Software Requirements Specification

for

MUVerse - Social Networking for Mahindra University Students

Version: 1.0

Prepared by

Group Name: ***SPACS***

D. Safdar Hussain	SE22UCSE085	Se22ucse085@mahindrauniversity.edu.in
P. Chaitra	SE22UCSE193	Se22ucse193@mahindrauniversity.edu.in
K. Pavithra	SE22UCSE136	Se22ucse136@mahindrauniversity.edu.in
K. Pranvith	SE22UCSE133	Se22ucse133@mahindrauniversity.edu.in
B. Shivani	SE22UCSE048	Se22ucse048@mahindrauniversity.edu.in
Alekhya Raavi	SE22UCSE022	Se22ucse022@mahindrauniversity.edu.in

Instructor:	*Vijay Rao Sir*
Course:	*Software Engineering*
Lab Section:	*IT- Block 2 (GF)*
Teaching Assistant:	*Surya Phani Teja Sir*
Date:	*March 10th, 2025*

Contents

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Safdar Hussain	Initial draft of SRS	10-03-2025

1 Introduction

MUVerse is a dedicated mobile social networking application tailored for Mahindra University students. The platform provides seamless communication, collaboration, and social engagement within the university community by enabling real-time messaging, media sharing, group discussions, and event coordination.

1.1 Document Purpose

This Software Requirements Specification (SRS) document outlines the requirements for MUVerse, a mobile social networking application designed exclusively for registered students of Mahindra University (MU) and defines the functional and non-functional requirements, design constraints, and quality benchmarks to ensure the system aligns with stakeholder expectations and industry best practices.

Version 1.0 represents the initial release of the requirements, covering the full scope of the system, including its features, interfaces, and constraints. The purpose of this document is to provide a clear and detailed specification for developers, testers, and stakeholders (including the Software Engineering course faculty and students) to ensure the successful development, testing, and deployment of MUVerse.

1.2 Product Scope

MUVerse is an engaging platform for university-exclusive networking aimed at providing secure communication among Mahindra University students. Similar to WhatsApp but tailored for an academic environment, it will enable students to connect, share updates, join interest-based groups, and communicate in real-time. The primary objectives are to facilitate real-time messaging, group interactions, and event sharing while ensuring privacy and ease of use within the MU ecosystem.

The application will offer:

- *Secure authentication using MU student credentials.*
- *Real-time one-on-one and group messaging with multimedia sharing.*
- *Profile personalization and privacy controls.*
- *Event creation and academic/extracurricular announcements.*
- *Notifications and activity updates.*
- *Robust moderation tools ensuring a safe and respectful environment.*

1.3 Intended Audience and Document Overview

This SRS is intended for the following audiences:

- ***Developers**: To understand the technical requirements and implement the system.*
- ***Project Managers**: To oversee timelines, milestones, and resource allocation.*
- ***Testers**: To design test cases based on functional and non-functional requirements.*
- ***Client (SE Course Faculty)**: To evaluate the project against academic expectations.*
- ***Professor**: To assess the document and project deliverables.*
- ***Students (End Users):** To understand app capabilities.*

The document is organized into five main sections, followed by appendices. Section 1 introduces the project, Section 2 provides an overall description, Section 3 details specific requirements, Section 4 covers non-functional requirements, and Section 5 addresses additional considerations. Readers should start with Section 1 for context, followed by Section 2 for an overview, and then proceed to Sections 3 and 4 based on their role (e.g., developers focus on Section 3, testers on Section 4).

