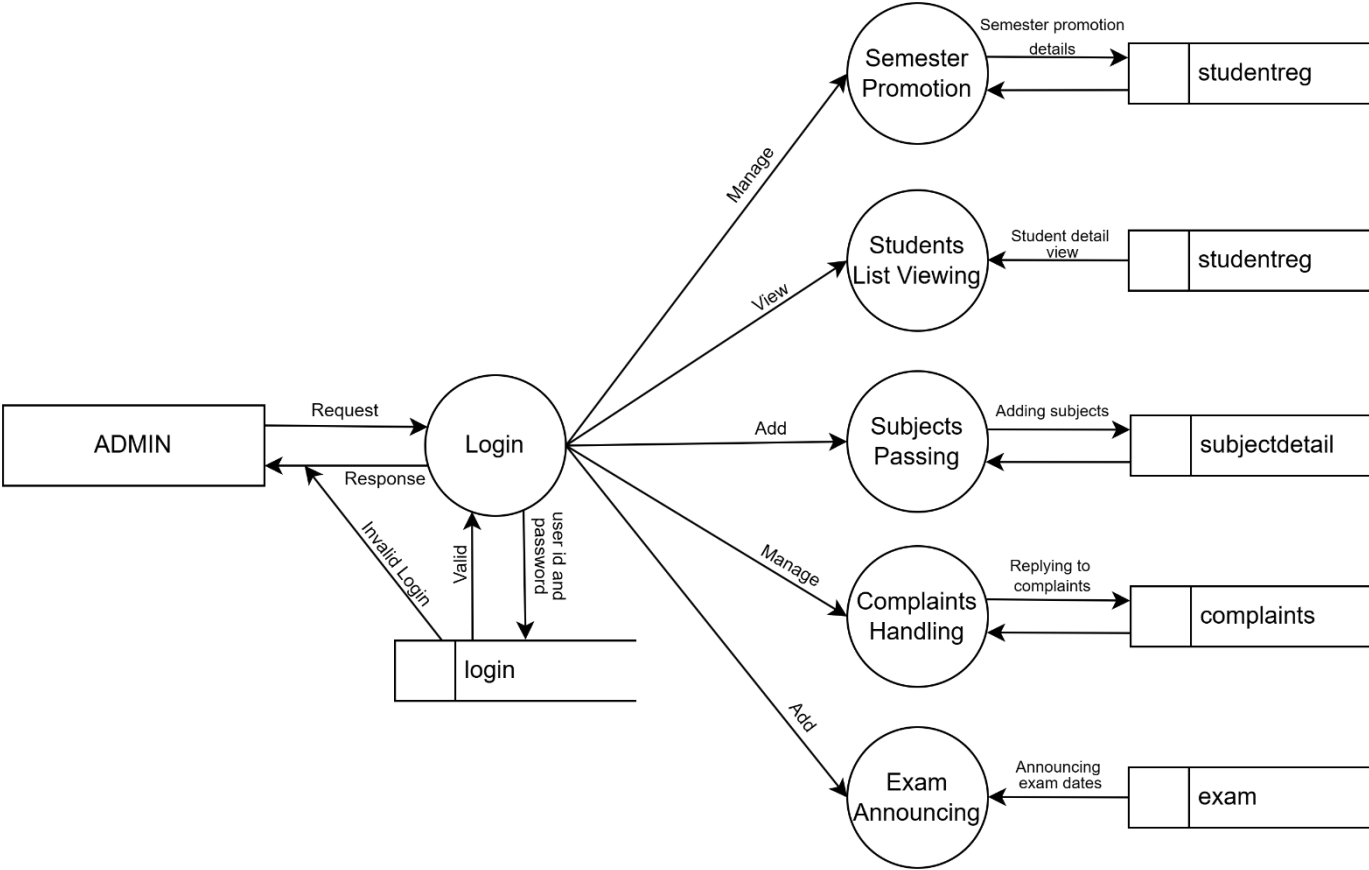


One-end opened rectangle

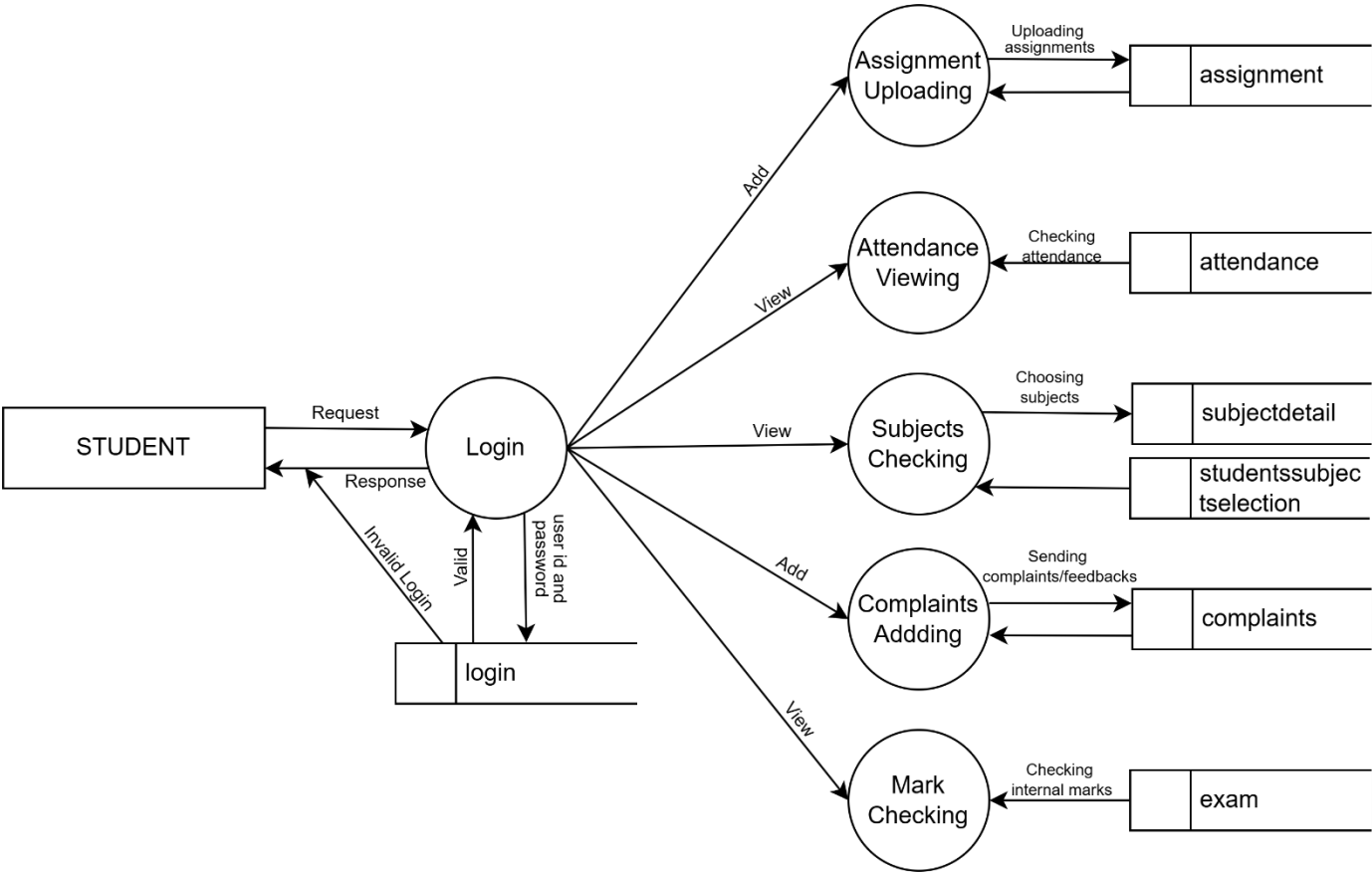
Context level



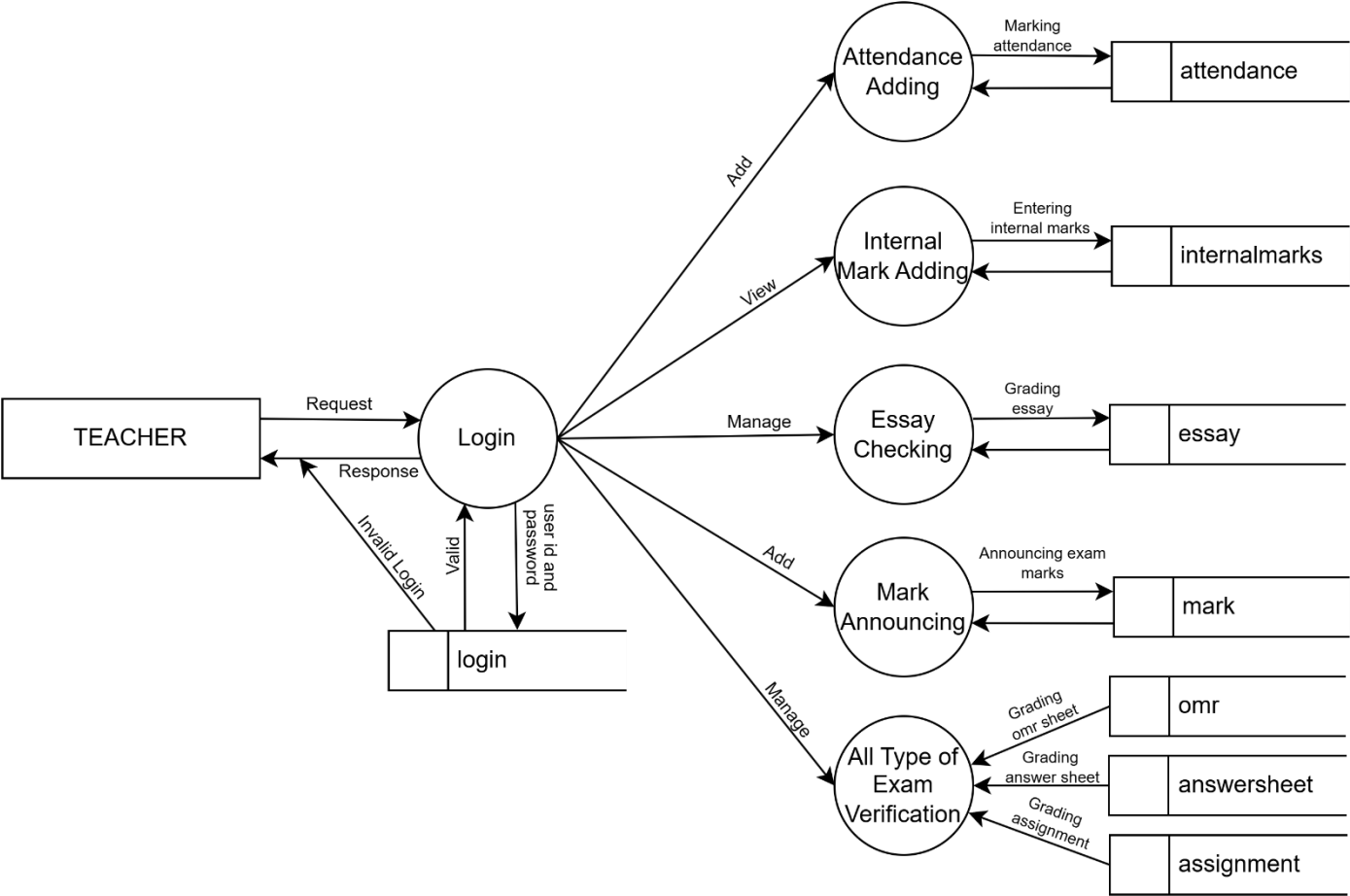
Level 1 Admin



Level 1 Student



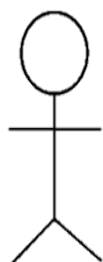
Level 1 Teacher



### 4.1.2 USE CASE DIAGRAM

A use case is a set of scenarios that describes the interaction between user and system. A use case diagram displays the relationship among the actors and use cases. The two main components of a use case diagram are cases and actors. A use case defines the interactions between external actors and system under consideration to accomplish goals. Actors must be able to make decisions, but need not to be human: An actor must be a person, a company or an organization.

