VIT Vedant

A Project Report

Submitted by

Group 1

In partial fulfilment for the award of the degree

of

(MASTER OF COMPUTER APPLICATION)

UNDER THE GUIDANCE OF

Dr. Rizwan Sir

At

VIT Bhopal University

University in Kothri Kalan, Madhya Pradesh

(2023-2025)

CERTIFICATE

This is to certify that the Project report entitled **"VIT Vedant" done by Harsh Kumar(23MCA10108),Kesri Nandan Shukla(23MCA10031),**

Swapnil Patel(23MCA10046),Love Yadav(23MCA10004) and

Patadiya Dhruvakumar Amrishbhai(23MCA10103) is an original work carried out by them in the Department of Master of Computer Application under my guidance. The matter embodied in this project work has not been submitted earlier for the award of any degree or diploma to the best of my knowledge and belief.

Dr. Rizwan	Dr. Anjali Mathur
(Project Guide)	(Program chair)

Contents

ACKNOWLEDGEMENT

The merciful guidance bestowed to us by the almighty made us stick out this project to a successful end. We humbly pray with sincere hearts for his guidance to continue forever

We pay thanks to our project guide Dr. Rizwan, Professor (MCA) who has given guidance and light to us during this project. His versatile knowledge has caused us in critical times during the span of this project.

We pay special thanks to our Program chair MCA Dr. Anjali Mathur who has been always present as a support and helped us in all possible ways during this project.

We also take this opportunity to express our gratitude to all those people who have been directly and indirectly with us during the completion of the project.

We want to thank our friends who have always encouraged us during this project.

Last but not least, thanks to all the faculty of the MCA department who provided valuable suggestions during the period of the project

ABSTRACT

VIT Vedant is a collaborative project initiated to revolutionise online education through a dynamic and user-centric web application. This initiative aims to create a comprehensive online teaching platform that seamlessly integrates advanced features to enhance the teaching and learning experience.

INTRODUCTION

1.1 PROJECT OVERVIEW

VIT Vedant

The primary goal of **VIT Vedant** is to develop a user-centric web application that transcends the limitations of traditional online teaching platforms. The project focuses on providing an intuitive, collaborative, and feature-rich environment to bridge the gap between physical and virtual classrooms. Key objectives include delivering versatile content, and facilitating effective assessment and feedback mechanisms.

Key Features:

1. Intuitive User Experience:

- Designing an interface that is user-friendly and accessible to individuals with varying technical proficiency.
- Prioritising simplicity and ease of navigation to create a positive and engaging user experience.

2. Versatile Content Delivery:

- Supporting a wide range of content formats, including video lectures, interactive quizzes, and multimedia presentations.
- Providing flexibility for educators to cater to diverse learning styles and preferences.

3. Comprehensive Assessment and Feedback:

- Developing robust assessment tools for educators to create and administer quizzes, assignments, and exams.
- Implementing a detailed feedback mechanism to enhance communication between educators and students, promoting continuous improvement.

5. Scalability and Customization:

- Designing an architecture that is scalable to accommodate a growing user base.
- Allowing educators to customise the platform to align with their teaching style, fostering adaptability and ownership.

6. Security Measures:

- Prioritising the security and privacy of user data through robust measures.

- Ensuring strict adherence to privacy protocols to comply with data protection regulations.

Restriction

Google programmable search engines restrict the search query to 10,000/day. A Heroku app (unverified) offers 550 dyno hours whereas Heroku app (verified) offers 1050 dyno hours. In order to reduce the consumption of these dyno hours and keep it to the minimum Heroku always sleeps the free app after 30 minutes of inactivity. To keep the app running continuously for 22 days (unverified account is enough). To make it work for 43 days assuming a continuous usage the account must be verified. 1 dyno offers 512 mb ram So more searches could lead to faster consumption of the dyno hours. A custom domain for all verified accounts is also available to the users. Heroku offers three methods to deploy the app onto it. You can use Heroku GIT (Heroku CLI), Connect your Github repository directly, container registry (for docker based app).

MODULES

Modules for VIT Vedant Web Project:

1. User Authentication Module:

- Functionality: Enables users to create accounts, providing necessary personal information for registration.

- Features:

- User registration with email verification.

- Secure password storage.

- Password recovery mechanisms.

2. Login Module:

- Functionality: Allows registered users to log in securely to access the platform.

- Features:

- Encrypted login credentials.

- Session management for user authentication.

3. User Profile Module:

- Functionality: Permits users to create and manage their profiles.

- Features:

- Profile creation and editing.

- Upload and manage profile pictures.

- Display relevant user information.

4. Permissions and Restrictions Module:

- Functionality: Governs user roles, permissions, and access restrictions.

- Features:

- User role management (e.g., student, teacher, admin).

- Setting access permissions based on user roles.

- Restrictions on certain features based on user roles.

5. Dashboard Module:

- Functionality: Serves as the central hub for users to access relevant information and features.

- Features:

- Personalised dashboard for each user.

- Display of upcoming events, courses, or assignments.

- Quick links to frequently used features.

6. Content Delivery Module:

- Functionality: Manages the delivery of educational content to users.

- Features:

- Upload and organise course materials.

- Support for various content formats (videos, documents, presentations).

- Content versioning for updates.

7. Collaboration Tools Module:

- Functionality: Facilitates real-time interaction and collaboration between users.

- Features:

- Live chat or messaging system.

- Virtual classrooms for interactive sessions.

- Collaborative project spaces.

8. Assessment and Feedback Module:

- Functionality: Handles assessment creation, submission, and feedback.

- Features:

- Quiz and assignment creation.

- Submission and grading functionalities.

- Feedback and communication channels.

9. Attendance Tracking Module:

- Functionality: Monitors and records user attendance.
- Features:
  - Automated attendance tracking.
  - Manual adjustments by instructors.
  - Reporting and analytics on attendance patterns.

10. Security Module:

- Functionality: Ensures the security and privacy of user data.
- Features:
  - Encryption of sensitive information.
  - Regular security audits.
  - Compliance with data protection regulations.

1.

1.2. SYSTEM ENVIRONMENT

The following hardware and software are required for the development and deployment of the system.

