A CASE STUDY (IEEE Format)

Software Requirements Specification Document

Version 1.0

Gurukul
(an Educational Platform)

TABLE OF CONTENTS

1. Introduction

1.1 Purpose of this Document

The purpose of this SRS document is to provide a detailed overview of our software product, its parameters, and goals. This document describes the project's target audience and its user interface, hardware and software requirements. **It defines how our client, team and audience see the product.**

1.2 Scope of the Development Project

The objective of the Gurukul project is to develop a comprehensive Educational platform that supports the creation, consumption, and evaluation of educational content. The platform is intended to serve both students and instructors by providing tools for interactive learning and course management.

The scope of the Gurukul platform includes the following key functionalities:

1. User Authentication and Authorization

- The system will support secure login and registration processes for both students and instructors.
- It will include features for password recovery, email verification, and role-based access control.

2. Course Management:

- Instructors will be able to create, edit, and manage courses, including uploading multimedia content such as videos, documents, and quizzes.
- The system will allow for CRUD (Create, Read, Update, Delete) operations on course content and structure.

3. Content Delivery:

- The platform will provide a user-friendly interface for students to access course materials, stream videos, and interact with course content.
- The content will be delivered through a responsive web interface that is accessible on both desktop and mobile devices.

4. Payment Integration:

- The platform will integrate with a payment gateway (e.g., Razorpay) to facilitate the purchase of courses by students.
- It will handle payment transactions, generate receipts, and manage refunds.

5. API Development:

- RESTful APIs will be developed to enable communication between the front-end and back-end systems.
- These APIs will handle various operations such as user management, course management, and payment processing.

6. Data Storage and Management:

- MongoDB will be used to store user data, course information, and transaction records.
- The system will support data consistency, integrity, and regular backups.

7. Deployment and Hosting:

- The front-end of the platform will be hosted on Vercel, and the back-end will be hosted on cloud services like Render or Railway.
- Media files will be managed using Cloudinary, and the database will be hosted on MongoDB Atlas.

8. Testing and Quality Assurance:

- The system will undergo rigorous testing, including unit tests, integration tests, and user acceptance tests, to ensure reliability and performance.

9. Future Expansion:

- While the initial deployment will focus on core functionalities, future enhancements may include gamification, personalized learning paths, social learning features, and mobile application development.

The scope of the Gurukul project is limited to the development and deployment of the above functionalities. The platform will be designed with scalability in mind, allowing for future expansion and the addition of new features as the platform evolves.

1.3 Definitions, abbreviations and acronyms

Table 1: Definitions for most commonly used terms:

S.No.	Term	Definition
1.	MERN Stack	A technology stack used for developing web applications, consisting of MongoDB, Express.js, React.js, and Node.js.
2.	REST	Representational State Transfer. An architectural style for designing networked applications, using stateless communication, usually via HTTP.
3.	API	Application Programming Interface. A set of functions and protocols for building and integrating application software.
4.	CRUD	Create, Read, Update, Delete. The four basic functions of persistent storage in software applications.
5.	UI/UX	User Interface/User Experience. The design and functionality of the user-facing part of an application.
6.	JWT	JSON Web Token. A compact, URL-safe means of representing claims to be transferred between two parties, used for authentication and authorization.
7.	NoSQL	A non-relational database that allows for flexible data storage and retrieval, particularly useful for large amounts of unstructured or semi-structured data.
8.	CI/CD	Continuous Integration/Continuous Deployment. A set of practices that enable development teams to deliver code changes more frequently and reliably.
9.	OTP	One-Time Password. A password that is valid for only one login session or transaction, used to enhance security.
10.	Razorpay	An online payment gateway that allows businesses to accept, process, and disburse payments with ease.
11.	Cloudinary	A cloud-based service that provides media management solutions, including image and video storage, manipulation, and delivery.
12.	Redux	A predictable state container for JavaScript apps, commonly used with React for managing application state.
13.	Figma	A web-based interface design tool used for designing user interfaces and user experiences.

14.	MongoDB Atlas	A cloud database service for MongoDB, providing a fully-managed database with features like automatic scaling and backups.