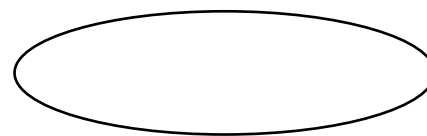
#### Components of Use Case Diagram



**Actor**



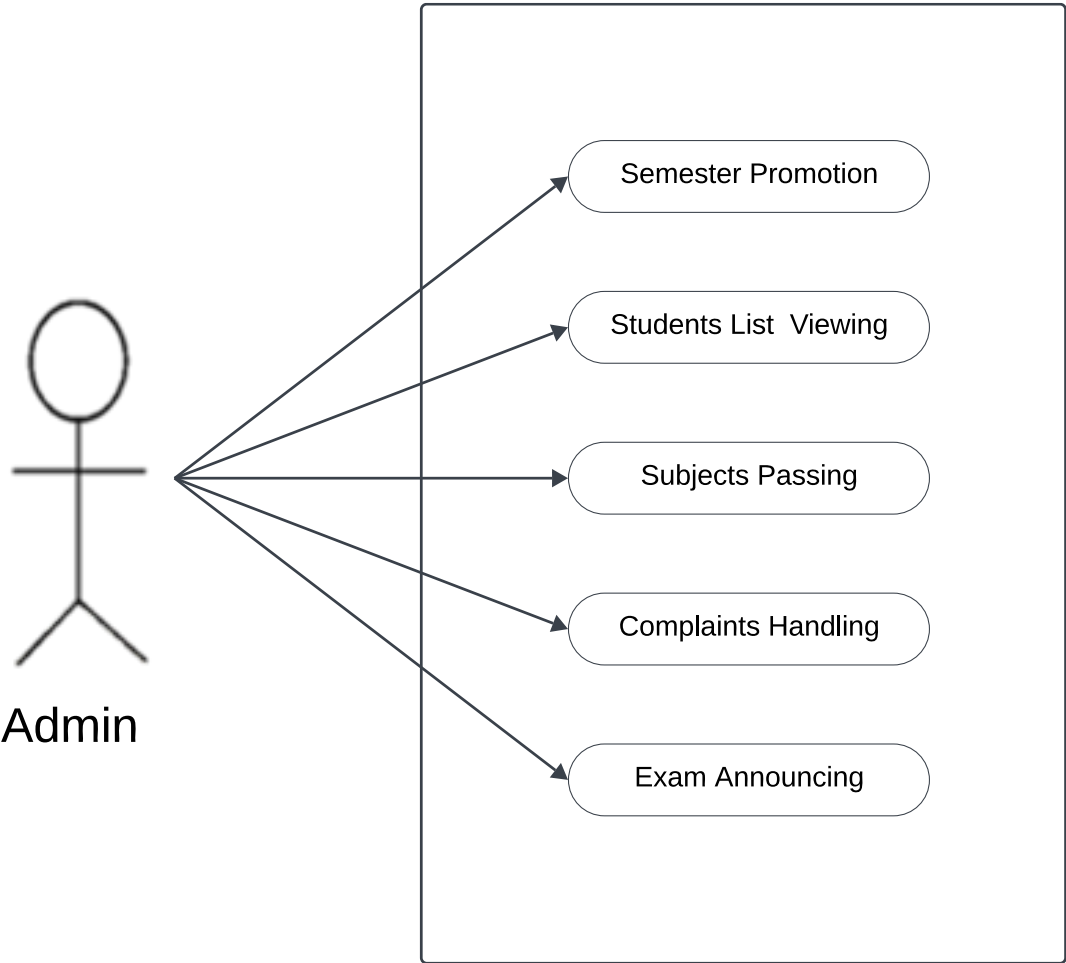
**System Boundary**



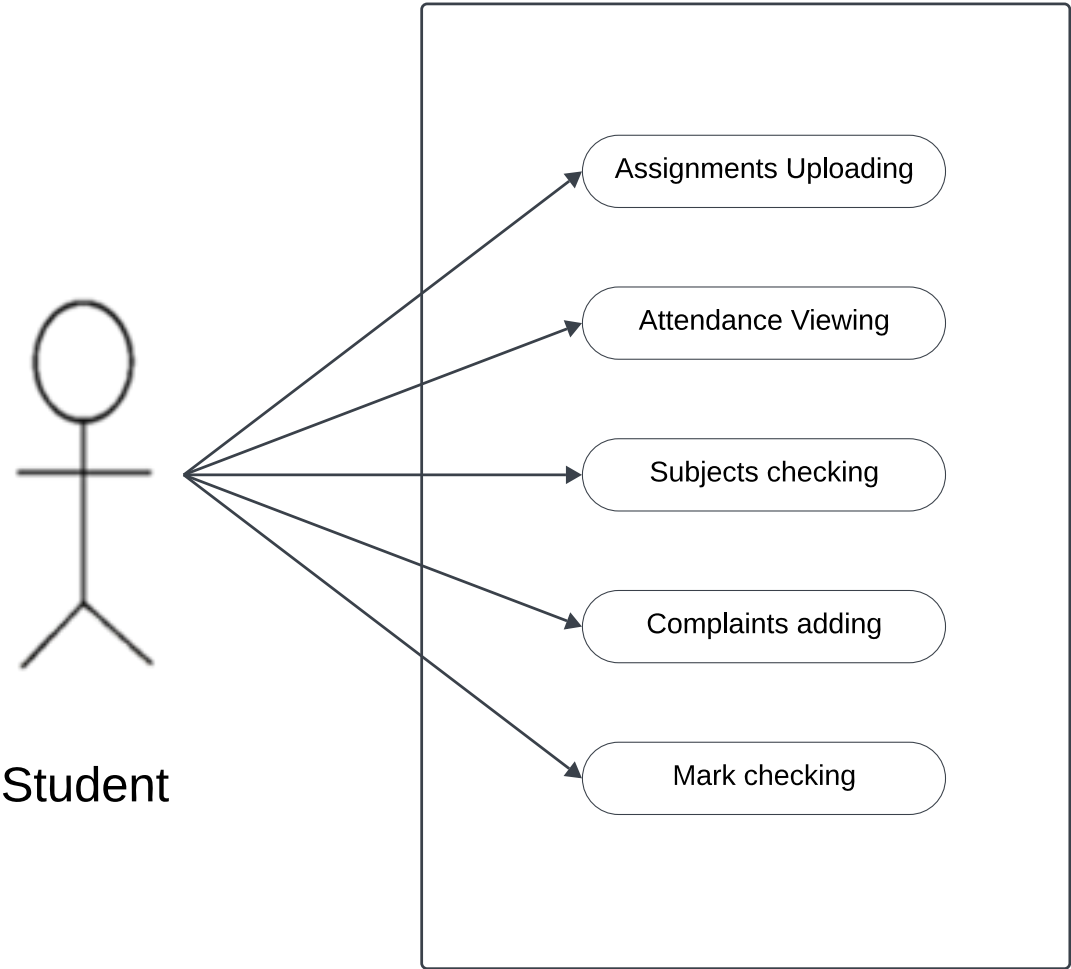
**Use Case**

A use case represents a user or another system that will interact with the system you are modeling. A use case is an external view of the system that represents actions that user might perform to complete a task. Use cases are used in almost every project. They are helpful in exposing requirements and planning the project. During the initial stage of a project most use case should be defined, but as the project continues might become visible.

