Software Requirements Specification

for

Mobile application for Social Networking of MU Students

Version <0.1>

Prepared by

Group Name: Team 1

|  |  |  |
| --- | --- | --- |
| Anunay Korupolu | se22uari019 |  |
| Chatur Varma Inampudi | se22uari034 |  |
| Krushik Teja Thati | se22uari078 |  |
| Madala Venkata Bhargav | se22uari086 |  |

|  |  |
| --- | --- |
| Instructor: | Prof. Avinash Arun Chauhan |
| Course: | Software Engineering |
| Lab Section: | *AI* |
| Teaching Assistant: | ***Nartkannai K*** |
| Date: | 22/02/2025 |

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 Operating Environment 2

2.4 Design and Implementation Constraints 2

2.5 User Documentation 2

2.6 Assumptions and Dependencies 3

2.7 Functional Requirements Specification

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

4. System Features 4

4.1 Functional Requirements 4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 4

5.3 Security Requirements 4

5.4 Software Quality Attributes 5

6. Other Requirements 5

Appendix A: Glossary 5

# Introduction

## Purpose

The purpose of this document is to outline the functional and non-functional requirements for the development of MU-Connect, a dedicated social networking mobile application for registered students of Mahindra University (MU). The app will facilitate real-time communication, media sharing, and university-specific collaboration tools, ensuring privacy, security, and ease of use.

## Document Conventions

Bold: Key terms and headings.

Italics: Emphasis or examples.

Monospace: Code snippets or technical terms.

Acronyms: Defined upon first use (e.g., SRS - Software Requirements Specification).

## Intended Audience and Reading Suggestions

Intended Audience:

Developers: To implement the app based on technical requirements.

Project Managers: To oversee timelines, resources, and deliverables.

Testers: To validate functionality and performance.

Documentation Writers: To create user manuals and technical guides.

Stakeholders (MU faculty): To review scope, features, and compliance.

End-Users (Students): To understand app functionality and usage.

Reading Suggestions:

All Readers: Start with Section 1 (Introduction) for an overview.

Developers: Focus on Sections 2, 3, and 4 for technical and functional details.

Testers: Refer to Sections 3 and 6 for functional and non-functional requirements.

Stakeholders: Review Sections 1 and 5 for scope and system features.

End-Users: Read Section 1 and the User Manual (post-development).

## Product Scope

The MU-Connect app will provide a secure and private platform for MU students to:

Communicate via real-time messaging (one-on-one and group chats).

Share media (images, videos, documents).

Access university-specific features (resource sharing, study groups).

Customize profiles and privacy settings.

## References

IEEE SRS Template.

Flutter Documentation: <https://flutter.dev/docs>.

Supabase Documentation: <https://supabase.com/docs>

PostGreSQL Documentation: <https://www.postgresql.org/docs/>

TypeScript Documentation: <https://www.typescriptlang.org/docs/>

# Overall Description

## Product Perspective

The MU-Connect app is a standalone mobile application designed to enhance communication and collaboration among MU students. It will integrate with Supabase for backend services and will be available on Android and iOS platforms.

## Product Functions

User Authentication: Login/Signup using university email.

Real-Time Messaging: One-on-one and group chats with text, emojis, and media.

Media Sharing: Upload and share images, videos, and documents.

User Profiles: Profile pictures, status, and privacy settings.

Notifications: Push notifications for new messages and updates.

University Features: Event announcements, study groups, and academic resources.

## Operating Environment

Platforms: Android (5.0 and above) and iOS (12.0 and above).

Devices: Smartphones and tablets.

Backend: Supabase (Authentication, Storage, Cloud Functions).

## Design and Implementation Constraints

Technology Stack: Flutter for frontend, Supabase for backend

Scalability: Support for up to 100 concurrent users. (on free package)

## User Documentation

User Manual: Step-by-step guide for app usage.

Admin Guide: Instructions for managing users and content.

API Documentation: For future integrations.

## Assumptions and Dependencies

Assumptions:

All users will have valid university email IDs.

Supabase services will remain available and scalable.