Abbreviations

Table 2: Full form for most commonly used mnemonics

S.No.	Mnemonic	Full form
1.	API	Application Programming Interface
2.	REST	Representational State Transfer
3.	CRUD	Create, Read, Update, Delete
4.	UI/UX	User Interface/User Experience
5.	JWT	JSON Web Token
6.	NoSQL	Non-relational SQL
7.	MERN	MongoDB, Express.js, React.js, Node.js
8.	CI/CD	Continuous Integration/Continuous Deployment
9.	OTP	One-Time Password
10.	NPM	Node Package Manager
11.	IDE	Integrated Development Environment
12.	CSS	Cascading Style Sheets
13.	HTTP	Hypertext Transfer Protocol
14.	JSON	JavaScript Object Notation
15.	MVC	Model-View-Controller (a software design pattern)

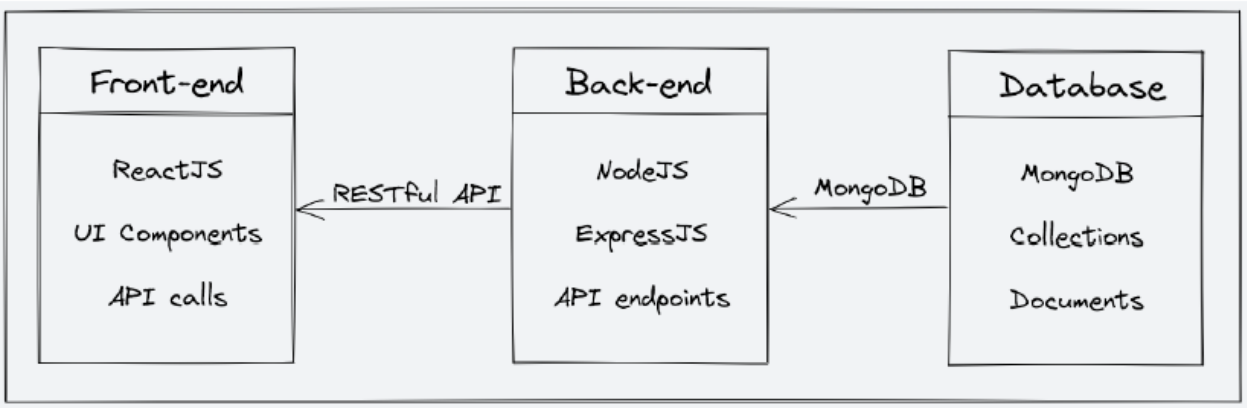
1.4 Overview

The following sections of this document provide a comprehensive outline of the Gurukul platform's requirements and design considerations. Section 2 presents an overall description, covering the product's perspective, functions, and user characteristics. It also discusses the general constraints, assumptions, and dependencies associated with the platform's development. Section 3 details the specific requirements, including external interface requirements and a breakdown of the functional and performance requirements. This section also describes the logical database structure, quality attributes, and any other pertinent requirements. Finally, the document concludes with a change history and a list of document approvers, ensuring that all stakeholders are aligned with the specified requirements.

2. Overall Description

2.1 Product Perspective

The Gurukul platform is designed with a modular architecture consisting of three main components: the Front-end, Back-end, and Database. The front-end is developed using ReactJS, handling user interactions and API calls to the back-end. The back-end is powered by NodeJS and ExpressJS, which manages business logic, processes API requests, and interacts with the MongoDB database for data storage and retrieval. The following diagram illustrates the system architecture:



This architecture allows for a scalable and maintainable system, where each component can be independently developed, tested, and deployed.

System Interfaces

- Client-Server Architecture: The platform follows a client-server architecture where the front-end (client) interacts with the back-end (server) via RESTful APIs. The front-end is built using ReactJS, while the back-end is developed using Node.js and Express.js.

- Database: The database is managed using MongoDB, a NoSQL database that stores all user data, course content, and transaction records. The database is hosted on MongoDB Atlas, ensuring scalability and reliability.

User Interfaces

- Front-End: The user interface (UI) is designed to be intuitive and responsive, accessible on both desktop and mobile devices. Students and instructors interact with the platform

through a series of web pages, including dashboards, course lists, and content delivery pages.

- Admin Interface (Future Scope): An administrative interface will be developed in future iterations, allowing administrators to manage users, courses, and platform settings.

Hardware Interfaces

- User Devices: The platform is accessible via standard web browsers on devices such as desktops, laptops, tablets, and smartphones. No special hardware is required for users to access the platform.

- Server Infrastructure: The platform's back-end and database components are hosted on cloud infrastructure. The front-end is hosted on Vercel, while the back-end and API services are hosted on cloud platforms like Render or Railway.