1.2.1 Hardware

Processor	:	Core 2 duo
Main Memory	:	2 GB RAM
Hard Disk	:	80 GB
Mouse	:	Standard two-button or higher
Keyboard	:	Standard 101-102 key keyboard
Display	:	14” Monitor
Screen Dimension	:	1025*829 pixels

1.2.2 Software

Operating System	:	Windows 2000/XP and Vista
Frontend Language	:	React js
Backend Language		Node js
Tools	:	Visual Studio Code
Database	:	MongoDB

1.2.3 ABOUT THE LANGUAGE

★ **React JS:**

React (also known as React.js or ReactJS) is a free and open-source front-end JavaScript library for building user interfaces based on UI components. It is maintained by Meta (formerly Facebook) and a community of individual developers and companies. React can be used as a base in the development of single-page, mobile, or server-rendered applications with frameworks like Next.js.

However, React is only concerned with state management and rendering that state to the DOM, so creating React application usually requires the use of additional libraries for routing, as well as certain client-side functionality.

Advantages of ReactJS

ReactJS is intuitive:

ReactJS is extremely intuitive to work with and provides interactivity to the layout of any UI, Plus, it enables fast and quality assured application development that in turn saves time for both – clients and developers.

ReactJS is declarative:

ReactJS enables significant data changes that result in automatic alteration in the selected parts of user interfaces. Owing to this progressive functionality, there is no additional function that you need to perform to update your user interface.

ReactJS provides Reusable Components:

ReactJS provides reusable components that developers have the authority to reuse and create a new application. Reusability is exactly like a remedy for developers. This platform gives the developers the authority to

reuse the components built for some other application having the same functionality. Thereby, reducing the development effort and ensuring a flawless performance.

★

Advantages of ReduxJS

ReduxJS is a pure reducer function:

A pure function is defined as any function that doesn't alter input data, doesn't depend on the external state, and can consistently provide the same output for the same input. As opposed to React, Redux depends on such pure functions. It takes a given state (object) and passes it to each reducer in a loop. In case of any data changes, a new object is returned from the reducer (re-rendering takes place). However, the old object is returned if there are no changes (no re-rendering).

★ **Node JS:**

Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on a JavaScript engine and executes JavaScript code outside a web browser, which was designed to build scalable network applications. Node.js lets developers use JavaScript to write command line tools and for server-side scripting – running scripts server-side to produce dynamic webs page content before the page is sent to the user's web browser.

Advantages of NodeJS

NodeJS boost response time:

Netflix had a start-up time of 40 minutes when they were using java & JavaScript. After that, they switched it to node.js and hit the jackpot by reducing start-up time to under 40 seconds.

NodeJS reduces loading time by quick caching:

The caching module of node.js makes it easy for coders to reduce task workload and re-execution of code.

NodeJS is cost effective with full stack JavaScript:

Node.js allows coders to write server-side code in JavaScript, along with JavaScript code on the frontend with absolute ease. Linked in's development team post migrated from ruby on rails to node.js and they gained performance momentum from node.js advantage. They reduced servers from 30 to only three.

1.2.3 ABOUT THE DATABASE

MongoDB:-

1. Document-Oriented:
   - Data is stored in BSON documents, which are binary representations of JSON-like documents.
   - Documents are composed of key-value pairs and can contain nested structures, arrays, and other data types.

2. Collections:
   - MongoDB stores documents in collections. Collections are analogous to tables in relational databases, but they don't enforce a schema across all documents in the collection.

3. Schema-less:
   - MongoDB is schema-less, meaning each document in a collection can have a different structure.
   - This flexibility makes it easy to evolve the data model as application requirements change.

4. Indexes:
   - MongoDB supports the creation of indexes to optimize query performance. Indexes can be created on single fields or combinations of fields.

5. Query Language:
   - MongoDB provides a rich set of query operators for filtering, sorting, and projecting data.
   - Queries are expressed as JSON-like documents and are powerful and expressive.

6. Scalability:
   - MongoDB is designed to scale horizontally, allowing you to distribute data across multiple servers or clusters to handle larger amounts of data and traffic.

7. Replication:
   - MongoDB supports replica sets, which are groups of MongoDB servers that maintain the same data set. Replica sets provide high availability and fault tolerance.

8. Sharding:
   - Sharding is a method of distributing data across multiple machines. MongoDB uses sharding to horizontally scale by partitioning data across shards.

9. Aggregation Framework:
   - MongoDB provides a powerful aggregation framework that allows you to perform data transformations and computations on the server side.

10. Official Drivers:
    - MongoDB provides official drivers for a variety of programming languages, making it easy to integrate MongoDB with different types of applications.

.

SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

1. Problem Definition and Project Scope:

- Objective Definition: Clearly define the objectives of the system. Identify problems in the existing system or define the purpose of a new system.
- Project Scope: Define the boundaries of the system. Determine what is included and excluded from the project.

2. Feasibility Study:

- Technical Feasibility: Assess whether the technology needed for the system is available or can be developed.
- Operational Feasibility: Evaluate if the system will be used effectively once it is developed and implemented.
- Economic Feasibility: Analyse whether the benefits of the system outweigh the costs.

3. Requirements Gathering:

- User Interviews: Conduct interviews with stakeholders to understand their needs and expectations.
- Surveys and Questionnaires: Use surveys and questionnaires to collect information from a large number of users.

- Document Review: Analyse existing documents, reports, and systems to gather relevant information.

- Brainstorming Sessions: Collaborate with stakeholders to generate ideas and requirements.

4. Requirements Analysis:

- Use Case Modelling: Identify and document use cases to describe the interactions between users and the system.

- Data Modeling: Create data models to represent the structure of the data processed by the system.

- Process Modelling: Develop process models to illustrate the flow of data and control in the system.

- Entity-Relationship Diagrams (ERD):** Define relationships between different entities in the system.

5. System Modelling:

- System Architecture: Define the overall architecture of the system, including hardware, software, and network components.

- Data Flow Diagrams (DFD): Illustrate the flow of data within the system.

- State Diagrams: Represent the different states a system can be in and the transitions between states.

- Class Diagrams: Depict the classes and their relationships in the system.

