Software Design Document (SDD)

Batch: B2 **Roll No.:** 16010421059 **Experiment No.:** 7

Aim

To prepare Software Design Document (SDD)

Resources needed

Internet Explorer, LaTeX Editor

Theory

The SDD shows how your software will be structured to satisfy the requirements. It describes the software structure, software components, interfaces, and data necessary for the implementation phase. In a complete SDD, each requirement must be traceable to one or more components. An SDD is a representation or model of the software system to be created. The model should provide the precise design information needed for planning, analysis, and implementation of the software system. It should represent a partitioning of the system into components and describe the important properties and relationships between them.

SDD Template

1. Introduction

1.1 Design Overview

The QuirkIq website is designed to provide a user-friendly platform for users to share quirky and interesting content with a community of like-minded individuals. The design approach focuses on simplicity, intuitiveness, and engaging user experience.

Key Features:

- **User Registration and Authentication:** Users can sign up for an account on the platform, providing basic information such as username, email, and password. Account authentication ensures secure access to user-specific features and content.
- **Content Creation and Sharing:** Registered users can create and share various types of content, including text posts, images, videos, and links. The platform supports multimedia content to encourage diverse and engaging interactions.
- **Social Interaction:** Users can engage with content by liking, commenting, and sharing posts. The platform fosters a sense of community by enabling users to follow each other, form connections, and discover new content based on their interests.

- **Customization and Personalization:** The website offers customization options for user profiles, allowing users to personalize their experience and express their unique identities. Personalization algorithms recommend tailored content based on user preferences and interactions.
- **Moderation and Safety:** To maintain a positive and respectful community environment, the platform implements moderation tools and guidelines. Automated and manual moderation mechanisms detect and address inappropriate content, ensuring a safe and enjoyable experience for all users.

1.2 Requirements Traceability Matrix

Table 1: Requirements Traceability Matrix

Requirement ID	Description	Design Component
REQ-001	User registration	User Authentication Module
REQ-002	Content creation	Content Management System
REQ-003	Social interaction	Social Networking Module
REQ-004	Customization options	User Profile Management
REQ-005	Moderation tools	Content Moderation System

2. System Architectural Design

2.1 Chosen System Architecture

The QuirkIq E-Learning Platform follows a client-server architecture, with the following major components:

- **Client Side:**
  - Web Browser: The primary interface for users to access the platform. Users can view courses, interact with content, and access learning materials through a web browser.
  - Mobile App: Optionally, a mobile application can be developed to provide users with a more convenient way to access the platform on mobile devices.
- **Server Side:**
  - Application Server: Responsible for processing client requests, handling business logic, and serving dynamic content. This server-side component manages user authentication, course enrollment, content delivery, and interaction tracking.
  - Database Server: Stores and manages persistent data related to users, courses, learning materials, assessments, and user progress. The database server supports efficient retrieval and storage of structured data to support the platform's functionality.

2.2 Discussion of Alternative Designs

Alternative system architectures, such as a peer-to-peer network or a microservices architecture, were considered but not chosen for the following reasons:

- **Peer-to-Peer Network:** While peer-to-peer networks offer decentralized data distribution and scalability, they introduce complexity in managing data consistency and security. Given the centralized nature of user accounts and course materials, a client-server architecture provides better control and security.

- **Microservices Architecture:** While microservices offer flexibility and scalability by breaking down the application into smaller, independently deployable services, they introduce additional complexity in managing inter-service communication and data consistency. A monolithic architecture with clear separation of concerns between client and server components is more suitable for the initial development phase of the platform.

2.3 System Interface Description

The system interfaces include:

- **User Interface (UI):** The web-based user interface provides interactive access to course catalogs, learning materials, assessments, and user profiles. The UI is designed to be intuitive, responsive, and accessible across different devices and screen sizes.
- **Application Programming Interface (API):** The platform exposes RESTful APIs for integrating with external systems, such as authentication providers, content management systems, and learning management systems. These APIs allow external services to interact with user data, course materials, and learning analytics.
- **Database Interface:** The application server interacts with the database server through structured query language (SQL) or an object-relational mapping (ORM) framework. This interface handles data retrieval, storage, and manipulation operations to support platform functionality.