Software Interfaces

- APIs: The platform exposes RESTful APIs for communication between the front-end and back-end. These APIs handle various functions such as user authentication, course management, and payment processing.

- Third-Party Services: The platform integrates with third-party services, including Razorpay for payment processing and Cloudinary for media management. These integrations are handled through secure API calls.

Memory Constraints

- Front-End: As a web application, the front-end is designed to be lightweight and efficient, with minimal memory usage on the client side.

- Back-End: The back-end is optimized for handling large volumes of data and concurrent users. MongoDB's flexible schema allows for efficient storage and retrieval of data, minimizing memory constraints on the server side.

Operations

- User Operations: Users can perform a variety of operations, including creating accounts, enrolling in courses, consuming educational content, and providing feedback.

- Instructor Operations: Instructors can create and manage courses, upload content, and interact with students through the platform.

- Administrative Operations (Future Scope): Future development will include administrative operations such as user management, content moderation, and system monitoring.

Site Adaptation Requirements

- Scalability: The platform is designed to scale horizontally, with the ability to add more servers to handle increased traffic and user load.

- Customizability: The platform is designed with modularity in mind, allowing for easy adaptation and customization for different educational institutions or use cases.

2.2 Product Functions

The Gurukul platform provides a comprehensive set of functionalities aimed at enhancing the educational experience for both students and instructors. The key functions of the platform include:

1. User Management

- User Registration and Login: Users (students and instructors) can register on the platform and log in using their email and password. The system supports secure authentication through JWT (JSON Web Tokens).
- Profile Management: Users can view and update their personal information, such as name, email, and profile picture.

2. Course Management

- Course Creation and Management: Instructors can create new courses by providing details such as course title, description, and price. They can also update or delete existing courses.
- Content Upload: Instructors can upload various types of educational content, including videos, documents, quizzes, and other multimedia resources. The platform supports content management with features like markdown formatting and cloud storage for media files.
- Course Structuring: Instructors can organize their courses into modules and lessons, providing a structured learning path for students.

3. Content Consumption

- Course Enrollment: Students can browse the course catalog, view course details, and enroll in courses of interest. Enrolled courses are added to the student's dashboard for easy access.

- Multimedia Playback: Students can stream video lectures, read documents, and interact with quizzes directly within the platform. The content is delivered in a responsive format suitable for various devices.
- Progress Tracking: The platform tracks student progress through courses, allowing them to see which lessons have been completed and what remains.

4. Payment Processing

- Course Purchase: The platform integrates with Razorpay to handle secure payments for course enrollment. Students can add courses to their cart, proceed to checkout, and complete transactions.
- Transaction Management: The system manages transaction records, allowing users to view their purchase history and access receipts.

5. Rating and Feedback

- Course Rating: Students can rate courses on a scale of 1 to 5 stars after completing them. These ratings are displayed publicly to help other users make informed decisions.
- Feedback Submission: Students can provide detailed feedback on courses, which is shared with instructors to help them improve course quality.

6. API Integration

- RESTful APIs: The platform provides a set of APIs to support various operations, such as user authentication, course management, and payment processing. These APIs ensure seamless communication between the front-end and back-end components.

7. Search and Filter

- Course Search: Students can search for courses using keywords, filters, and categories to find content that matches their interests.
- Filtering Options: The platform offers filtering options such as course ratings, price, and difficulty level to help users narrow down their search results.

8. Administrative Tools (Future Scope)

- User Management: Administrators will have the ability to manage users, including approving or rejecting instructor applications, suspending accounts, and handling user queries.
- Platform Analytics: The platform will eventually include tools for monitoring platform usage, tracking key performance metrics, and generating reports for administrators.

2.3 User Characteristics

The Gurukul platform is designed to cater to a diverse group of users with varying roles, skills, and levels of experience. The primary user groups include students, instructors, and administrators, each with distinct characteristics and requirements.

1. **Students**

- Background: Students using the platform can range from high school learners to university students and professionals seeking to upgrade their skills. They may have varying levels of familiarity with online learning platforms.
- Technical Expertise: Most students are expected to have basic computer literacy, including the ability to navigate websites, stream videos, and interact with online forms.
- Learning Preferences: Students may have different learning styles, such as visual, auditory, or kinesthetic. The platform provides a mix of multimedia content to accommodate these preferences.
- Motivation: Students typically use the platform to acquire new knowledge, improve existing skills, or achieve academic or professional goals. They are motivated by the desire for personal growth, career advancement, or academic success.