6. Prototyping:

- Develop Prototypes: Create working prototypes of the system to give stakeholders a tangible feel for the proposed solution.

- Collect Feedback: Gather feedback on the prototypes to refine and enhance the system requirements.

7. Risk Analysis:

- Identify Risks: Identify potential risks that could affect the development and implementation of the system.

- Mitigation Strategies: Develop strategies to mitigate or manage identified risks.

8. Documenting the System Requirements:

- Create SRS: Prepare a comprehensive System Requirements Specification document that outlines all the requirements gathered during the analysis phase

- Review and Approval: Review the SRS with stakeholders to ensure accuracy and obtain approval.

9. Cost-Benefit Analysis:

- Quantify Costs and Benefits: Estimate the costs associated with developing and implementing the system. Compare these costs with the anticipated benefits.

10. Presenting Findings:

- Communication: Present the findings and proposed solutions to stakeholders, addressing any concerns or questions they may have.

System analysis is an iterative process, and feedback from stakeholders is crucial for refining the requirements and ensuring the proposed solution aligns with the organisation's needs. This phase sets the stage for system design, where the detailed specifications for the system are developed based on the requirements identified during analysis.

SYSTEM REQUIREMENTS SPECIFICATION

3. SYSTEM REQUIREMENTS SPECIFICATION

1.1 Purpose

The purpose of this document is to outline the system requirements for the development of an online learning web application.

1.2 Scope

The online learning web application will provide a platform for users to access educational materials, engage in courses, and interact with instructors and other learners.

1.3 Definitions, Acronyms, and Abbreviations

- LMS: Learning Management System
- UI: User Interface
- UX: User Experience
- API: Application Programming Interface

2. System Overview

2.1 System Description

The online learning web application will be a user-friendly platform that facilitates the creation, delivery, and management of educational content.

2.2 System Architecture

The system will follow a client-server architecture with a web-based front end and a backend server for handling data storage, processing, and authentication.

3. Functional Requirements

3.1 User Registration and Authentication

- The system shall allow users to register with a valid email address.
- Users should be able to log in securely with email/password or third-party authentication (e.g., Google, Facebook).
- Passwords must be securely hashed and stored.

3.2 Course Management

- Instructors shall be able to create, edit, and delete courses.
- Each course shall have a unique identifier, title, description, and associated materials.
- Courses can be categorised by subject or level.

3.3 User Dashboard

- Users shall have a personalised dashboard displaying enrolled courses, progress, and recommendations.
- The dashboard shall show upcoming assignments, announcements, and notifications.

3.4 Content Delivery

- Courses shall include multimedia content such as text, video, and interactive quizzes.
- Users shall be able to navigate through course content seamlessly.
- Content should be accessible on various devices.

3.5 Collaboration and Communication

- Users shall have the ability to participate in discussion forums for each course.
- Instructors shall be able to post announcements and updates.
- Private messaging between users and instructors shall be supported.

3.6 Assessment and Evaluation

- The system shall support various types of assessments, including quizzes, assignments, and exams.
- Automated grading shall be implemented for objective assessments.

- Instructors should have the ability to grade subjective assessments.

4. Non-functional Requirements

4.1 Performance

- The system shall support a minimum of 1000 concurrent users.
- Response time for user interactions should be less than 3 seconds.

4.2 Security

- User data, including passwords, shall be encrypted during transmission and storage.
- The application shall implement secure authentication practices.
- Regular security audits and updates shall be conducted.

4.3 Scalability

- The system architecture should be scalable to accommodate a growing user base.
- Database and server configurations must be scalable to handle increased load.

4.4 Usability

- The UI shall be intuitive and user-friendly.

- The application shall be accessible to users with disabilities.
- A responsive design must ensure usability on various devices and screen sizes.

4.5 Reliability

- The system should have a high level of availability (at least 99%).
- Regular backups of the database should be performed.

5. Constraints

- The application must comply with relevant privacy and data protection regulations.
- The development must adhere to budgetary constraints and timelines.

6. Appendices

6.1 Glossary

- List of terms and their definitions used in the document.

6.2 References

- Any external documents or standards referenced in the SRS.

SYSTEM DESIGN

4. SYSTEM DESIGN

System design is the second phase of the software life cycle. The system goes through a logical and physical state of development. The user oriented performance specification is extended into a design specification, while designing the needed system. The design phase begins when the Requirement Specification document for the software to be developed is available. When the Requirement Specification activity is entirely in the problem domain, design is the first step to move from the problem domain to the solution domain. Design is essentially the bridge between the requirements specification and the final solution for satisfying these requirements.

4.1 INPUT DESIGN

Input design is the process of converting a user-oriented description of the inputs to a computer-based business system into a programmer-oriented specification. The design decision for handling input specifies how data are accepted for computer processing. Input design is a part of overall design that needs careful attention.

The collection of input data is considered to be the most expensive part of the system design. Since the inputs have to be planned in such a way so as to get the relevant information, extreme care is taken to obtain the pertinent information. If the data going into the system is incorrect then the processing and outputs will magnify these errors. The goal of designing input data is to make data entry as easy, logical and free from errors as possible.

The following are the objectives of input design:

- To produce a cost effective method of input.

- To ensure validation.

Effort has been made to ensure that input data remains accurate from

The stage at which it is recorded and documented to the stage at which it is accepted by the computer. Validation procedures are also present to detect errors in data input, which is beyond control procedures. Validation procedures are designed to check each record, data item or field against certain criteria.

4.2 OUTPUT DESIGN

The output design phase of the system design is concerned with the conveyance of information to the end users in user-friendly manner. The output design should be efficient, intelligible so that the system relationship with the end user is improved and thereby enhancing the process of decision making.

The output design is an ongoing activity almost from the beginning of the project, efficient and well-defined output design improves the relation of the system and the user. The primary considerations in the design of the output are the requirement of the information and the objective of the end user.

The system output may be of any of the following

- A report

- A document

- A message

4.3ARCHITECTURAL DESIGN

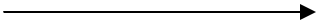
DATA FLOW DESIGN

Data flow diagram (DFD) was first developed by Larry Constantine as a way of representing system requirements in a graphical form; this led to modular design. A DFD describes what data flows (logical) rather than how they are processed, so it does not depend on hardware, software, and data structure or file organisation. It is also known as a 'bubble chart'.