1.4 Definitions, Acronyms and Abbreviations

Term/Acronym	***Definition***
API	*Application Programming Interface*
Firebase	*A platform for app development (Google)*
MERN	*MongoDB, Express.js, React, Node.js stack*
MU	*Mahindra University*
MUVerse	*Mahindra University Verse (project name)*
SRS	*Software Requirements Specification*
UI/UX	*User Interface/User Experience*
UML	*Unified Modeling Language*
WebSocket	*A protocol for real-time communication*
SPACS	*Student Project Alliance for Computer Science*

1.5 Document Conventions

This SRS adheres to IEEE formatting standards. Text is written in Arial font, size 11, single-spaced, with 1-inch margins on all sides. Section and subsection titles follow the template's bold formatting. Italics are used for comments or emphasis. Technical terms are defined in Section 1.4, and requirements are numbered for traceability (e.g., F1, P1). Diagrams and interfaces reflect SPACS's creative vision, ensuring clarity and professionalism.

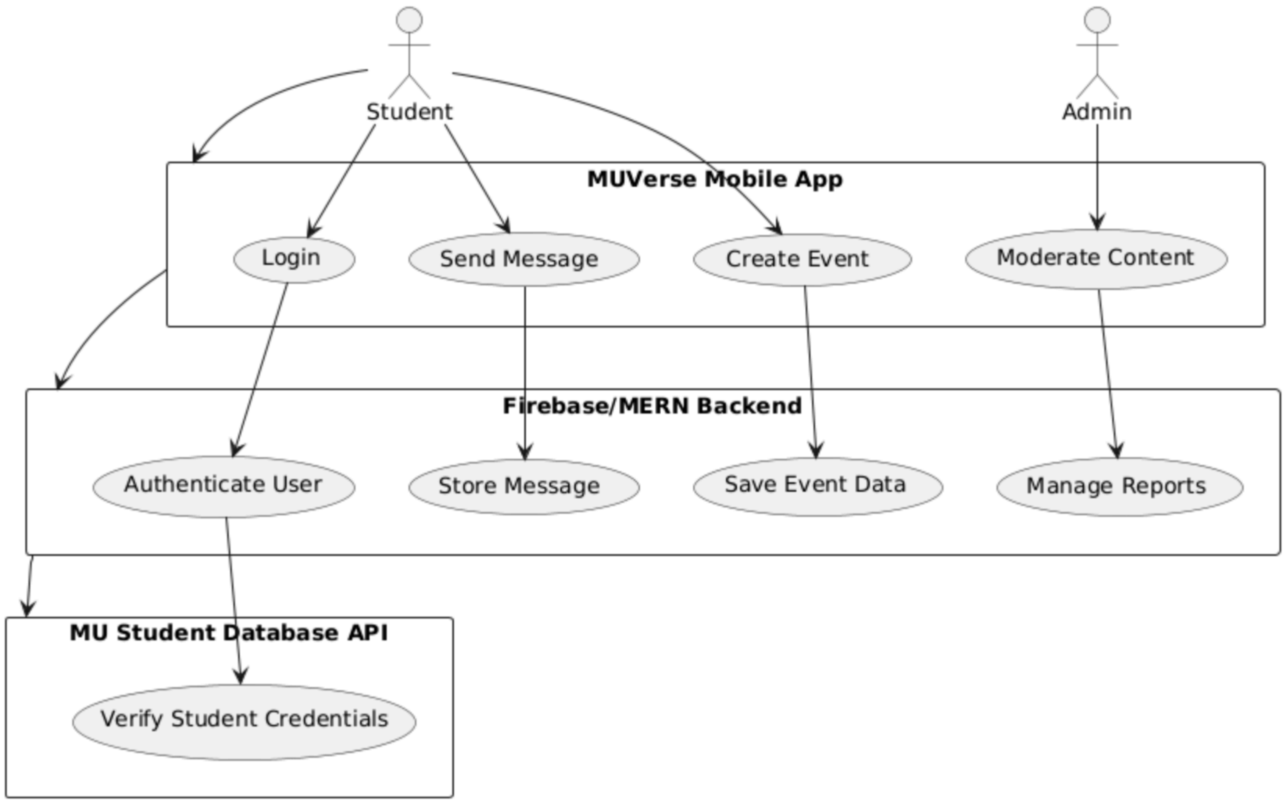
1.6 References and Acknowledgments

- *IEEE SRS Template Guidelines.*
- *"COMET: A UML-Based Software Modeling and Design Method" by H. Gomaa.*
- *Firebase Documentation: https://firebase.google.com/docs.*
- *React Native Documentation: https://reactnative.dev/docs.*
- *MDN: https://developer.mozilla.org/en-US/.*
- *Expo Documentation: https://docs.expo.dev/.*
- *Acknowledgments: SPACS thanks our Software Engineering faculty for guidance.*