Use case 1: Admin

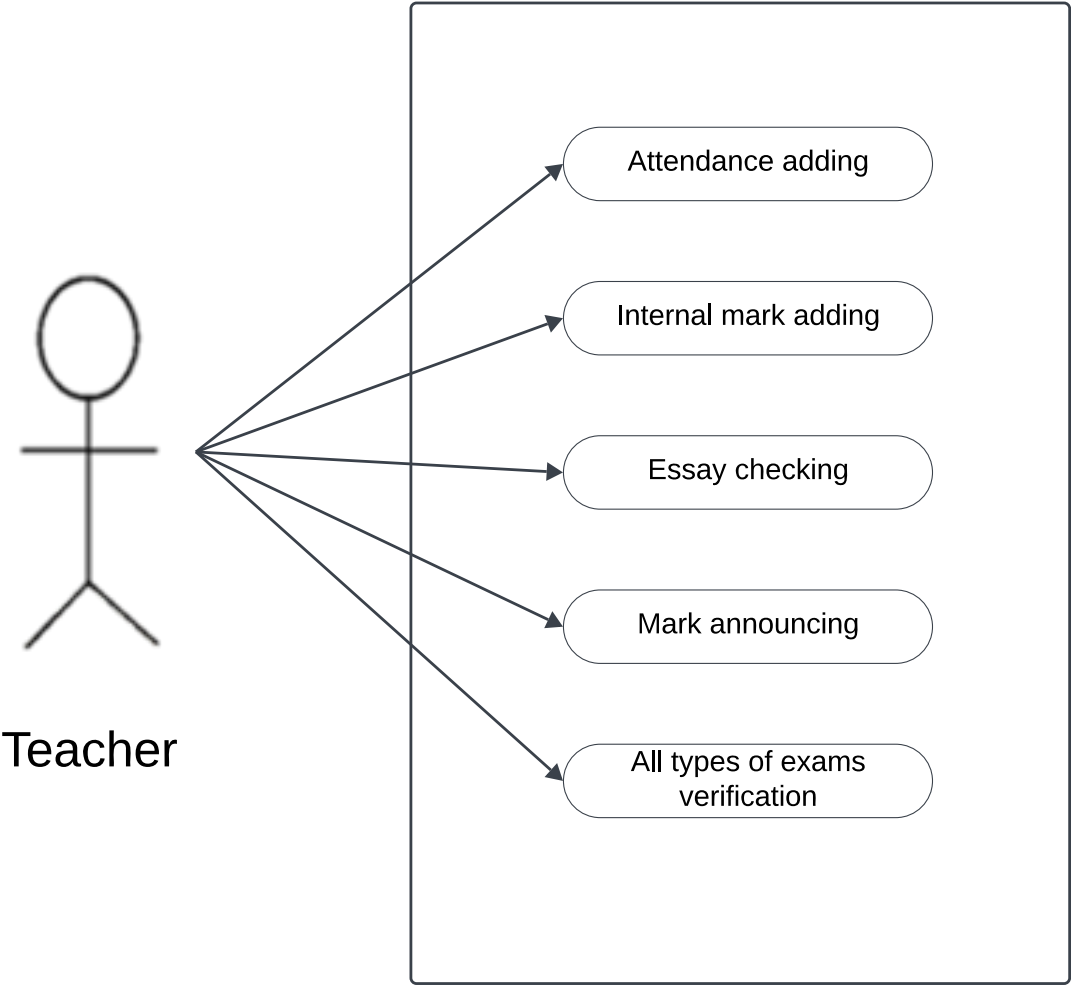


Use case 2: Student

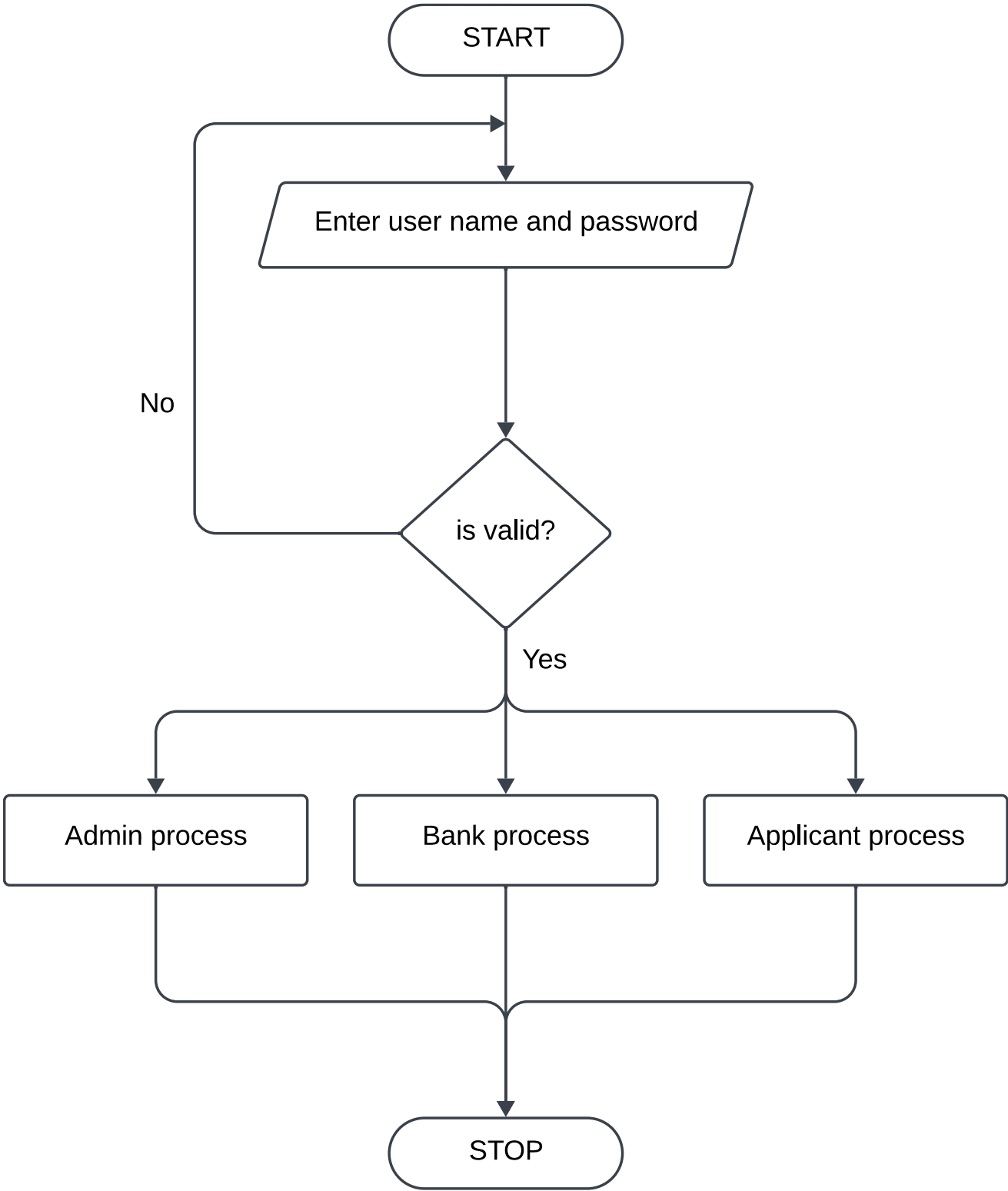




Use case 3: Teacher



4.1.3 FLOWCHART



## 4.2 PROCEDURAL DESIGN

Procedural design is best used to model programs that have an obvious flow of data from input to output. It represents the architecture of a program as a set of interacting processes that data pass data from one another. The steps involved in this are:

- Computer procedure: Specify what functions will be carried out on computer, what will be different programs and in what sequence program will be run.
- Non-computer procedure: Specify the manual procedures for feeding input data, receiving outputs etc.

## 4.3 INTERFACE DESIGN

The purpose of interface design is to communicate effectively through form designs: there are several major requirements:

- The form title must clearly identify its purpose. Columns and rows should be labeled to avoid confusion. The form should be identified by firm name or code number to make it easy to work with.
- The form must be easy to use and fill out. It should be legible, intelligible and uncomplicated.
- The data requested should reflect a logical sequence.

## 4.4 INPUT DESIGN

The input design is the process of converting the user-oriented inputs into the computer-based format. The goal of designing input data is to make automation as easy and free from errors as possible. The input design requirements such as user friendliness, consistent format and interactive dialogue for giving the right message and help for the user at right time are also considered for the development of the project.

The following points should be considered while designing the input:

- What data to input?
- What medium to use?
- How the data should be arranged or coded?

Inaccurate input data is the most common cause of error in processing data. The arrangement of messages as well as placement of data, headings and titles on display screens or source document is also a part of input design. The design of input also includes specifying the means by which end user and system operators direct the system what action to take.

## 4.5 OUTPUT DESIGN

Output generally refers to the results and information that are generated by the system. When output, system analyst must accomplish the following:

- Determine what information to present.
- Decide whether to display, print the information and select the output medium.
- Arrange the presentation of information in an acceptable format.
- Decide how to distribute the output to intended recipients.

**SYSTEM CODING**

## 5. SYSTEM CODING

The coding step is it process that transforms design into a programming language. It translates a details representation of software into a programming language realization. The translation process continues when a compiler accepts source code as input as produces machine dependent object codes output. Quality of source code can be improved by the use of structures coding techniques, good coding style and readable, consistent code format. During coding, some coding standards are to be allowed. This has a purpose, reducing the chance of making it easier for some time to modify the code later on. Coding phase affects both testing and maintenance profoundly.