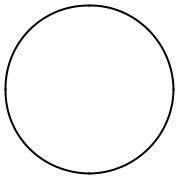
A data Flow Diagram is a structured analysis and design tool that can be used for flowcharting in place of, or in association with, information-oriented and process-oriented system flowcharts. A DFD is a network that describes the flow of data and the processes that change, or transform, data throughout a system. This network is constructed by using a set of symbols that do not imply a physical implementation. It has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design phase that functionally decomposes the requirement specifications down to the lowest level of detail.

A DFD can be considered as an abstract of the logic of an information-oriented or a process-oriented system flow-chart. For these reasons DFDs are often referred to as logical flow diagrams. The four basic symbols used to construct data flow diagrams are shown below:

A rectangle represents a data source or destination.



A directed line represents the flows of data, which is a data stream.



An enclosed figure, usually a circle or an oval bubble, represents a process that transforms data streams.

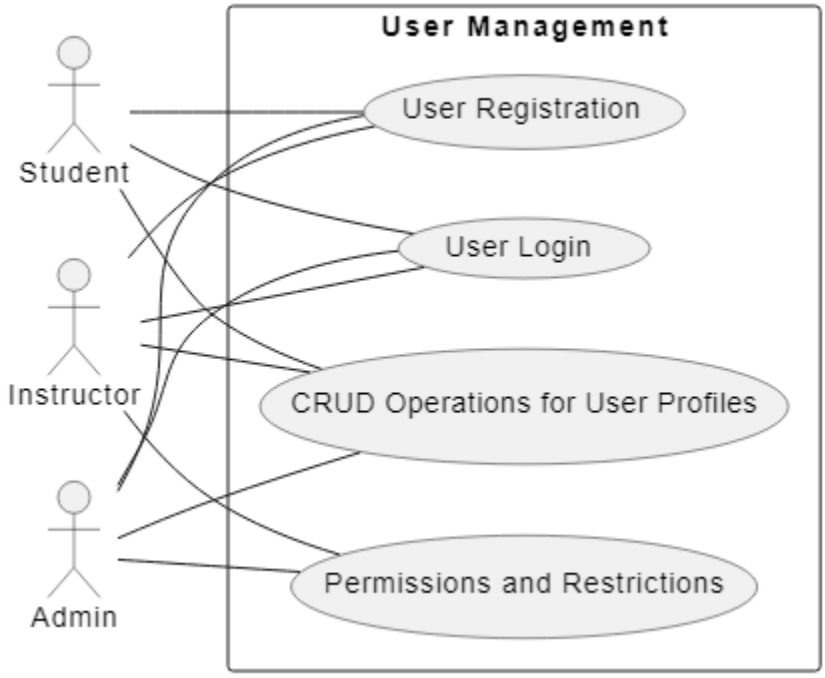
An open-ended rectangle represents data storage.

These are symbols that represent data flows, data sources, data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes. The principal processes that take place at nodes are

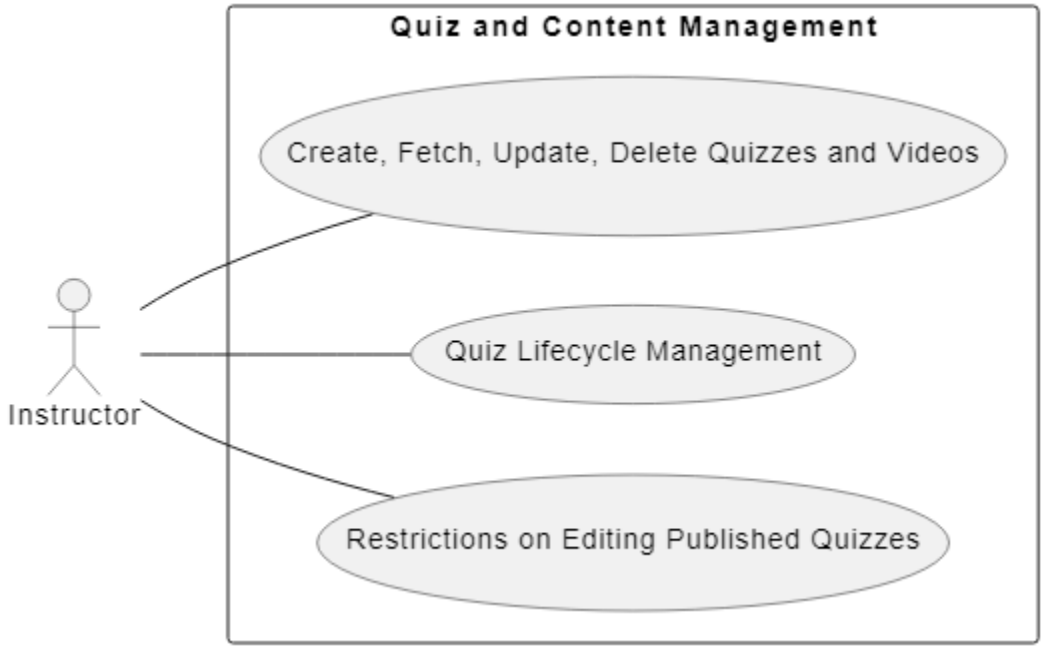
1. Combining data streams
2. Splitting data streams
3. Modifying data streams