Dependencies:

Android Studio, VSCode, Flutter SDK, Supabase services, PostGreSQL, TypeScript and third-party libraries.

## Functional Requirement Specifications

1 User Authentication

FR-1: Users can register using their university email.

FR-2: Users can log in using email/password.

2 Real-Time Messaging

FR-3: Users can send and receive text messages in real-time.

FR-4: Users can create and participate in group chats.

3 Media Sharing

FR-5: Users can upload and share images, videos, and documents.

FR-6: Media files are compressed to reduce storage and bandwidth usage.

4 User Profiles

FR-7: Users can customize their profiles (profile picture, status).

FR-8: Users can set privacy preferences (profile visibility).

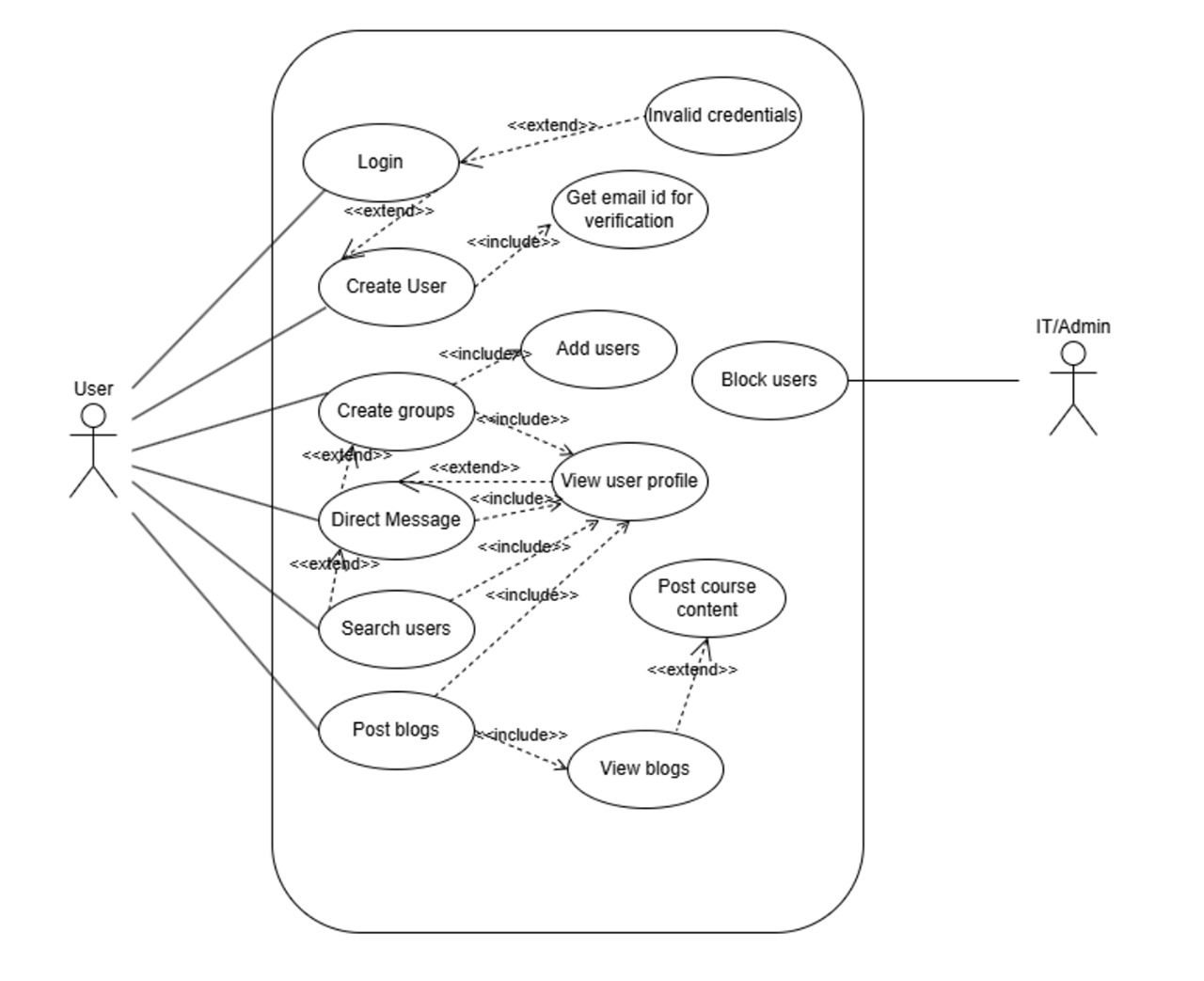
5 Notifications

FR-9: Users receive push notifications for new messages.