In order to begin with the coding section, we must first create the default Flutter project after installing the required technologies and dependencies. So, we have to open terminal to type:

```
Username@computer:~$ flutter create attenz
```

Thus, we create a sample project and now we have to add a new folder asset inside it where we add our required fonts, lotties and images to be used in the application. We also create folders screens and databases to work effectively inside the lib folder.

### CODE

#### **views.py**

```
def studentreg(request):
    if request.method == 'POST':
        form=studentform(request.POST,request.FILES)
        logins=loginform(request.POST)
        print(form)
        if form.is_valid() and logins.is_valid():
            a=logins.save(commit=False)
            a.usertype=1
            a.save()
            b=form.save(commit=False)
            b.login_id=a
            b.save()
            messages.success(request,"Form successfully submitted")
        return redirect('main')
```

```
else:
    form=studentform()
    logins=loginform()
    return render(request,'studentreg.html',{'form':form,'login':logins})

def teacherregister(request):
    if request.method == 'POST':
        form = teacherform(request.POST,request.FILES)
        logins = loginform(request.POST)
        print(form)
        if form.is_valid() and logins.is_valid():
            a = logins.save(commit=False)
            a.usertype = 2
            a.save()
            b = form.save(commit=False)
            b.login_id = a
            b.save()
            messages.success(request, "Form successfully submitted")
            return redirect('main')
        else:
            form = teacherform()
            logins=loginform()
            return render(request,'teacherreg.html',{'form':form,'login':logins})

def login(request):
    if request.method == 'POST':
        form=login_check(request.POST)
        if form.is_valid():
            print('hiiii')
            email=form.cleaned_data['email']
            password=form.cleaned_data['password']
            try:
                user=Login.objects.get(email=email)
                if user.password==password:
                    if user.usertype==1:
                        request.session['stud_id']=user.id
```

```

        return redirect('user')
    elif user.usertype==2:
        request.session['t_id']=user.id
        return redirect('tuser')
    elif user.usertype==3:
        request.session['a_id']=user.id
        return redirect('admin')
    else:
        messages.error(request,'invalid password')
except Login.DoesNotExist:
    messages.error(request,'User Does Not Exist')
else:
    form=login_check()
return render(request,'login.html',{ 'login':form})

```

### **forms.py**

```

class studentform(forms.ModelForm):
    gender_choices = (
        ('male', 'Male'),
        ('female', 'Female'),
        ('others', 'Others')
    )
    gender = forms.ChoiceField(choices=gender_choices, widget=forms.RadioSelect())
    dept_choices = (
        ('bca', 'BCA'),
        ('bcom', 'B.Com.Computer Application'),
        ('bba', 'BBA'),
        ('b.a.english', 'B.A. English'),
        ('b.sc.psychology', 'B.Sc.Psychology'),
        ('b.com.taxation', 'B.Com.Taxation'),
    )
    department = forms.ChoiceField(choices=dept_choices, widget=forms.Select(attrs={'class':
'form-control'})))
    sem_choices = (
        ('1', 'Semester 1'),
        ('2', 'Semester 2'),