Use Case Diagram

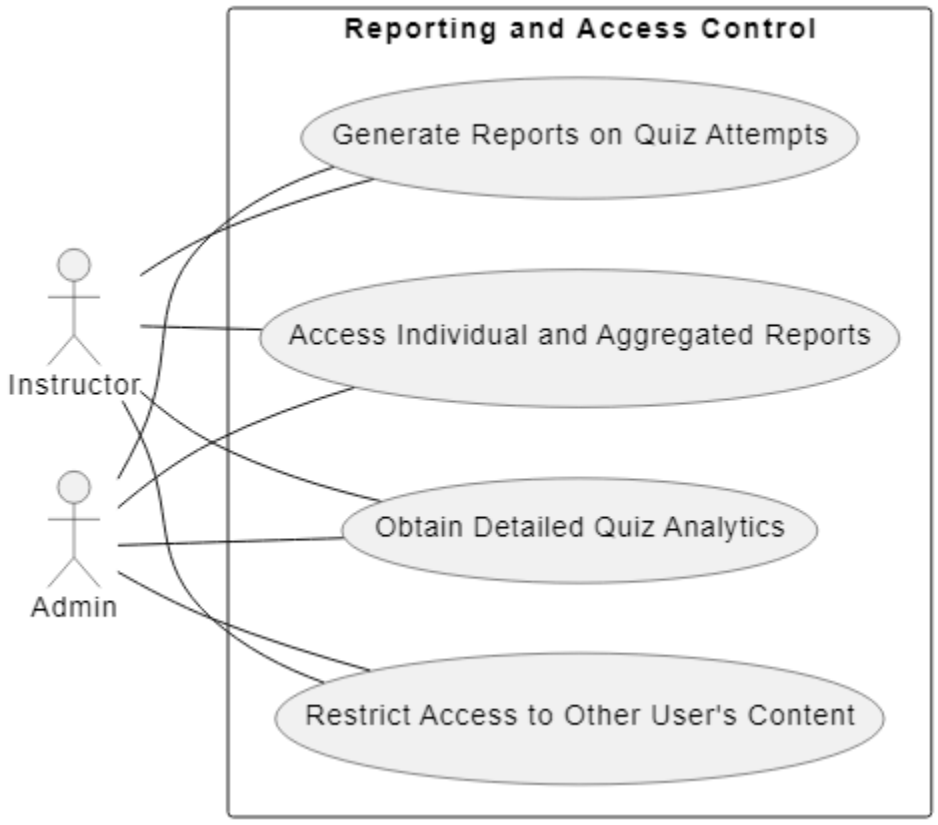
1. User Management



2. Quiz and Content Management



3. Reporting and Access Control



Full diagram (all connected)

Flow Chart

Code

Code Tree Structure

SOFTWARE TESTING

9. SOFTWARE TESTING

System testing provides the file assurance that software once validated must be combined with all other system elements. System testing verifies whether all elements have been combined properly and that overall system function and performance is achieved.

Characteristics of a Good Test:

- Tests are likely to catch bugs
- No redundancy
- Not too simple or too complex

9.1 TYPES OF TESTING

9.1.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of software designing the module. To check whether each module in the software works properly so that it gives desired outputs to the given inputs. All validations and conditions are tested in the module level in the unit test. Control paths are tested to ensure the information properly flows into, and

output of the program unit and out of the program unit under test. Boundary conditions are tested to ensure that the modules operate at boundaries. All independent paths through the control structure ensure that all statements in a module have been executed at least once.

9.1.2 Integration Testing

The major concerns of integration testing are developing an incremental strategy that will limit the complexity of entire actions among components as they are added to the system. Developing a

components as they are added to the system, developing an implementation and integration schedules that will make the modules available when needed, and designing test cases that will demonstrate the viability of the evolving system. Though each program works individually they should work after linking them together. This is also referred to as interfacing. Data may be lost across interfaces and one module can have adverse effects on another. Subroutines after linking may not do the desired function expected by the main routine. Integration testing is a systematic technique for constructing program structure while at the same time conducting tests to uncover errors associated with the interface. In the testing, the programs are constructed and tested in small segments.

Here our objective is to edit, compile and execute Java programs within a single platform. Using an integration test plan prepared in the design phase of the system development guide, the integration test is carried out and all the errors found in the system are corrected for the next testing steps.

9.1.3 System Testing

When a system is developed, it is hoped that it performs properly. In practice however some errors always occur. The main purpose of testing and information systems is to find the errors and correct them. A successful test is one which finds an error.

The main objectives of system testing are:-

- To ensure during operation the system will perform as per specifications.
- To make sure that the system meets the user's requirements during operation.
- To verify that the controls incorporated in the system function as intended.
- To see that when correct inputs are fed to the system the outputs are correct.
- To make sure that during operation incorrect input and output will be deleted.

The scope of a system test should include both manual operations and computerization. Operations system testing is a comprehensive evaluation of the programs, manual procedures, computer operations and controls. System testing is the process of checking if the developed system is working according to the original objectives and requirements. All testing needs to be conducted in accordance with the test conditions specified earlier.

SYSTEM IMPLEMENTATION

10. IMPLEMENTATION

System implementation is the conversion of a new system into an operating one which involves creating compatible files, training staff and installing hardware. A critical factor in conversion is not disrupting the functioning of an organisation. User training is crucial for minimising resistance to change and giving chances to prove its worth.

Training aids, user friendly manuals and healthy screens provide the user with a good start. Software maintenance follows conversion to the extent that changes are necessary to maintain satisfactory operations relative to changes in the user's environment. Maintenance often includes minor enhancements or corrections to the problem that surface late in the systems operations.

In the implementation phase, the team builds the components either from scratch or by composition. Given the architecture document from the design phase and the requirement document from the analysis phase, the team should build exactly what has been requested, though there is still room for innovation and flexibility. For example, a component may be narrowly designed for this particular system, or the component may be made more general to satisfy a reusability guideline. The architecture document should give guidance. Sometimes, this guidance is found in the requirement document.

The implementation phase deals with issues of quality, performance, baselines, libraries, and debugging. The end deliverable is the product itself. During the implementation phase, the system is built according to the specifications from the previous phases. This includes writing code, performing code reviews, performing tests, selecting components for integration, configuration, and integration.

The implementation includes the following things.

- Careful planning
- Investigation of system and constraints.
- Design the methods to achieve the charge over.
- Training the staff in the changed phase.
- Evaluation of change over method.

The method of implementation and time scale to be adopted are found out initially.

MAINTENANCE

11. MAINTENANCE

When the system is in maintenance phase, some people within the system are responsible for collecting maintenance requests from users and other interested parties. The process of maintaining a system is the process of returning to the beginning of the system development phase until changes are implemented.

System maintenance is the activity that occurs following the delivery of the software product enhancement to software products adapting products to new environments and correcting errors. Software product enhancement may involve providing new functional capabilities, improving user displays and modes of intersection or upgrading the performance characteristics of the system.