FR-10: Notifications are customizable (sound, vibration).

## Role Use Case

## C:\Users\DELL\AppData\Local\Packages\5319275A.WhatsAppDesktop_cv1g1gvanyjgm\TempState\8F822AC814829DA24A7065B8131BDF47\WhatsApp Image 2025-05-08 at 23.28.47_85daa118.jpg



# External Interface Requirements

## User Interfaces

UI-1: Login/Signup screen.

UI-2: Chat interface with text input, emojis, and media options.

UI-3: Profile customization screen.

## Hardware Interfaces

HI-1: Camera and gallery access for media sharing.

HI-2: Microphone for voice messages (future scope).

## Software Interfaces

SI-1: Supabase for real-time database.

SI-2: Supabase Authentication for user login.

## Communications Interfaces

CI-1: WebSockets for real-time messaging.

CI-2: REST APIs for future integrations.

# System Features

SF-1: Users can search for other users or groups by name.

SF-2: Search results are displayed in real-time.

## Search Article

|  |  |
| --- | --- |
| **Use Case Name** | *Search Chats, Groups and Contacts* |
| **XRef** | *Section 4.1, Search Chats, Groups and Contacts*  *SDD, Section 7.1* |
| **Trigger** | *The user accesses the search feature within the application* |
| **Precondition** | *The Results are displayed in a list containing the name and picture* |
| **Basic Path** | 1. *If the search is by Contact:* 2. *The system presents the list of all saved contacts* 3. *The user selects a contact* 4. *All the previous chats of the user are displayed in a new page* 5. *The user can send messages, media and search for other users* 6. *If the search is by Messages:* 7. *The user searches for a specific keyword or phrase* 8. *The system searches chat history and present a list of matches* 9. *The user selects a message which navigates to the specific chat* |
| **Alternative Paths** | *If the user searches by group or unknown number:*   1. *The user selects the group or unknown number* 2. *The user requests to join the group or start the chat* 3. *If the other user or the admin accepts the chat, the user can continue to chat and save the numbers* |
| **Post condition** | *The User successfully locates and interacts with contacts, groups, or messages based on the search criteria* |
| **Exception Paths** | *The Reader may abandon the chat at any time, if no search results are found then an empty list is shown with the text “No search results found”* |
| **Other** | *The user can back up the chats for use in another device* |

# Other Nonfunctional Requirements

## Performance Requirements

PR-1: The app should load in under 2 seconds.

PR-2: Messages should be delivered in real-time (latency < 1 second).

## Safety Requirements

SR-1: User data must be encrypted at rest and in transit.

## Security Requirements

SR-2: End-to-end encryption for messages.

SR-3: Regular security audits and penetration testing.

## Software Quality Attributes

QA-1: The app should have 99.9% uptime.

QA-2: The app should be user-friendly with a high usability score.

# Other Requirements

OR-1: The app must comply with university policies and data privacy regulations.

OR-2: The app must be scalable to support future features.

Appendix A: Glossary

|  |  |
| --- | --- |
| Term | Definition |
| MU-Connect | The social networking mobile application for Mahindra University students. |
| Flutter | A cross-platform UI toolkit for building natively compiled applications |
| Supabase | A backend-as-a-service (BaaS) platform for real-time database, authentication, and storage. |
| Real-Time Messaging | Instant exchange of messages between users with minimal latency. |
| End-to-End Encryption | A secure communication method where only the communicating users can read the messages. |
| API | Application Programming Interface, a set of protocols for building software. |
| UI/UX | User Interface/User Experience, focusing on app design and usability. |