2 Overall Description

2.1 Product Overview

MUVerse is a new, self-contained mobile application designed to serve as a social networking platform exclusively for Mahindra University students. Unlike general-purpose apps like WhatsApp, MUVerse is tailored to the academic context of MU, integrating with student credentials for authentication and focusing on features that enhance student collaboration and community engagement. The app operates within the MU ecosystem, interfacing with the university's student database for secure access and providing a scalable backend for real-time communication.



2.2 Product Functionality

- *User authentication via MU student credentials. [FR-01]*
- *Instant messaging between individual students. [FR-02]*
- *Group chats for interest-based or course-related discussions. [FR-03]*
- *Media sharing (images, videos, files, documents) for collaboration. [FR-04]*
- *Event announcements to inform students of university activities. [FR-5]*
- *University news feed for official updates. [FR-06]*

2.3 Design and Implementation Constraints

- *Must use the COMET method for software design (Reference: Gomaa, H., "Software Modeling and Design: UML, Patterns, and Software Architectures").*
- *Must employ UML modeling language for diagrams (e.g., use case diagrams).*
- *[CON-01] Dependent on MU providing access to the Student Database API for authentication.*
- *[CON-02] Strict academic deadlines (final deployment by May 18, 2025).*
- *[CON-03] Limited to React Native and Expo for frontend development and Firebase/MERN stack for backend.*
- *[CON-04] Real-time features rely on third-party services like Firebase, constraining scalability to service limits and must use open-source libraries to avoid licensing costs.*
- *[CON-05] The system must support Android and iOS platforms.*

2.4 Assumptions and Dependencies

- ***Assumptions**:*
  - *MU will provide access to student data for authentication by March 2025.*
  - *Students will adopt MUVerse as their primary communication tool for academic purposes.*
  - *Development tools (React Native, Firebase, etc.) will remain available and supported.*
  - *Firebase's free tier suffices for initial deployment (10k MAU).*
- ***Dependencies**:*
  - *Availability of MU Student Database API.*
  - *Stability of Firebase for real-time database and authentication services.*
  - *Students use Android/iOS devices with internet access.*
  - ***AWS S3**: For media storage.*
  - ***Twilio**: SMS-based 2FA.*

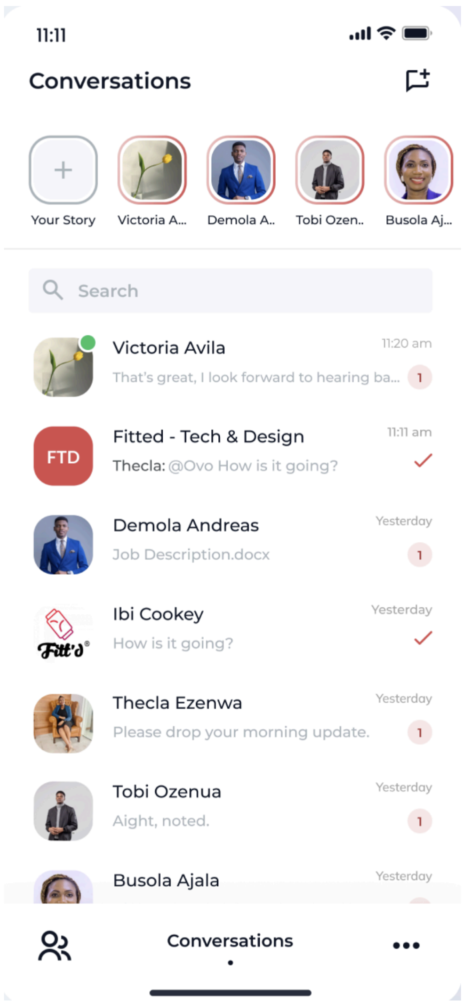
3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

