Software Requirement Specification

for

CAMPUS CONNECT

Version 1.0

Prepared by

Josna John S6 CSE B **42**

Fathima Nazar S6 CSE B **11**

Merin P Wilson S6 CSEB **59**

Harinandana K Biju S6 CSEB **26**

08-01-2025

Table of Contents

Revision History

Name	***Date***	***Reason for Change***	***Version***

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to define the features and functionality of the **Campus Connect** website, a platform designed to serve students for communication, collaboration, and resource sharing within the campus. The website aims to provide an efficient means of accessing campus information, joining discussions, and receiving important updates, all managed by an admin and faculties.

1.2 Document Conventions

- **Font**: Calibri(Body)
- **Title Font Size**: 11
- **Content Font Size**: 11
- **Line Height**: 1.5

1.3 Intended Audience and Reading Suggestions

- **Students**: Use the website to communicate, access announcements and resources, and participate in events.
- **Faculty**: Manage faculty-specific events and provide additional resources.
- **Admin**: Manage events, announcements, and resources, and moderate student interaction on the platform.

1.4 Product Scope

The **Campus Connect Website** is a digital platform developed exclusively for students. It will enable students to access and share important information, participate in group chats, stay informed about upcoming events. The platform will also allow admins to manage all student interactions, create events, and share resources.

1.5 References

- React.js Documentation
- Firebase Documentation
- MongoDB Documentation
- Node.js Documentation

2. Overall Description

2.1 Product Perspective

Campus Connect is designed to replace traditional methods of campus communication, such as noticeboards and emails, with a centralized digital platform. It provides students and faculty with an easy way to stay updated on campus events, communicate with peers, and access important resources. The platform will have admin and faculties, each with distinct roles and responsibilities to ensure smooth operation.

2.2 Product Functions

- **Group Chats**: Real-time chat for students to engage in discussions about classes, clubs, and campus activities.
- **Announcements & Notifications**: Admins can send real-time updates for events, schedules, or urgent news.
- **Event Management**: Students can view upcoming campus events.
- **Resource Sharing**: Admin can upload important resources like PDFs, links, and documents for student access.
- **AI Chatbot**: A virtual assistant to answer common student questions and provide help.

2.3 User Classes and Characteristics

- **Students**:
  - Request to join group chats and events.
  - View for campus events.
  - Get the resources uploaded by admins.
- **Admin**:
  - Manage events, announcements, group chats, and resources.
  - Moderate student interaction and manage platform content.
- **Faculties**:
  - Provide academic resources and urgent announcements.

2.4 Operating Environment

- **Frontend**: React.js for building the user interface.

- **Backend**: Node.js with Express.js for handling server-side functionality.
- **Database**: MongoDB for storing student data, event details and resources.
- **Real-Time Communication**: Firebase for chat and real-time notifications.
- **Authentication**: Firebase Authentication for secure student login.

2.5 Design and Implementation Constraints

- The platform must be responsive to work seamlessly across all devices (desktop, mobile, and tablet).
- It will be hosted on cloud platforms (AWS or Google Cloud) to ensure scalability and availability.
- Reliable internet access is required for real-time features (chat and notifications) to function properly.

2.6 User Documentation

- **Installation Guide**: Step-by-step instructions for setting up the website and the required tools.
- **User Guide**: A manual for students,faculties and admin, detailing how to use the platform's features effectively.

3. External Interface Requirements

3.1 User Interfaces

- **Frontend**:
  - The website interface will be developed using React.js for a smooth, interactive user experience.
  - Users will interact with the platform via web browsers.
  - Students and faculty access their respective dashboards.
  - Admins have dedicated admin panel.
  - Students will have access to a dashboard where they can see events, participate in chats, and view resources.
- **Backend**:
  - Node.js will handle the server-side logic for student authentication, event creation, and data management.
  - Firebase will be integrated for real-time messaging and notifications.

3.2 Hardware Interfaces

- The platform is web-based, accessible through a computer or mobile device (smartphones, tablets).

3.3 Software Interfaces

- **Operating Systems**: Windows, Linux, macOS, iOS, and Android (for mobile access).
- **Database**: MongoDB will store all student and event data.
- **Real-Time Communication**: Firebase for instant notifications and chat functionality.

4. System Features

4.1 Group Chats

- **Description**: Real-time communication channels for students to discuss topics like classes, events, and campus activities.
- **Priority**: High
- **Functional Requirements**:
  - Students can create and join group chats.
  - Admin can moderate group chats if necessary.
  - Firebase will handle the real-time messaging service.

4.2 Announcements & Notifications

- **Description**: Admins will send real-time notifications for upcoming events, class schedules, and important updates.
- **Priority**: High
- **Functional Requirements**:
  - Admin can send push notifications to all students or specific groups.
  - Students will receive real-time updates about announcements.

4.3 Event Management

- **Description**: The admin will manage campus events, and students can see event details, and receive reminders.
- **Priority**: Medium
- **Functional Requirements**:
  - Admins can create, edit, and delete events.
  - Students can join events directly from the platform.

4.4 Resource Sharing

- **Description**: Faculties can upload resources such as PDFs, links, and event flyers for students to access.
- **Priority**: Medium
- **Functional Requirements**:
    - Faculties upload and organize resources.
    - Students can view or download resources from the platform.

4.5 AI Chatbot

- **Description**: A chatbot that answers common questions about campus activities, schedules, and events.
- **Priority**: Low
- **Functional Requirements**:
    - AI-powered assistant to respond to student queries.
    - Can provide information like event details, class schedules, or general campus information.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The platform must handle up to 1000 concurrent users with minimal latency.
- Real-time features such as chat and notifications should have a delay of no more than 5 seconds.

5.2 Security Requirements

- **Authentication**: Firebase Authentication will be used to ensure secure logins.
- **Data Security**: All sensitive data will be encrypted in transit and at rest.
- **Access Control**: Role-based access will be implemented, allowing only the student admin and faculty admin to manage content.

6. Other Requirements

6.1 Data Requirements

- All student data and event detailswill be stored in a MongoDB database.

- Periodic backups will be conducted to prevent data loss.

Appendix A: Glossary

- **Firebase**: A platform providing tools for real-time communication, notifications, and authentication.

Appendix B: Analysis Models

- Use case diagrams and entity relationship diagrams will be developed.

Appendix C: To Be Determined List

- Development timeline for mobile app version.
- Scalability testing for future expansion of the platform.

This document serves as the blueprint for the **Campus Connect Website**, providing clear guidance for developers and stakeholders to implement the platform.