3. Detailed Description of Components

For each component, the following items should be described here as appropriate: responsibilities, constraints, composition, interactions, and resources. Use appropriate diagrams or other notation to describe your design.

4. User Interface Design

In this section describe the design of the user interface in detail.

4.1 Description of the User Interface

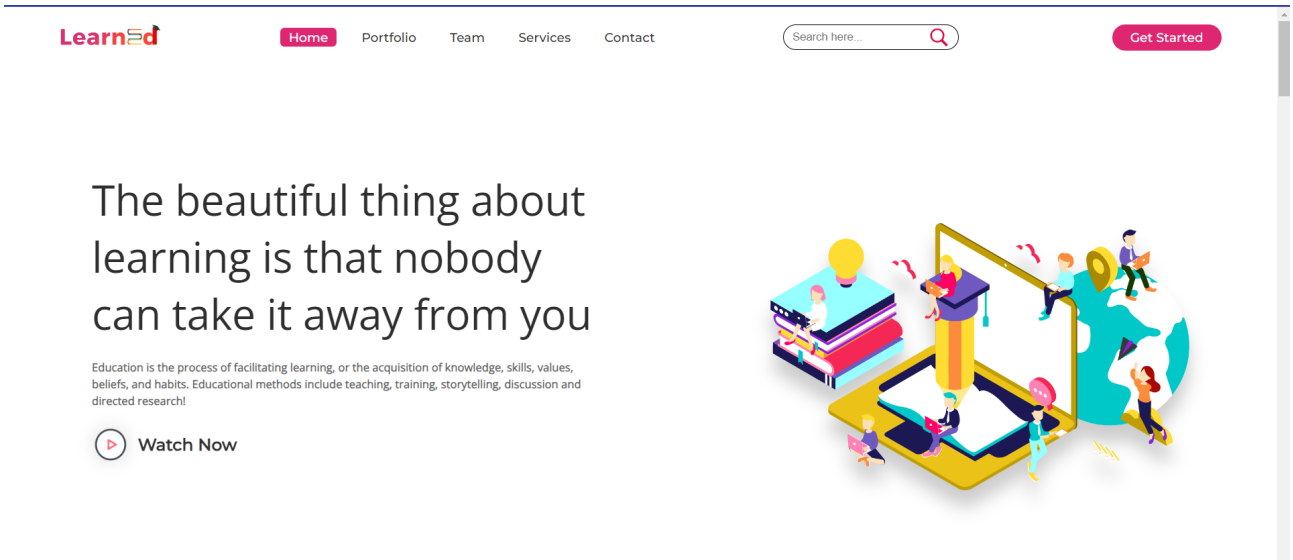

Figure 1: Home Page

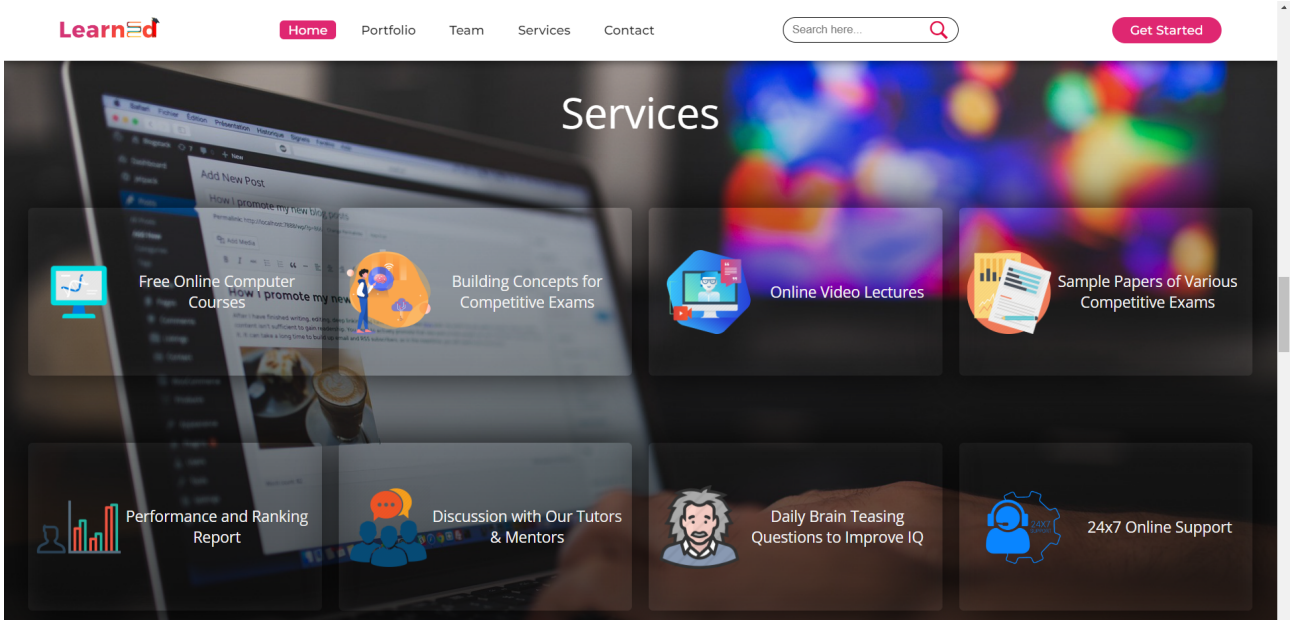

Figure 2: Services

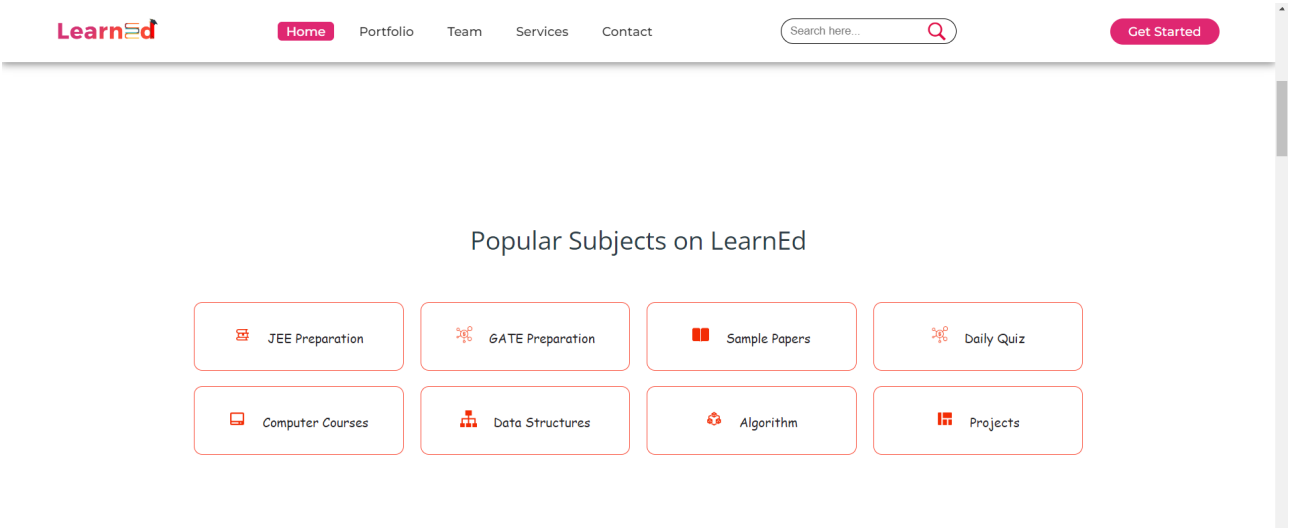

Figure 3: Facilities

2.3 System Interface Description

The system interfaces include:

- **User Interface (UI):** The web-based user interface provides interactive access to course catalogs, learning materials, assessments, and user profiles. The UI is designed to be intuitive, responsive, and accessible across different devices and screen sizes.
- **Application Programming Interface (API):** The platform exposes RESTful APIs for integrating with external systems, such as authentication providers, content management systems, and learning management systems. These APIs allow external services to interact with user data, course materials, and learning analytics.
- **Database Interface:** The application server interacts with the database server through structured query language (SQL) or an object-relational mapping (ORM) framework.