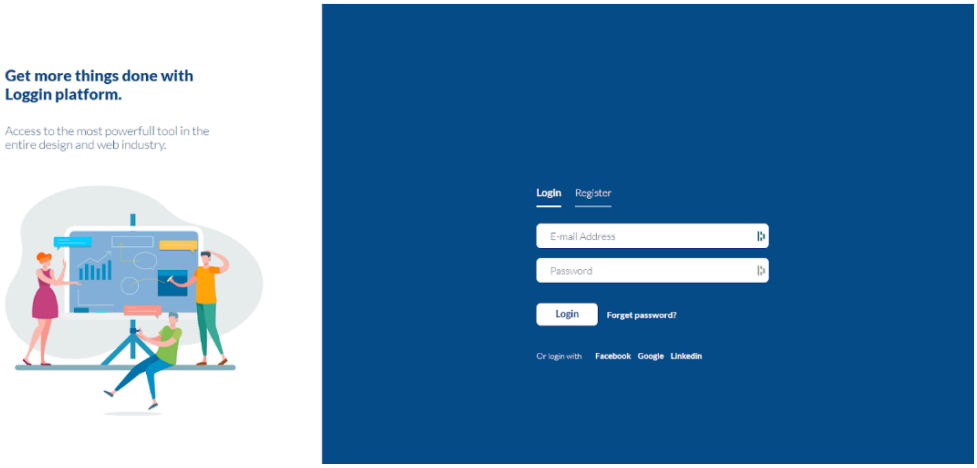
Problem correction involved modification and revalidation of software to correct errors. The process that includes the diagnosis and correction of one or more errors is known as corrective maintenance.

As the software is used for new capabilities, modification and general enhancement may be obtained and this leads to perfect maintenance when software is changed to improve feature maintainability or reliability to preventive maintenance. For maintaining this system the following have to be made strictly.

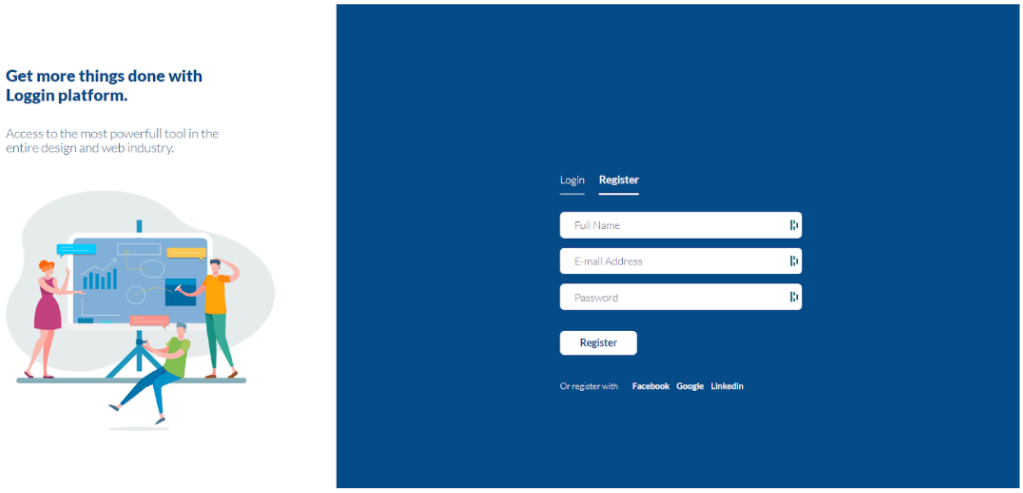
The executable generated several of the forms and reports are only given to the end users. The backup should be taken in order to safeguard the system against any loss of data due to system manufacturing.