MUVerse, crafted by SPACS, features a vibrant, intuitive mobile interface prototyped in Figma, reflecting MU's blue-and-gold palette. Key screens include:

- ***Home Screen**: Displays news feed and navigation to chats, groups, and events.*
- ***Chat Screen**: Supports instant messaging and media sharing with a clean, WhatsApp-like layout.*
- ***Group Screen**: Lists groups with options to join or create.*
- ***Event Screen**: Shows upcoming events with RSVP functionality.*
  Users will interact via touch-driven, with swipe gesture with menus and buttons for navigation.



Victoria Avila 11:20 am
That's great, I look forward to hearing ba... 1

Fitted - Tech & Design 11:11 am
Thecla: @Ovo How is it going?

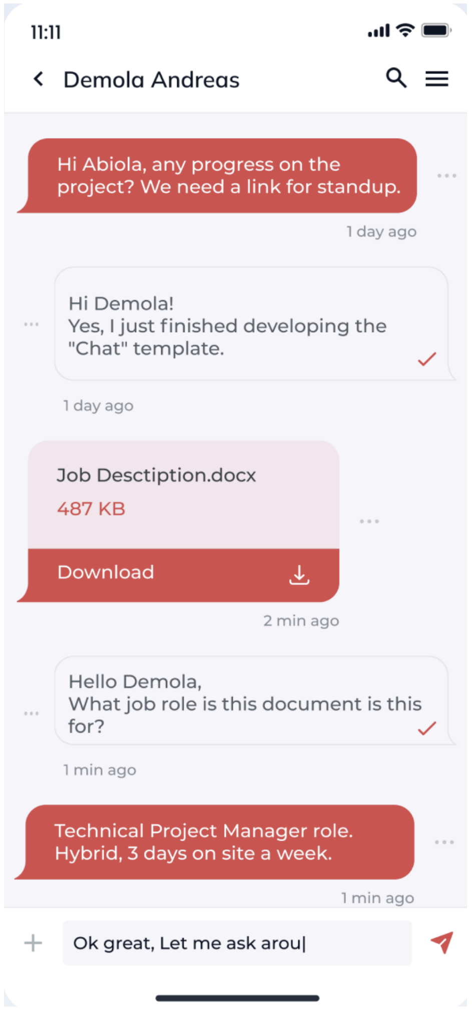
Demola Andreas Yesterday
Job Description.docx 1

Ibi Cookey Yesterday
How is it going?

Thecla Ezenwa Yesterday
Please drop your morning update. 1

Tobi Ozenua Yesterday
Aight, noted. 1

Busola Ajala Yesterday



Hi Abiola, any progress on the project? We need a link for standup.
1 day ago

Hi Demola!
Yes, I just finished developing the "Chat" template.
1 day ago

Job Desctiption.docx
487 KB
2 min ago

Hello Demola,
What job role is this document is this for?
1 min ago

Technical Project Manager role. Hybrid, 3 days on site a week.
1 min ago

Ok great, Let me ask arou|

3.1.2 Hardware Interfaces

MUVerse interfaces with the physical world through the hardware of modern smartphones, ensuring compatibility and leveraging device capabilities to deliver a stellar user experience by SPACS.

***Supported Hardware**:*

- ***Mobile Devices**: Compatible with Android (version 9.0 Pie and above) and iOS (version 13.0 and above) smartphones. Minimum specs include 2GB RAM and a dual-core processor for smooth performance.*
- ***Camera**: Interfaces with the device's front and rear cameras (minimum 8MP resolution) to capture high-quality images and videos for media sharing. The app requests camera access with a user-friendly prompt.*
- ***Storage**: Utilizes local storage (minimum 100MB free space) for caching messages, media, and offline news feed access. Data is read and written in English units (e.g., MB for file sizes).*
- ***Network Interface**: Relies on Wi-Fi or cellular data (4G/5G) for real-time communication, with a "read" interface to check connectivity status.*
- ***Touch Screen**: The primary input mechanism, supporting multi-touch gestures (e.g., pinch-to-zoom on media).*

3.1.3 Software Interfaces

MUVerse thrives on a carefully orchestrated network of software components, connecting the mobile app to external systems with precision and security. Designed by SPACS, these interfaces enable the app's core functionality while maintaining a seamless user experience.