This interface handles data retrieval, storage, and manipulation operations to support platform functionality.

Use Case Specification using template

Table 2: Use Case Details

Use Case ID:	1
Use Case Name:	User Registration
Created By:	Chinmay Mhatre
Last Updated By:	Pragathi Rao
Date Created:	14th April 2024
Date Last Updated:	16th April 2024
Primary Actors:	Website User
Secondary Actors:	Database
Description:	Allow users to register an account on the quirkiq website.
Trigger:	User clicks on the "Sign Up" button.
Preconditions:	None
Postconditions:	User account is created in the database.
Normal Flow:	1. User clicks on "Sign Up" button. 2. User fills out the registration form with required details. 3. User submits the form. 4. System validates the input data. 5. System creates a new user account. 6. System redirects the user to the login page.
Alternative Flows:	- User provides invalid email format. - User provides weak password. - User account already exists.
Exceptions:	- System fails to create user account.
Includes:	None
Priority:	High
Frequency of Use:	High
Business Rules:	- User must provide a valid email address. - Password must be at least 8 characters long.
Special Requirements:	None
Open Issues:	None
Assumptions:	- Website is accessible and functional. - Database is operational.
Notes and Issues:	None

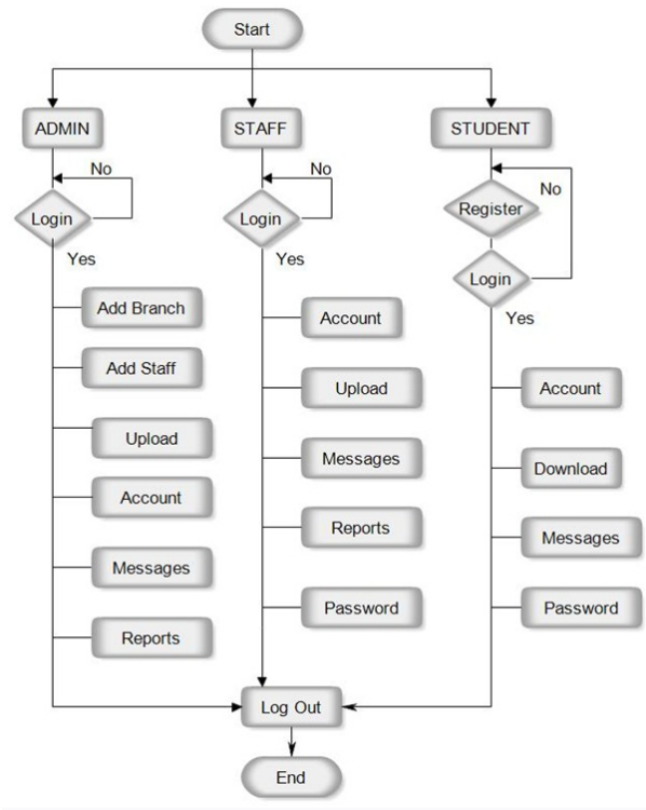

Figure 4: Workflow

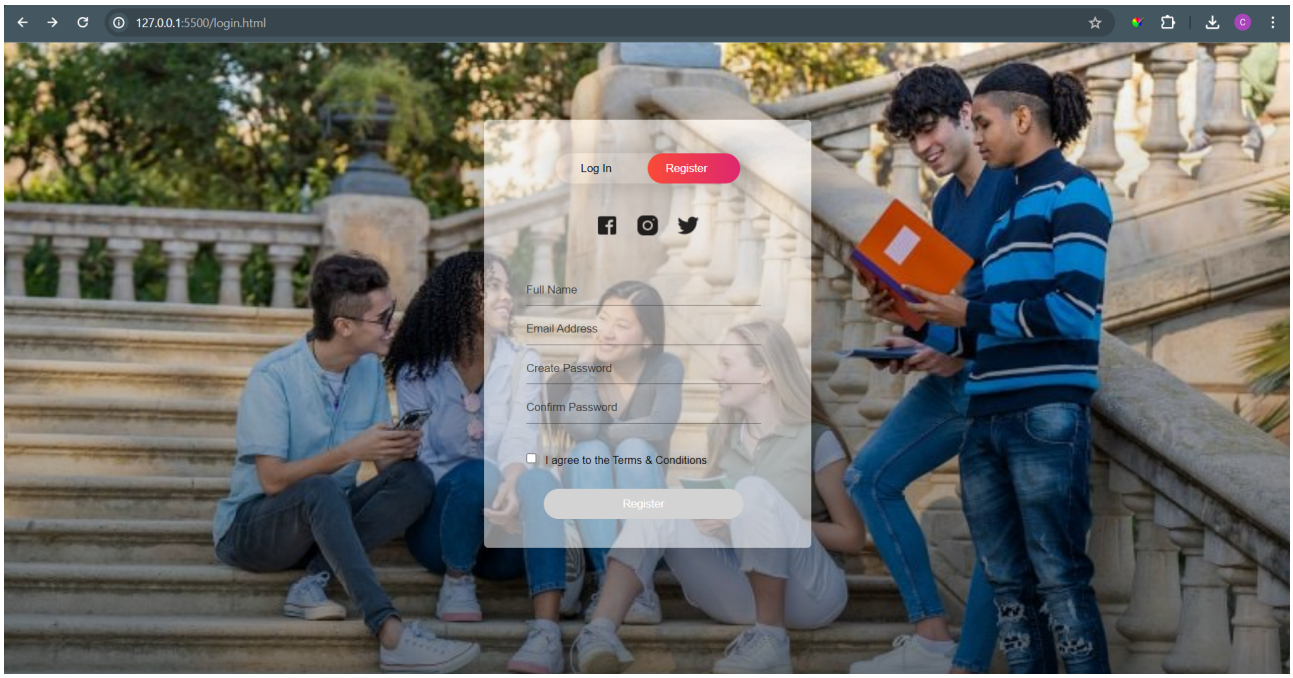

Figure 5: Signup

Questions

Explain Architecture design patterns and styles with examples.

- Architecture design patterns and styles are foundational concepts in software engineering that provide solutions to recurring design problems. They offer proven approaches for

structuring and organizing software systems to achieve specific goals such as modifiability, scalability, and maintainability. Here are explanations of some common architecture design patterns and styles along with examples:

- Layered Architecture:
  - Description: Layered architecture organizes the system into horizontal layers, where each layer represents a specific level of abstraction or responsibility. Each layer only communicates with adjacent layers, promoting separation of concerns and modularity.
  - Example: In a web application, you might have layers such as presentation layer (UI), business logic layer (services), and data access layer (database interaction).
- Model-View-Controller (MVC):
  - Description: MVC separates the application into three interconnected components: Model (data and business logic), View (presentation/UI), and Controller (handles user input and updates the model).
  - Example: A web framework like Django or Ruby on Rails implements MVC, where models represent database tables, views render HTML pages, and controllers handle HTTP requests.

Outcomes

CO3: Demonstrate requirements, modeling and design of a system

Conclusion

The Software Design Document (SDD) provides a structured blueprint for the development of the Quirkiq website. By outlining the system architecture, user interface design, and component specifications, we have established a clear path for implementation. With careful consideration of design principles and potential risks, we are well-equipped to proceed with confidence to the next phase of development.

Grade

AA / AB / BB / BC / CC / CD /DD