```



```

        ('3', 'Semester 3'),
        ('4', 'Semester 4'),
        ('5', 'Semester 5'),
        ('6', 'Semester 6'),
        ('7', 'Semester 7'),
        ('8', 'Semester 8'),
    )
    semester = forms.ChoiceField(choices=sem_choices, widget=forms.Select(attrs={'class':
'form-control'}))

class Meta:
    model = Studentreg
    fields = ['admno', 'name', 'address', 'gender', 'dob', 'department', 'semester',
'contactno', 'photo']
    widgets = {
        'admno': forms.TextInput(attrs={'class': 'form-control'}),
        'name': forms.TextInput(attrs={'class': 'form-control'}),
        'address': forms.Textarea(attrs={'class': 'form-control'}),
        'gender': forms.RadioSelect(), # Radio buttons
        'department': forms.Select(attrs={'class': 'form-control'}), # Select dropdown
        'semester': forms.Select(attrs={'class': 'form-control'}), # Select dropdown
        'dob': forms.DateInput(attrs={'type': 'date', 'class': 'form-control'}),
        'contactno': forms.TextInput(attrs={'class': 'form-control'}),
        'photo': forms.FileInput(attrs={'class': 'form-control'}),
    }

class loginform(forms.ModelForm):
    class Meta:
        model=Login
        fields=['email','password']
        widgets={
            'email':forms.EmailInput(attrs={'class': 'form-control'}),
            'password':forms.PasswordInput(attrs={'class': 'form-control'}),
        }

```

```

class teacherform(forms.ModelForm):
    gender_choices=(
        ('male','Male'),('female','Female'),('others','Others')
    )
    tgender=forms.ChoiceField(choices=gender_choices, widget=forms.RadioSelect())
    tdept_choices = (
        ('bca', 'BCA'),
        ('bcom', 'B.Com.Computer Application'),
        ('bba', 'BBA'),
        ('b.a.english', 'B.A. English'),
        ('b.sc.psychology', 'B.Sc.Psychology'),
        ('b.com.taxation', 'B.Com.Taxation'),
    )
    tdepartment = forms.ChoiceField(choices=tdept_choices, widget=forms.Select(attrs={'class':
'form-control'})))
    class Meta:
        model = teacherreg
        fields =
['tname','tgender','age','tdepartment','tcontactno','tphoto','tcertificate','tqualification','treferencelette
r','texp']
        widgets={
            'tname':forms.TextInput(attrs={'class':'form-control'}),
            'tgender':forms.Select(attrs={'class':'form-control'}),
            'tdepartment':forms.Select(attrs={'class':'form-control'}),
            'age':forms.TextInput(attrs={'class':'form-control'}),
            'tcontactno':forms.TextInput(attrs={'class':'form-control'}),
            'tphoto':forms.FileInput(attrs={'class':'form-control'}),
            'tcertificate':forms.FileInput(attrs={'class':'form-control'}),
            'tqualification':forms.TextInput(attrs={'class':'form-control'}),
            'treferenceletter':forms.FileInput(attrs={'class':'form-control'}),
            'texp':forms.TextInput(attrs={'class':'form-control'}),
        }

```

**urls.py**

```
from django.contrib import admin
from django.urls import path, include
```

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path("", include('EduVance.urls'))
]
```

**settings.py**

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'EduVance'
]
```

**login.html**

```
{% load static %}
<!DOCTYPE html>
<html>
    <head>
        <title>Login</title>
    </head>
    <body>
        <div class="container">
            <div class="login-box">
                <h3>Login</h3>
                <form method="POST">
                    {% csrf_token %}
                    <div class="input-box">
                        <h4>Email</h4>
```

```

        {{ login.email }}
    </div>

<div class="input-box">
    <h4>Password</h4>
    <input type="password" name="password" id="id_password" placeholder="Enter
your password" required>
</div>
<!-- <div class="forgot-pass">
    <a href="#">Forgot your password?</a>
</div> -->
<div class="form_group">
    <button type="submit" class="btn" value="login">Login</button>
</div>
</form>
</form>
</div>
</div>
</body>
</html>

```

### studentreg.html

```

{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Register - Edumon</title>
</head>
<body>
    <section class="login_register section-padding">
        <div class="container">
            <div class="row">
                <div class="col-lg-6 offset-lg-3 col-xs-12 wow fadeIn">
                    <div class="register">
                        <h4 class="login_register_title">Create a new account:</h4>

```

```
<form method="post" enctype='multipart/form-data'>
  { % csrf_token % }
  <table align="center">

    <tr>
      <td> PHOTO</td>
      <td> {{ form.photo }}</td>
    </tr>
    <tr>
      <td> APAR ID</td>
      <td> {{ form.admno }}</td>
    </tr>
    <tr>
      <td>FULL NAME</td>
      <td> {{ form.name }}</td>
    </tr>
    <tr>
      <td> ADDRESS</td>
      <td> {{ form.address }}</td>
    </tr>
    <tr>
      <td>GENDER</td>
      <td> {{ form.gender }}</td>
    </tr>
    <tr>
      <td> DATE OF BIRTH</td>
      <td> {{ form.dob }}</td>
    </tr>
    <tr>
      <td> DEPARTMENT</td>
      <td> {{ form.department }}</td>
    </tr>
    <tr>
      <td>SEMESTER</td>
      <td> {{ form.semester }}</td>
    </tr>
```

```
<tr>
  <td> CONTACT NO</td>
  <td> {{ form.contactno }}</td>
</tr>
<tr>
  <td> EMAIL ADDRESS</td>
  <td> {{ login.email }}</td>
</tr>
<tr>
  <td>PASSWORD</td>
  <td> {{ login.password }}</td>
</tr>
</form>
</table>
<div class="form-group col-lg-12">
  <button
    class="bg_btn btn"
    type="submit"
    name="submit">Signup now</button>
</div>
<p>Already have an account? <a href="login.html">Login</a></p>
</div>
</div>
</div>
</section>
</body>
</html>
```

**teacherreg.html**

```

{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Register - Edumon</title>
</head>
<body>
    <section class="login_register section-padding">
        <div class="container">
            <div class="row">
                <div class="col-lg-6 offset-lg-3 col-xs-12 wow fadeIn">
                    <div class="register">
                        <h4 class="login_register_title">Create a new account:</h4>
                        <form method="post" enctype='multipart/form-data'>
                            {% csrf_token %}
                            <table align="center">
                                <tr>
                                    <td>PHOTO</td>
                                    <td>{{ form.tphoto }}</td>
                                </tr>
                                <tr>
                                    <td>FULL NAME</td>
                                    <td>{{ form.tname }}</td>
                                </tr>
                                <tr>
                                    <td>GENDER</td>
                                    <td>{{ form.tgender }}</td>
                                </tr>
                                <tr>
                                    <td>AGE</td>
                                    <td>{{ form.age }}</td>
                                </tr>
                                <tr>
                                    <td>DEPARTMENT</td>
                                    <td>{{ form.tdepartment }}</td>
                                </tr>
                                <tr>
                                    <td>UPLOAD CERTIFICATE</td>
                                    <td>{{ form.tcertificate }}</td>
                                </tr>
                                <tr>
                                    <td>REFERNECE LETTER </td>
                                    <td>{{ form.treferenceletter }}</td>
                                </tr>
                                <tr>
                                    <td>QUALIFICATION </td>
                                    <td>{{ form.tqualification }}</td>
                                </tr>
                            </table>
                        </form>
                    </div>
                </div>
            </div>
        </div>
    </section>

```

```
<tr>
  <td>EXPERIENCE </td>
  <td>{{ form.texp }}</td>
</tr>

<tr>
  <td>CONTACT NO</td>
  <td>{{ form.tcontactno }}</td>
</tr>

<tr>
  <td>EMAIL ADDRESS</td>
  <td>{{ login.email }}</td>
</tr>
<tr>
  <td>PASSWORD</td>
  <td>{{ login.password }}</td>
</tr>
</table>
<button type="submit">Submit</button>
</form>

<div class="form-group col-lg-12">
  <button
    class="bg_btn bt"
    type="submit"
    name="submit">Signup now</button>
</div>
<p>Already have an account? <a href="login.html">Login</a></p>
</div>

</div>
</div>
</div>
</section>
</body>
</html>
```



**SYSTEM TESTING**

## **6. SYSTEM TESTING**

### **6.1 TESTING**

Testing is the last stage of the software development before we release the product to the customer. Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. Software testing can be looked upon as one among the many processes. Testing cannot show the absence of defects, it can only show that software defects are present.