2. **Instructors**

- Background: Instructors on Gurukul are educators, industry professionals, or subject matter experts who create and deliver educational content. They may have varying degrees of experience with digital teaching tools.
- Technical Expertise: Instructors are expected to have a moderate level of technical proficiency, including the ability to create and upload content, manage course structures, and use basic online tools for interacting with students.
- Content Creation: Instructors are responsible for developing course materials, including video lectures, documents, quizzes, and other interactive content. They should be comfortable with creating digital content and using the platform's course management tools.
- Engagement: Instructors are motivated by the opportunity to share knowledge, build a reputation in their field, and connect with a global audience of learners. They are also interested in receiving feedback to improve their courses.

2. **Administrators (Future Scope)**

- Background: Administrators are responsible for overseeing the platform's operations, managing users, and ensuring that content meets quality standards. They may have experience in education management or technical support.

- Technical Expertise: Administrators are expected to have a higher level of technical proficiency, including the ability to manage user accounts, monitor platform performance, and generate analytical reports.
- Responsibilities: Their primary responsibilities include approving or rejecting course submissions, handling user queries and complaints, managing platform security, and ensuring that the platform operates smoothly.
- Decision-Making: Administrators are motivated by the need to maintain the platform's quality, user satisfaction, and operational efficiency.

3. General Characteristics

- Device Usage: All user groups are expected to access the platform via various devices, including desktops, laptops, tablets, and smartphones. The platform is designed to be responsive and user-friendly across all these devices.
- Internet Access: Users are expected to have a stable internet connection to access the platform, stream content, and participate in interactive activities. The platform's design takes into account varying levels of internet bandwidth.

2.4 General Constraints, Assumptions and Dependencies

The following list presents the constraints, assumptions, dependencies or guidelines

2.4.1 General Constraints

- **Performance Constraints**: The platform must handle concurrent users effectively, with a target response time of less than 2 seconds for most operations. The system must be optimized to perform well under typical user loads without significant degradation.
- **Scalability Constraints**: The system should be able to scale horizontally to accommodate increasing numbers of users, courses, and content without requiring major architectural changes.
- **Security Constraints**: Sensitive data such as user credentials and payment information must be securely stored and transmitted. The platform must comply with relevant data protection regulations (e.g., GDPR).
- **Browser Compatibility**: The platform should be compatible with major modern web browsers (Chrome, Firefox, Safari, Edge) and responsive across different screen sizes (desktops, tablets, smartphones).

- **Third-Party Services**: The platform relies on third-party services like Razorpay for payments, Cloudinary for media management, and MongoDB Atlas for database hosting. Any disruptions in these services could affect the platform's functionality.

2.4.2 Assumptions

- **User Technical Proficiency**: It is assumed that users (students, instructors, and admins) have basic computer literacy and are familiar with common web-based platforms.
- **Stable Internet Connection**: It is assumed that users will have a stable internet connection to access the platform, as it is a web-based application.
- **Regular Maintenance**: It is assumed that the platform will undergo regular maintenance and updates to ensure security patches, bug fixes, and feature improvements are applied in a timely manner.
- **Content Availability**: It is assumed that instructors will regularly create and update course content, keeping the platform's offerings current and relevant.

2.4.3 Dependencies

- **Technology Stack:** The platform is dependent on the MERN stack (MongoDB, Express.js, React.js, Node.js) for its core functionalities. Any changes or updates to these technologies could impact the platform's development or performance.
- **API Integrations:** The platform depends on various API integrations, such as payment gateways (Razorpay), media storage (Cloudinary), and email services (e.g., for OTPs and notifications). The availability and performance of these APIs are crucial for the platform's operation.
- **Cloud Hosting Services:** The platform’s back-end and database are hosted on cloud services like Render or Railway for Node.js applications and MongoDB Atlas for the database. The reliability of these services is critical for the platform's uptime and performance.
- **Legal and Regulatory Compliance:** The platform must adhere to local and international laws and regulations regarding online education, data privacy, and payment processing.

This section clearly outlines the boundaries within which the Gurukul platform will operate, as well as the assumptions made during the design and development process. It also highlights key dependencies that could impact the platform's success.

2.5 Apportioning of requirements

The Gurukul platform will be developed in multiple phases, with each phase delivering a set of prioritized features. The requirements have been apportioned based on their importance, complexity, and the need for core functionality in the initial release.