Screenshots

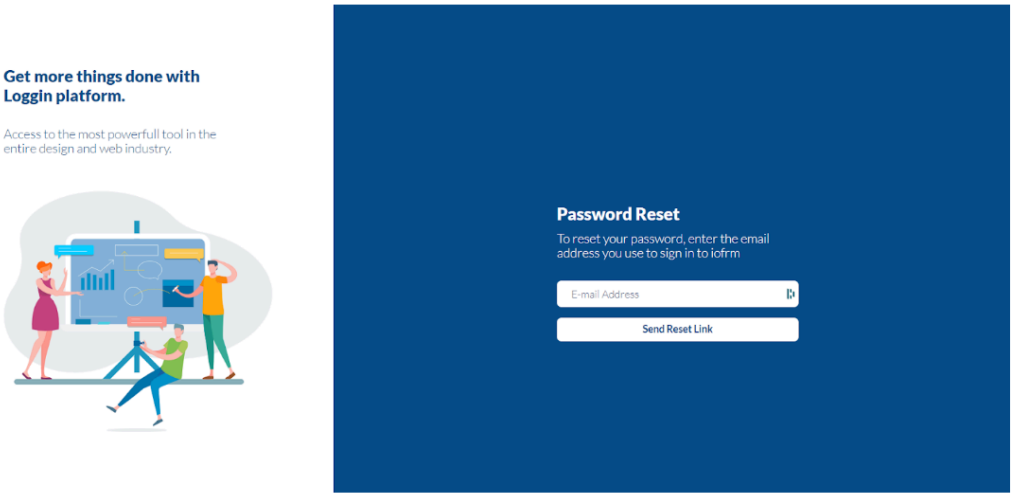
Login Page



Registration Page



Password Reset Page



CONCLUSION

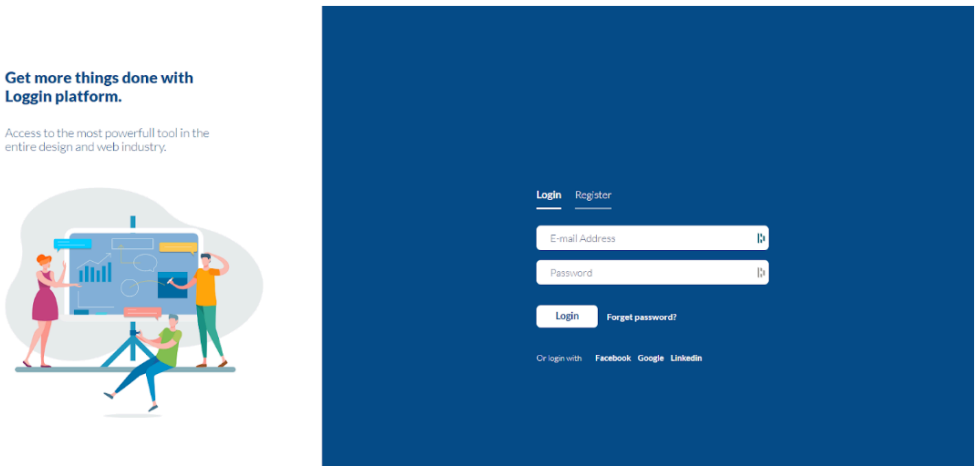
12. CONCLUSION

All the suggestions forwarded during the software proposal have been successfully completed and the final threshold of application has been crossed. Some errors were spotted during the system testing and were corrected. The system developed for the given conditions was found working efficiently. More future work can be established on the project by linking useful websites & adding resources based upon user search query over the period of time. We can make it more reliable and trustworthy by adding quality websites and autosuggestions that can make searches closer to the user's search intent. Adding Diagrams, Images can make the entire app more interactive and helpful.

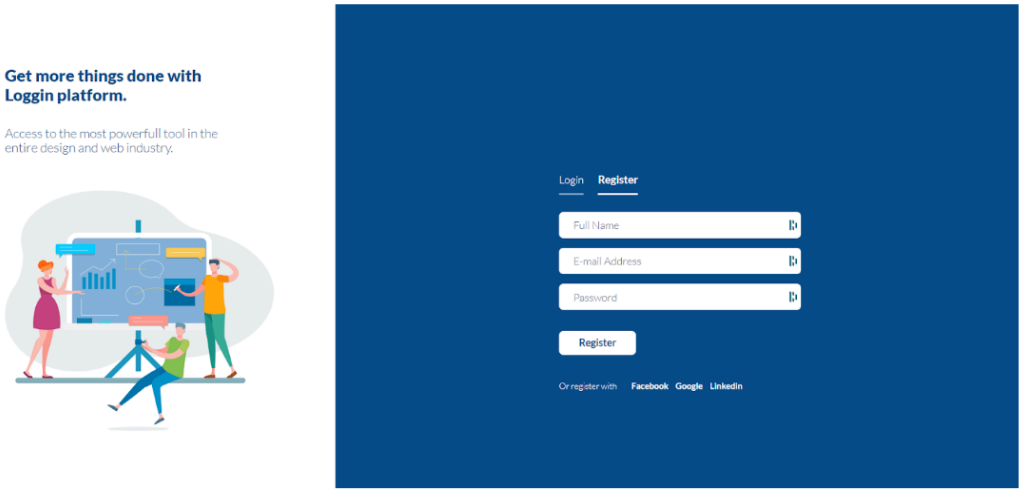
SCREENSHOTS

(Login page)