#### **Software Testing Techniques**

The importance of testing and its impact on software cannot be underestimated. The greater visibility of software systems and the cost associated with the software failure are motivating factors for planning through testing. It is not uncommon for a software organization to spend 40% of its effort on testing.

A number of rules that act as Testing Objectives are:

- Testing is the process of executing a program with the aim of finding errors.
- A good test case will have a good chance to find an undiscovered error.
- A successful test case uncovers a new error.

### **6.2 UNIT TESTING**

The first level testing is unit testing. Unit testing concentrates on each unit if the software is implemented in source code. Initially tests focus on each module individually, ensuring that it functions properly as a unit. The modules must then be assembled or integrated to form the complete software package.

There are tests that occur as part of unit testing. The module interfaces are tested to ensure that information properly flows into and out of a program under test. The data structures are also tested for integrity. Boundary conditions are tested to ensure that the module operates properly at boundaries established to limit or restrict processing.

## **6.3 INTEGRATION TESTING**

The next level of testing is often called Integration testing in which many tested modules are combined into a sub-system, which are then tested. Data can be lost across an interface; one module can have an adverse effect on the other sub functions, when combined may not produce the desired major functions. Integrated testing is the systematic testing for constructing the uncovered errors within the interface. This testing was done with sample data. The developed system has run successfully for this sample data. The need for integrated test is to find the overall system performance

### **Data Validation Testing**

Data Validation is the process of testing the accuracy of data; a set of rules you can apply to a control to specify the type and range of data that users can enter. It can be used to display error alerts when users enter incorrect values into a form. Rather than checking for errors after a form is completed; data validation verifies values as the form is being filled out.

A strategy for software testing integrates software test case design method into a well-planned series of steps that result in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and then undertaken, and how much effort, time and resources will be required. Therefore, any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote a customized testing approach.

System tests are carried out to validate a duly developed system with a view assuring that it meets its requirements. There are essentially three kinds of system testing.

#### **1. Alpha Testing**

It refers to the system testing that is carried out by the test team within the organization.

#### **2. Beta Testing**

Beta testing is the system testing performed by a selected group of friendly customers.

### **3. Acceptance Testing**

Acceptance testing is the system testing performed by the customer to determine whether or not to accept the delivery of the system. The application is tested to ensure the requirements. Different sets of input data are entered to validate the system. In all cases the system produces a reasonable output.

### **6.3 SYSTEM TESTING**

System testing is a type of software testing that evaluates the complete and integrated software system to ensure it meets the specified requirements. It is performed after integration testing and checks the system as a whole, including all modules, interfaces, and interactions with external systems. The goal is to verify that the software behaves correctly in its intended environment, covering both functional and non-functional aspects like performance, security, and usability.

### **6.4 USER ACCEPTANCE TESTING**

Acceptance testing is a key factor to the success of any system. The system under the consideration was treated for user acceptance by constantly keeping in touch with the prospective system user at the time of developing and making changes wherever and whenever required.

### **White Box Testing**

White box testing strategy deals with the internal logic and structure of the code. White box testing is also called glass, structural, open box or clear box testing. The tests written based on the white box testing strategy incorporate coverage of the code written, branches, paths, statements and internal logic of the code etc.

In order to implement white box testing, the tester has to deal with the code and hence is needed to possess knowledge of coding and logic i.e., internal working of the code. The White box test also needs the tester to look into the code and find out which unit/ statement/ chunk of the code is malfunctioning.

## **Black Box Testing**

Black Box Testing is not a type of testing; it instead is a testing strategy, which does not need any knowledge of internal design or code etc. As the name “black box” suggests, no knowledge of internal logic or code structure is required. The types of testing under this strategy are totally based / focused on the testing for requirements and functionality of the work product/software application. Black box testing is sometimes also called “Opaque testing”, “Functional/Behavioral Testing” and “Closed Box Testing”.

## **Test Cases**

A test case in software engineering is a set of conditions or variables under which a tester will determine whether an application or software system is working correctly or not. The mechanism for determining whether a software system or system has passed or failed such a test is known as test oracle. In some settings, an oracle could be a requirement or use case, while in other side could be a heuristic.

## **Need For Testing**

Testing is essential as

- The existence of program defects or inadequacies is inferred.
- Test the performance of the system.
- Verifies whether the software behaves as intended by its designer.

## **SYSTEM IMPLEMENTATION AND MAINTENANCE**

## **7. SYSTEM IMPLEMENTATION AND MAINTENANCE**

Systems implementation is the process of: defining how the information system should be built (i.e., physical system design), ensuring that the information system is operational and used, ensuring that the information system meets quality standard (i.e., quality assurance)

System maintenance is an umbrella term that encompasses various forms of computer maintenance needed to keep a system running. The two main components of system maintenance are preventive and corrective maintenance.

### **7.1 SYSTEM IMPLEMENTATION**

System implementation phase is the most difficult in the lifecycle. It is defined as the process of converting a new or revised system design into an operational one. Implementation includes all those activities that take place to convert from the old system to new. The old systems consist of manual operations, which are operated in very different manner from the proposed new system. These are several methods for handling the implementation and the consequent conversion from the old system to the new computerized system.

The most secure method for conversion from the old system to the new system is to run both systems in parallel. In this approach a person may operate in the manual older processing system as well as start operating the computerized system, we can depend upon the manual system.

Another commonly used method is a direct cut over from the existing manual system to the computerized system. The system may work within a week or a day. There are no parallel activities. However, there is no remedy in case of a problem. This strategy requires careful planning. A working version of the system can also be implemented in one part of the organization and the personal will be piloting the system and changes can be made as and when required. But this method is less preferable due to the loss of the entire system. This type of conversion is usually difficult and is not properly planned; it can give way to a number of problems. Implementations of a modified application to replace an existing one, using the same computer are the next method. This type of conversion is relatively easy to handle. Provided there are no changes in the file. In our project we implemented a new computer system to replace an existing one.

## **7.2 SYSTEM MAINTENANCE**

Maintenance is a characteristic of design and implementation, which is expressed, as the probability that an item will be retained in or restored to a specific condition within a given period of time, when maintenance is performed to accordance with the prescribed procedures and resource.

Maintenance is the enigma of system development. It holds the software industry captive, tying up programming resources. Analyst and programmers spend far more time maintaining program than they do writing them.

Maintenance can be classified as corrective, adaptive or prefecture. Corrective maintenance means repairing processing or performance failures or making changes because of previously uncorrected problem or false assumptions. Adaptive maintenance means repairing processing or performance or modifying the program to respond to the users additional or changing needs. Of this type more time and money are spent on prefecture than on corrective and adaptive maintenance.

Technical and management approaches to the maintenance phase can be implemented with little upheaval. However, tasks performed during the software engineering process define maintainability and have an important on the success of any maintenance approach.

## **7.3 FUTURE SCOPE & FUTURE ENHANCEMENT**

The future scope of the Automated Grading System with College Handling lies in its potential for further integration, scalability, and adaptability to emerging educational technologies. One significant direction for expansion is the integration of Artificial Intelligence (AI) and Machine Learning (ML) for more personalized grading and feedback. Another opportunity lies in expanding the system's compatibility with different types of assessments, such as project-based evaluations, oral exams, or interactive simulations, which would further enhance the system's flexibility.

Additionally, the system could be extended to support integration with Learning Management Systems (LMS) and other third-party platforms. This would create a seamless connection between grading, course content, and student performance analytics. With the growing trend of online and hybrid learning, future development could also include enhanced mobile functionality, providing students and faculty with access to grading, results, and academic records on the go. Furthermore, the system could be scaled to support multi-institutional use, allowing universities and colleges from different regions to collaborate and share academic data securely.



To ensure the system remains adaptable to future needs, several enhancements can be considered. A key enhancement would be incorporating advanced security features to safeguard student data and ensure compliance with global data privacy regulations (such as GDPR). Implementing blockchain technology could further enhance security and transparency, especially for grading and certification processes. Another enhancement could be the incorporation of predictive analytics to help instructors and administrators make data-driven decisions based on trends in student performance. The system could flag students at risk of underperforming, recommend tailored learning interventions, or even suggest curriculum adjustments to better meet student needs. Additionally, providing multilingual support would make the system more accessible to a global audience, ensuring that institutions worldwide can implement it without language barriers.