2.5.1 Initial Release (Version 1.0)

The following requirements will be implemented in the initial release:

- **User Authentication and Authorization**: Implementation of secure user sign-up, login, and session management using JWTs. Support for both student and instructor roles.
- **Course Creation and Management:** Instructors will be able to create, update, and delete courses, including uploading course content (videos, documents, etc.).
- **Course Enrollment and Consumption:** Students will be able to browse available courses, enroll, and access course content, including videos, documents, and other media.
- **Basic Payment Integration:** Integration with Razorpay to handle course purchases and enrollment fees.
- **Basic User Profile Management**: Users will be able to manage their profiles, including updating personal information and viewing enrolled courses.
- **API Development:** Core API endpoints for user authentication, course management, and course consumption will be implemented.

2.5.2 Future Enhancements (Planned for Subsequent Releases)

The following requirements are planned for future releases based on user feedback and additional needs:

- **Advanced Gamification Features:** Implementation of badges, points, and leaderboards to increase user engagement and motivation.
- **Personalized Learning Paths:** Development of algorithms to provide personalized course recommendations and learning paths based on user preferences and progress.
- **Social Learning Features:** Integration of discussion forums, peer-to-peer feedback, and collaborative projects to enhance community interaction and learning.
- **Mobile Application:** Development of a mobile app for iOS and Android to allow users to access Gurukul on the go.
- **Advanced Analytics and Insights**: Implementation of detailed analytics for instructors and administrators, including course performance metrics, student progress tracking, and more.
- **Machine Learning Integration:** Introduction of machine learning algorithms to provide personalized content recommendations and predictive analytics.

- **Virtual Reality (VR) and Augmented Reality (AR) Integration:** Exploration of VR/AR technologies to create immersive learning experiences for certain types of courses.
- **Admin Dashboard and Management Tools**: Development of an admin panel with comprehensive management tools for instructors, courses, and users.

2.5.3 Deferred Requirements

The following requirements have been identified as lower priority and will be considered for implementation in later versions:

- **Support for Multi-Language Content**: Allowing content to be available in multiple languages to cater to a more global audience.
- **Integration with External Learning Management Systems (LMS):** Connecting Gurukul with other LMS platforms to broaden content availability and interoperability.
- **Advanced Payment Features:** Support for additional payment methods and more complex pricing models (e.g., subscriptions, installment payments).
- **Enhanced Security Features:** Implementation of advanced security features such as biometric authentication and two-factor authentication.

This apportioning of requirements helps manage the project scope, ensuring that essential features are delivered in the initial release while providing a roadmap for future enhancements based on user needs and feedback.

3. Specific Requirements

3.1 External Interface Requirements

3.2 Detailed Description of Functional Requirements

Table 3: Template for describing functional requirements

Purpose	A description of the functional requirements and its reasons
Inputs	What are the inputs; in what form will they arrive; from what sources can the inputs come; what are the legal domains of each input.
Processing	Describes the outcome rather than the implementation; includes any validity checks on the data, exact timing of operation (if needed), how to handle unexpected or abnormal situations

Output	The form, shape, destination and volume of output; output timing; range of parameters in the output; unit of measure of the output; process by which output is stored or destroyed; process for handling error message produced as output.

3.2.1 Functional Requirements for Student Welcome Screen

Table 4: Functional Requirements for Student Welcome Screen

Purpose	The student interface provides students with access to educational content, course management, and personal account management.
Inputs	Students interact with the platform through various UI components. They can select courses, add items to their wishlist, complete purchases, and view course content.
Processing	The platform processes student inputs to retrieve and display course information, manage wishlist items, process payments, and update personal information.
Outputs	The outputs include personalized content displays, such as the course list, course details, and user profile information. For example, when a student selects a course, the platform will display detailed information about the course, including videos and other materials.

3.2.2 Functional Requirements for Instructor Interface

Table 5: Functional Requirements for Instructor Interface

Purpose	The instructor interface allows instructors to manage courses, view insights, and interact with students.
Inputs	Instructors can create, update, or delete courses, manage course content, and view course performance metrics.
Processing	The platform processes instructor inputs to update course details, manage media content, and generate insights based on course interactions.

Outputs	Outputs include updated course lists, performance metrics, and student feedback. For example, an instructor can view detailed insights into course engagement, including the number of views and ratings.