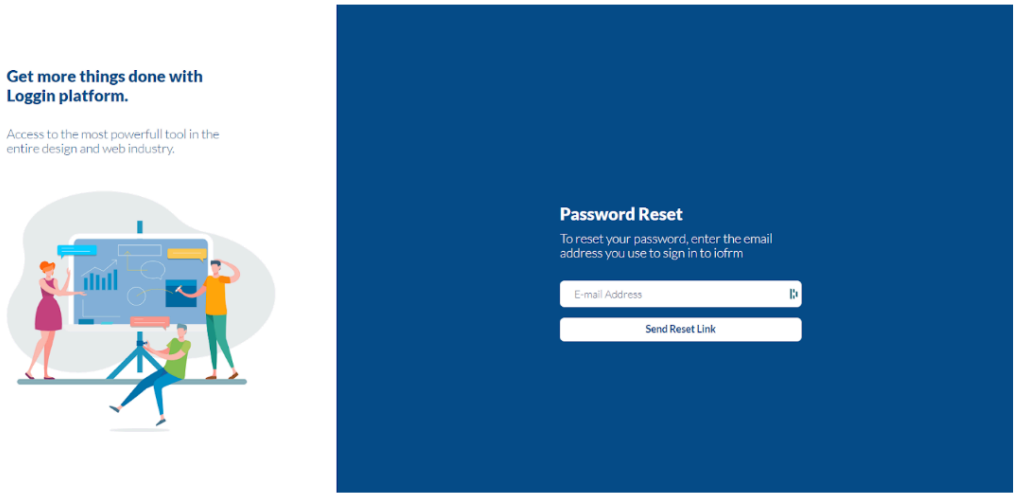
Login Page



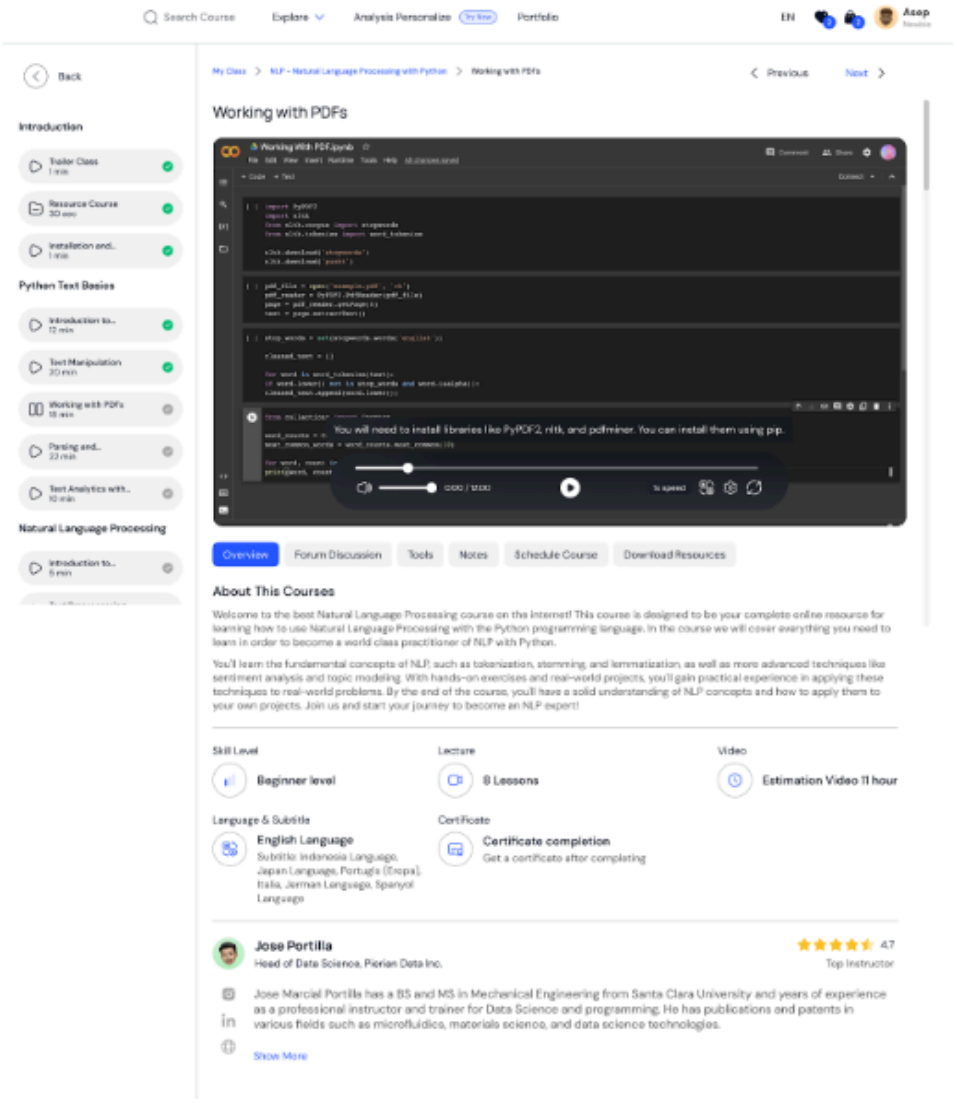
Registration Page



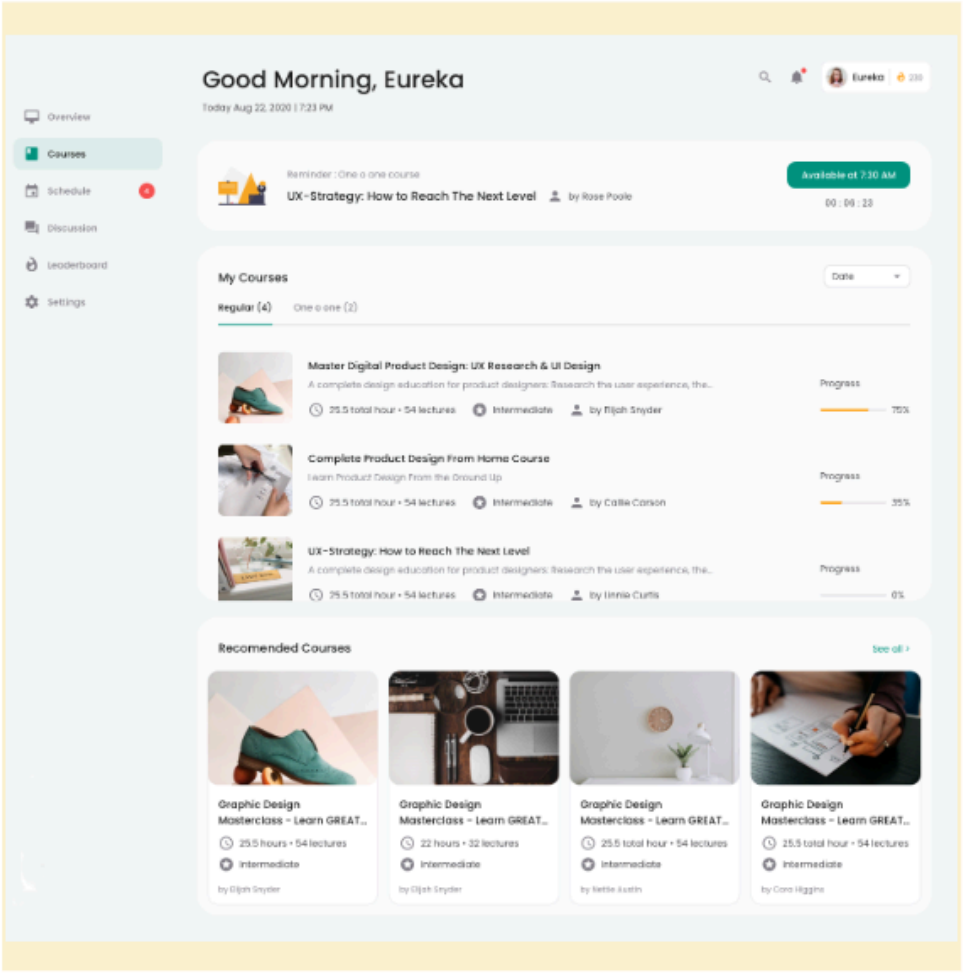
Password Reset Page



Video Course Page



Main Dashboard Page



BIBLIOGRAPHY

14. BIBLIOGRAPHY

WEBSITES

1. Educause
   - Website: [https://www.educause.edu/]

2. Online Learning Consortium (OLC)
   - Website: [https://onlinelearningconsortium.org/]

3. EDUCAUSE Review
   - Website: [https://er.educause.edu/]

4. Inside Higher Ed: Technology
   - Website: [https://www.insidehighered.com/technology]

5. Coursera Blog
   - Website: [https://blog.coursera.org/]

6. eLearning Industry
   - Website: [https://elearningindustry.com/]

7. Chronicle of Higher Education: Teaching
   - Website: [https://www.chronicle.com/section/Teaching/160/]

8. MERLOT (Multimedia Educational Resource for Learning and Online Teaching)
   - Website: [https://www.merlot.org/]

9. Canvas Community
   - Website: [https://community.canvaslms.com/]

10. GitHub Education Community
    - Website: [https://education.github.community/]