Lastly, enhancing the user experience with improved dashboards and reporting tools for administrators and faculty would streamline the management process. This could include advanced visualization of student performance trends, automated report generation, and more granular control over the grading criteria and process customization.

**CONCLUSION**

## 8. CONCLUSION

In conclusion, the “**Automated Grading System with College Handling**” is a robust and efficient solution that streamlines the grading process while also integrating essential college management features. By automating the grading tasks, the system reduces the chances of errors, saves valuable time for educators, and ensures consistent evaluation. Additionally, the inclusion of college handling functions, such as student registration, course management, and result analysis, provides a holistic approach to academic administration. This system enhances overall productivity, improves transparency, and fosters a more organized academic environment. Ultimately, the software offers a seamless user experience for both students and faculty, supporting the effective management of academic operations in modern educational institutions.

The Automated Grading System not only simplifies the grading process but also offers scalability to accommodate institutions of varying sizes. By centralizing grading data and automating routine tasks, the system minimizes administrative overhead and empowers faculty members to focus more on personalized teaching and student interaction. The real-time feedback feature is particularly beneficial, as it provides students with immediate insights into their performance, facilitating a more proactive approach to learning. With the integration of algorithms capable of grading assignments, quizzes, and exams, the system ensures fairness and consistency in evaluating students' work.

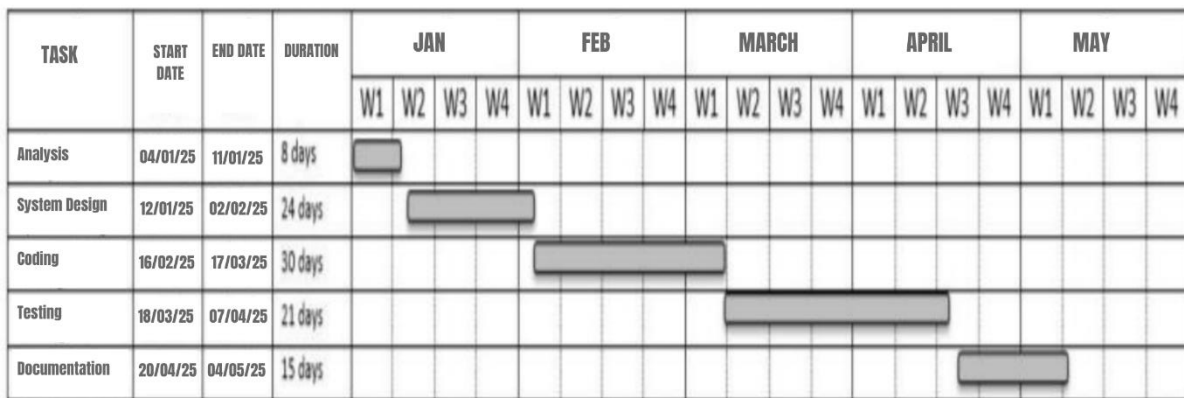
Furthermore, the college handling aspect of the system enhances its utility by enabling seamless management of student records, course registrations, and academic progress tracking. With a user-friendly interface, administrators can easily manage complex academic data and generate insightful reports. The system's flexibility also allows for easy customization to meet the specific needs of different academic programs or educational institutions. Overall, this software system significantly contributes to improving the operational efficiency and academic experience, making it an invaluable tool for both educational administrators and students.

**APPENDIX**

## 9. APPENDIX

### 9.1 GANTT CHART

The risk and uncertainty raise multirole with respect to the size of the project, even when the project is developed according to set methodologies. There are tools available which aid effective project management. A few are described using the Gantt chart. It was devised by Henry Gantt (1971). It represents a project schedule with respect to time periods. It is a horizontal bar chart with bars representing activities and time scheduled for the project activities. Tasks that can be completed in parallel:



## 9.2 MEETING MINUTES

### MEETING MINUTES

**Date:** 20/12/2024

**Time:** 10.00 am to 4.00 pm

**Location:** Softzane Solutions, Ayur

**Present:**

- 1) Abhirami. S
- 2) Jesiya Joseph
- 3) Parvathy Prasannan
- 4) Sanjana Mariyam Sunil

#### Individual Progress Report

We learned the fundamentals of Python Django language and analyzed our project.

There are five modules in the project

- 1) Admin Module
- 2) Teacher Module
- 3) Students Module

#### Discussion of the Problem To Be Solved By The Software

Our project is **AUTOMATED GRADING SYSTEM WITH COLLEGE HANDLING**.

The software will provide facility for the smooth operation of the application which makes the monitoring by the government easier.

#### Discussion of Software Requirements

Operating System: Windows 10 or 11

Front End: HTML, CSS, JavaScript

Back End: MYSQL

Service – Side: Python Django

#### Schedule Next Meeting

The team meeting will be on 21/12/2024

External Guide

## MEETING MINUTES

**Date:** 21/12/2024

**Time:** 10:00 am to 4:00 pm

**Location:** Softzane Solutions, Ayur

**Present:**

- 1) Abhirami. S
- 2) Jesiya Joseph
- 3) Parvathy Prasannan
- 4) Sanjana Mariyam Sunil

### Individual Progress Report

We learned the fundamentals of Python Django language and studied with the help of examples

### Discussion of Management Plan

The **admin** module is assigned to Jesiya Joseph. The **student** module is assigned to Sanjana Mariyam Sunil. The **teacher** module is assigned to Parvathy Prasannan and Abhirami S.

### Specific Task Assigned with Deadlines

Abhirami. S: learned the fundamentals of Python Django language.

Jesiya Joseph: learned the fundamentals of Python Django language.

Parvathy Prasannan: learned the fundamentals of Python Django language.

Sanjana Mariyam Sunil: learned the fundamentals of Python Django language.

### Schedule Next Meeting:

The team meeting will be on 22/12/2024

External Guide

## MEETING MINUTES

**Date:** 22/12/2024

**Time:** 10:00 am to 4:00 pm

**Location:** Softzane Solutions, Ayur

**Present:**

- 1) Abhirami. S
- 2) Jesiya Joseph
- 3) Parvathy Prasannan
- 4) Sanjana Mariyam Sunil

### Individual Progress Report

We prepared the abstract, collected data and agencies and searched web for templates and images needed for designing

### Discussion of Management Plan

Jesiya Joseph: To design the login page of the project and to design form for admin.

Sanjana Mariyam Sunil: To design forms for student

Parvathy Prasannan: To design forms for teacher

Abhirami S: To design forms for teacher

### Schedule Next Meeting

The team meeting will be on 09/01/2025

External Guide



## MEETING MINUTES

**Date:** 09/01/2025

**Time:** 10:00 am to 4:00 pm

**Location:** Softzane Solutions, Ayur

**Present:**

- 1) Abhirami. S
- 2) Jesiya Joseph
- 3) Parvathy Prasannan
- 4) Sanjana Mariyam Sunil

### Individual Progress Report

Every member completed the designing of forms of each module.

### Discussion Of the Problem to be Solved by the Software

Database is required to store all information and notification entered by the user.

### Discussion Of Management Plan

Tables should be created for storing data for admin, teacher, student.

Specific Task Assigned with Deadlines

Jesiya Joseph: To create tables for admin module

Sanjana Mariyam Sunil: To create tables for student module

Parvathy Prasannan: To create tables for teacher module

Abhirami S: To create tables for teacher module

### Schedule Next Meeting

The team meeting will be on 15/01/2025

External Guide

## MEETING MINUTES

**Date:** 15/01/2025

**Time:** 10:00 am to 4:00 pm

**Location:** Softzane Solutions, Ayur

**Present:**

- 1) Abhirami. S
- 2) Jesiya Joseph
- 3) Parvathy Prasannan
- 4) Sanjana Mariyam Sunil

**Discussion Of Management Plan**

Decided to start the coding for each module

**Schedule Next Meeting**

The team meeting will be on 28/02/2025

External Guide

## MEETING MINUTES

**Date:**28/02/2025

**Time:** 10:00 am to 4:00 pm

**Location:** Softzane Solutions, Ayur

**Present:**

- 1) Abhirami. S
- 2) Jesiya Joseph
- 3) Parvathy Prasannan
- 4) Sanjana Mariyam Sunil

### Individual Progress Report

Every member checked coding for each module. Necessary correction and optimization have been performed.

