Software Requirements Specification (SRS) for College Website

1. Introduction

1.1 Purpose

The purpose of this document is to describe the requirements for a college website that serves as a comprehensive platform for managing and displaying information about students, teachers, courses, projects, and other relevant college activities. The website will be used by students, faculty, and administrative staff.

1.2 Scope

This SRS document covers the functionalities required for a college website, including student profiles, teacher profiles, attendance tracking, project showcases, social media integration, course-related problem management, and additional features to enhance user experience.

1.3 Definitions, Acronyms, and Abbreviations

SRS: Software Requirements Specification

UI: User Interface

UX: User Experience

1.4 References

Web Development Standards

College Administration Policies

2. Overall Description

2.1 Product Perspective

The college website is an independent web application designed to provide easy access to information and management tools for students, teachers, and administrators. It will integrate various modules to ensure seamless functionality and user experience.

2.2 Product Functions

The main functions of the college website include:

Student Profiles

Teacher Profiles

Attendance Management

Student Projects

Social Media Integration

Daily Course-related Problems

Course Management

Event Calendar

News and Announcements

Resource Management

User Authentication and Authorization

Feedback System

2.3 User Classes and Characteristics

Students: Access personal profiles, course materials, submit projects, view attendance, and interact with peers.

Teachers: Manage their profiles, upload course materials, track attendance, review projects, and address student queries.

Administrators: Oversee the website operations, manage user accounts, update news and announcements, and ensure data security.

2.4 Operating Environment

Web Server: Apache/Nginx

Database: MySQL/PostgreSQL

Client: Modern web browsers (Chrome, Firefox, Edge)

2.5 Design and Implementation Constraints

Compliance with web accessibility standards.

Secure data transmission and storage.

Scalable architecture to handle increased user load.

2.6 User Documentation

User manuals and online help resources will be provided for students, teachers, and administrators.

2.7 Assumptions and Dependencies

Users have basic internet and computer skills.

Reliable internet connection.

3. Specific Requirements

3.1 Functional Requirements

3.1.1 Student Profiles

Description: Each student will have a personal profile displaying their information, academic progress, and other relevant details.

Inputs: Student personal details, academic records.

Outputs: Displayed student profile information.

3.1.2 Teacher Profiles

Description: Each teacher will have a profile displaying their professional information, courses taught, and contact details.

Inputs: Teacher personal details, professional records.

Outputs: Displayed teacher profile information.

3.1.3 Attendance Management

Description: Track and display attendance records for students.

Inputs: Daily attendance data.

Outputs: Attendance reports for students and teachers.

3.1.4 Student Projects

Description: Showcase students’ projects with descriptions, media, and feedback options.

Inputs: Project details, media files.

Outputs: Project showcase pages.

3.1.5 Social Media Integration

Description: Link student and teacher profiles to their social media accounts.

Inputs: Social media links.

Outputs: Social media links displayed on profiles.

3.1.6 Daily Course-related Problems

Description: A section for students to report and discuss daily problems related to their courses.

Inputs: Problem descriptions, comments.

Outputs: Problem and discussion threads.

3.1.7 Course Management

Description: Manage and display course information, schedules, and materials.

Inputs: Course details, schedules, materials.

Outputs: Course pages with relevant information.

3.1.8 Event Calendar

Description: Display upcoming events, deadlines, and important dates.

Inputs: Event details.

Outputs: Calendar with event listings.

3.1.9 News and Announcements

Description: Post and manage news articles and announcements relevant to the college community.

Inputs: News and announcement content.

Outputs: News and announcement pages.

3.1.10 Resource Management

Description: Manage and provide access to educational resources like e-books, research papers, and lecture notes.

Inputs: Resource files and metadata.

Outputs: Resource library accessible to users.

3.1.11 User Authentication and Authorization

Description: Secure login system to authenticate users and control access to different sections based on roles.

Inputs: User credentials.

Outputs: Access granted or denied.

3.1.12 Feedback System

Description: Collect feedback from students and teachers about courses, projects, and the website.

Inputs: Feedback forms.

Outputs: Feedback reports.

3.2 Non-functional Requirements

3.2.1 Performance Requirements

The website should load within 3 seconds for users with a standard internet connection.

The system should handle up to 1000 concurrent users without performance degradation.

3.2.2 Security Requirements

Data encryption for sensitive information.

Regular security audits and updates.

3.2.3 Usability Requirements

Intuitive UI/UX design for ease of navigation.

Responsive design for compatibility with various devices.

3.2.4 Maintainability Requirements

Modular code structure for easy maintenance and updates.

Comprehensive documentation for developers.

3.2.5 Reliability Requirements

99.9% uptime.

Backup and recovery procedures in place.

4. External Interface Requirements

4.1 User Interfaces

User-friendly web interface accessible via major browsers.

Mobile-friendly design.

4.2 Hardware Interfaces

Server hardware to host the website and database.

4.3 Software Interfaces

Integration with social media APIs.

Database management system for data storage.

4.4 Communication Interfaces

Secure HTTP/HTTPS protocols for data transmission.

5. Other Requirements

5.1 Legal and Regulatory Requirements

Compliance with data protection laws (e.g., GDPR).

5.2 Ethical Requirements

Ensuring user privacy and data confidentiality.

5.3 Standards Compliance

Adherence to web development standards and best practices.

1. Front-End Development:

HTML5: For structuring the web pages.

CSS3: For styling the web pages.

JavaScript: For adding interactivity.

Front-End Frameworks: React.js or Angular for building a dynamic and responsive user interface.

Responsive Design: Bootstrap or Tailwind CSS for ensuring the website is mobile-friendly.

2. Back-End Development:

Server-Side Language: Node.js (with Express.js) or Python (with Django or Flask).

Database Management System:

Relational Database: PostgreSQL or MySQL for structured data.

NoSQL Database: MongoDB for flexible, JSON-like data storage.

APIs: RESTful APIs or GraphQL for communication between the front-end and back-end.

Authentication and Authorization: JWT (JSON Web Tokens) or OAuth for secure login and access control.

3. Additional Tools and Technologies:

Version Control: Git for version control and collaboration.

Deployment and Hosting: AWS, Google Cloud, or Azure for hosting the web application.

Alternatively, Heroku for a more straightforward deployment process.

Continuous Integration/Continuous Deployment (CI/CD): Jenkins, Travis CI, or GitHub Actions for automated testing and deployment.

Containerization: Docker for containerizing the application to ensure consistency across development and production environments.

Security: HTTPS, SSL certificates, and regular security audits.

4. Project Management and Communication:

Project Management Tools: JIRA, Trello, or Asana for managing tasks and tracking progress.

Communication Tools: Slack or Microsoft Teams for team communication.