3.2.3 Functional Requirements for Administrator Interface (Future Scope)

Table 6: Functional Requirements for Student cum Staff Welcome Screen

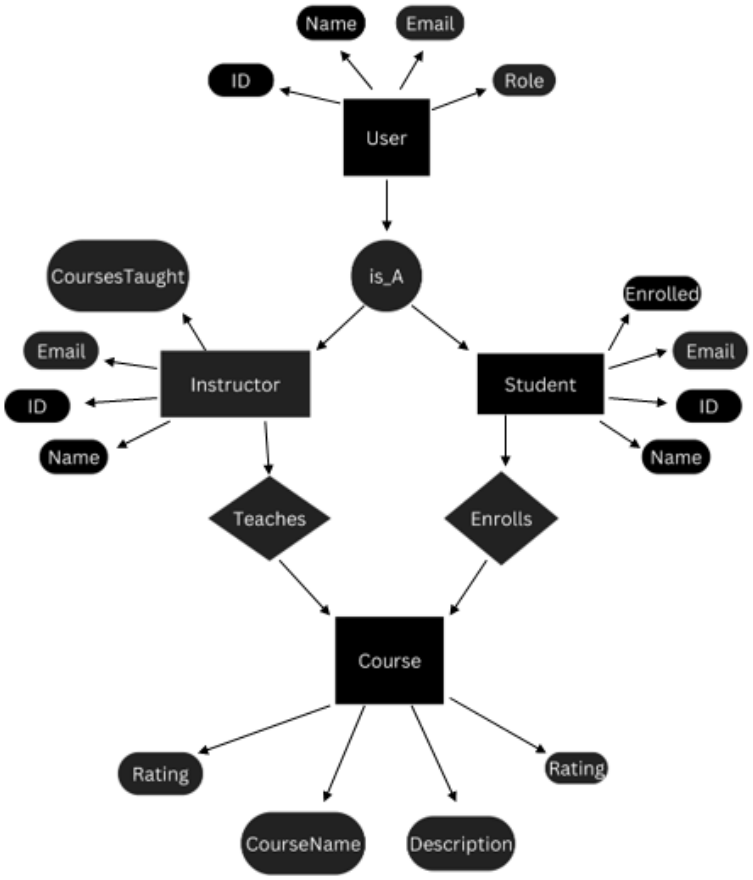
Purpose	The administrator interface is designed for platform management, including user and course management, and platform-wide insights.
Inputs	Administrators manage user accounts, oversee instructor activities, and monitor platform metrics.
Processing	The platform processes these inputs to maintain the overall health and functionality of the platform, ensuring security and smooth operation.
Output	The outputs are detailed reports on platform usage, user activity, and financial data. For example, administrators can generate reports on the number of active users, total revenue, and course performance.

3.3 Performance Requirements

- **System Load**: The system must handle user interactions efficiently. It should support multiple simultaneous users without performance degradation.
- **Response Time**: The platform should provide quick responses to user inputs. For normal operations, 95% of the transactions should be processed in less than 5 seconds.
- **Data Handling**: The platform should effectively handle and process textual information, accommodating varying amounts of data depending on the user's interaction

3.4 Logical Database Requirements

E-R diagram for entire system



3.5 Quality Attributes

The Gurukul platform is designed with several quality attributes to ensure a high level of performance, usability, and reliability. These attributes include:

1. **Performance**: The platform must load quickly and efficiently, with 95% of transactions being processed in less than 5 seconds under normal conditions. This is crucial to maintaining a smooth user experience.

2. **Scalability**: Gurukul is built to handle an increasing number of users and data over time without degradation in performance. The use of scalable cloud infrastructure ensures that the platform can grow as needed.

3. **Security**: The platform uses secure password hashing (Bcrypt) and JWTs for secure authentication. This helps protect user data and maintain the integrity of the platform.

4. **Usability**: The user interface is designed to be intuitive and user-friendly, ensuring that users of all technical levels can navigate and use the platform effectively.

5. **Reliability**: The platform ensures high availability and uptime, leveraging cloud services like Vercel and MongoDB Atlas for redundancy and failover capabilities.

6. **Maintainability**: The platform's codebase is modular and well-documented, making it easier to maintain and update over time. The use of modern frameworks like ReactJS and NodeJS contributes to easier maintenance and scaling.

3.6 Other Requirements

Not at this time

4. Change History

082004	Version 1.0 - Initial Release

5. Document Approvers

SRS for Gurukul (an Educational platform) approved by:

_____

Name: Dr. Harkiran Kaur

Designation: Assistant Professor

Date: 30/08/2024