### Specific Task Assigned with Deadlines

Deadlines to integrate all 3 modules and execute the entire project is on

### Schedule Next Meeting

The team meeting will be on 01/04/2025

External Guide

## MEETING MINUTES

**Date:**01/04/2025

**Time:** 10:00 am to 4:00 pm

**Location:** Softzane Solutions, Ayur

**Present:**

- 1) Abhirami. S
- 2) Jesiya Joseph
- 3) Parvathy Prasannan
- 4) Sanjana Mariyam Sunil

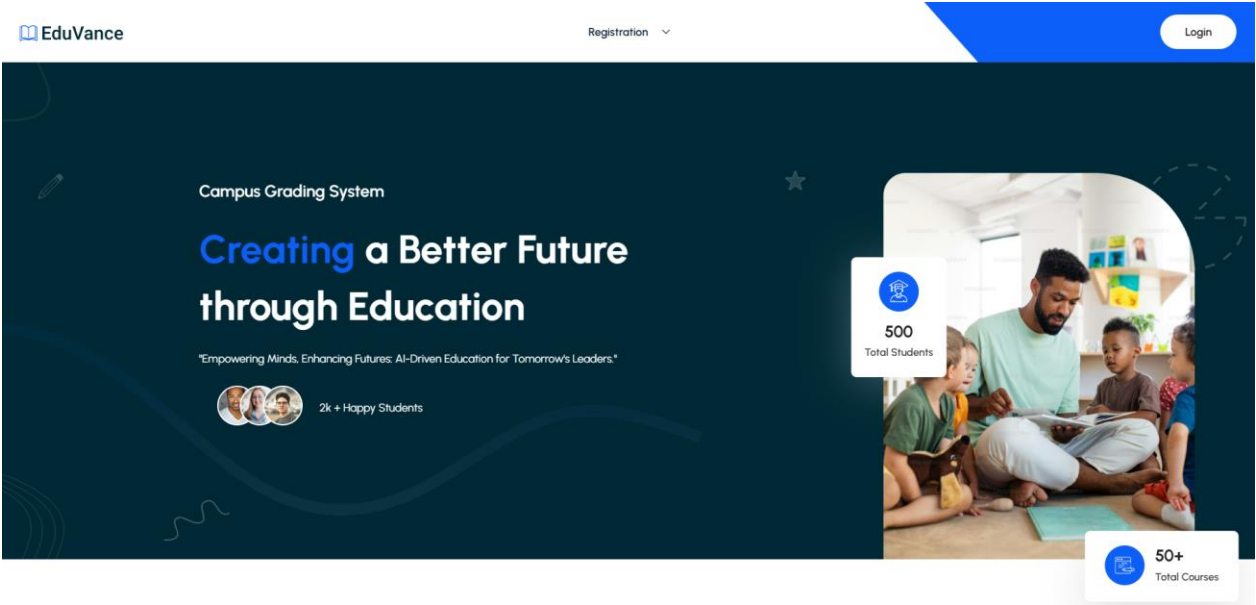
### Individual Progress Report

Necessary correction was made, and the entire project is executed and tested by inputting sample data.

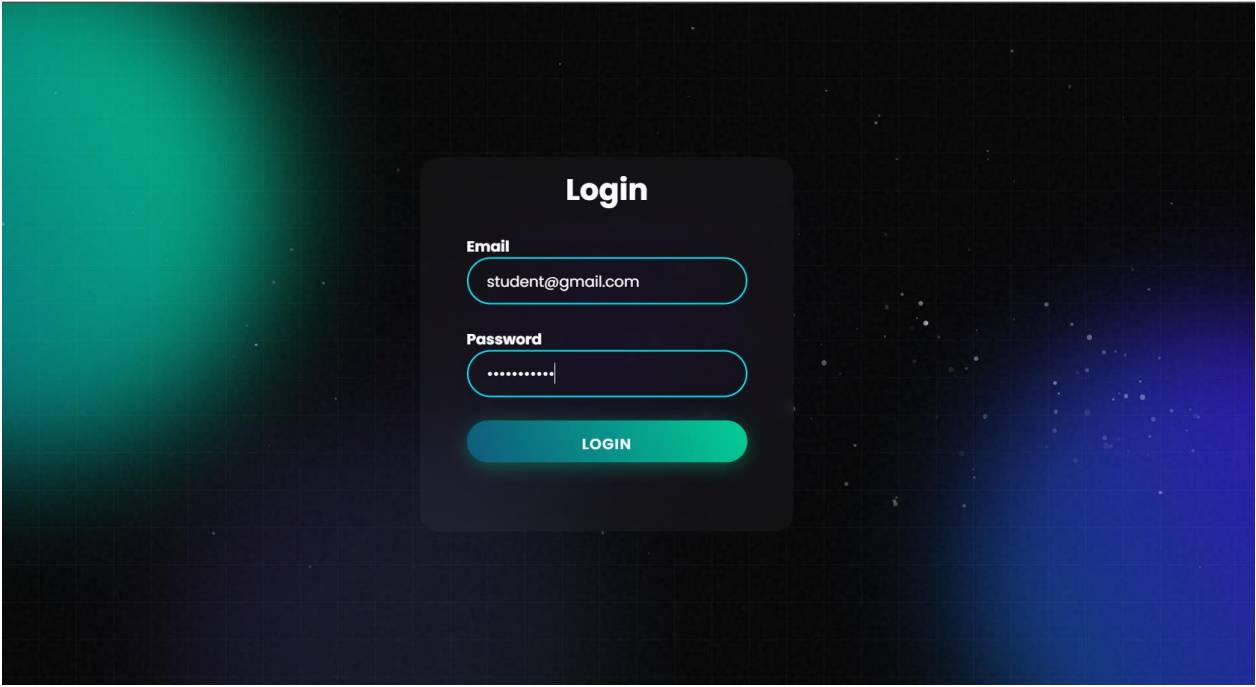
External Guide

9.3 SCREEN LAYOUTS AND REPORTS

Main homepage



Login page



Student registration

EduMan

Home

Control

Pages

Blog

Contact

Register

Register

Create A New Account:

NAME

First Name

Last Name

PHONE NO

EMAIL ID

FULL NAME

ADDRESS

COUNTRY

India

USA

Others

DATE OF BIRTH

DEPARTMENT

SUBJECT

CONTACT NO

EMAIL ADDRESS

REGISTERED

Register Now

Already have an account? Login

Student homepage

EduVance

Choose Subjects

Upload

Attendance

Complaint

Profile

Log Out

Creating a Better Future  
through Education

"Empowering Minds. Enhancing Futures: AI-Driven Education for Tomorrow's Leaders."

50+

Total Courses

Teacher registration

Create A New Account:

PHOTO

Choose File | img.png

FULL NAME

Arya S

GENDER

Male

☒ Female

Others

AGE

37

DEPARTMENT

BCA

UPLOAD CERTIFICATE

Choose File | img.png

REFERNECE LETTER

Choose File | img.png

QUALIFICATION

BCA, MCA

EXPERIENCE

5

CONTACT NO

921290239

EMAIL ADDRESS

aryas@gmail.com

PASSWORD

Submit

Signup now

Already have an account? [Login](#)

Teacher homepage

EduVance

Students

Uploads

Attendance

Internal Exam

Profile


Log out

Creating a Better Future  
through Education

"Empowering Minds. Enhancing Futures: AI-Driven Education for Tomorrow's Leaders."

50+

Total Courses



MAR CHRYSOSTOM COLLEGE OF ARTS AND SCIENCE

**BIBLIOGRAPHY**



## 10. BIBLIOGRAPHY

- <https://github.com/jesiya13/essaygrading>



- <https://chatgpt.com>



- <https://scholar.google.com>



- <https://marchrysostomcollege.com>

- <https://admissions.keralauniversity.ac.in/fyugp2024/programlist.php>