***Key Software Components**:*

- ***MU Student Database API**:*
  - ***Purpose**: Authenticates users by validating MU student credentials (student ID and password).*
  - ***Interaction**: The app sends encrypted login requests and receives a success/failure response. On success, it retrieves a user profile (name, ID, course).*
  - ***Data Format**: JSON (e.g., {"student_id": "SE22UCSE085", "status": "authenticated"}).*
- ***Firebase**:*
  - ***Purpose**: Powers real-time database, authentication, and push notifications.*
  - ***Interaction**: The app writes messages to Firebase's Realtime Database and reads updates instantly via WebSocket connections. Push notifications alert users to new messages or events.*
  - ***Data Format**: Structured key-value pairs (e.g., chat_id: { "message": "Hi!", "timestamp": "2025-03-09T10:00:00Z" }).*
- ***WebSocket Protocol**:*
  - ***Purpose**: Facilitates low-latency, bidirectional communication for instant messaging.*
  - ***Interaction**: Establishes a persistent connection between the app and Firebase, pushing messages to recipients in real time.*
  - ***Data Format**: Lightweight text packets (e.g., {"to": "SE22UCSE193", "text": "Meeting now!"}).*

3.2 Functional Requirements

Functional requirements define the core behaviors of MUVerse, a revolutionary mobile social networking platform designed by the SPACS team for Mahindra University (MU) students. These requirements extend the high-level functionalities outlined in Section 2.2, translating them into precise, actionable services that the system must perform. Each function is engineered to foster connectivity, collaboration, and community engagement within MU's academic ecosystem, delivering a seamless and secure user experience.

3.2.1 F1: The system shall enable users to send instant message

- ***Description***: *Registered MU students can send real-time text messages to individual peers, ensuring swift and reliable communication. Messages support up to 500 characters and include delivery/read receipts.*
- ***Behavior***: *Upon sending, the message is transmitted via WebSocket to the recipient's device and stored in Firebase for persistence.*

3.2.2 F2: The system shall facilitate the creation and management of group chats.

- ***Description***: *Users can create group chats with a maximum of 100 members, assign a group name (up to 50 characters), and designate admins. Features include adding/removing members and muting notifications.*
- ***Behavior***: *Groups are instantiated in the backend database, with real-time updates synced across all members' devices.*

3.2.3 F3: The system shall allow users to share media within chats and groups.

- ***Description***: *Students can upload and share images, videos, and files (e.g., PDFs, docs) up to 50MB per item. Supported formats include JPEG, PNG, MP4, and PDF, with previews generated for visual media.*
- ***Behavior***: *Media is compressed client-side, uploaded to Firebase Storage, and linked in the chat stream for download/viewing.*

3.2.4 F4: The system shall enable the posting and management of event announcements.

- ***Description***: *Authorized users (e.g., students with event privileges or admins) can post events with details (title, date, time, location, description) and enable RSVP functionality. Events are visible to all users on the Event Screen.*
- ***Behavior***: *Events are stored in MongoDB, with push notifications sent to users upon posting or RSVP changes.*

3.2.5 F5: The system shall provide a university news feed.

- ***Description***: *A curated feed displays official MU updates (e.g., announcements, deadlines) posted by authorized personnel. Each entry includes a title, timestamp, and optional media (up to 50MB).*
- ***Behavior***: *Updates are fetched from the backend in chronological order, cached locally for offline access.*

3.2.6 F6: The system shall authenticate users using MU student credentials.

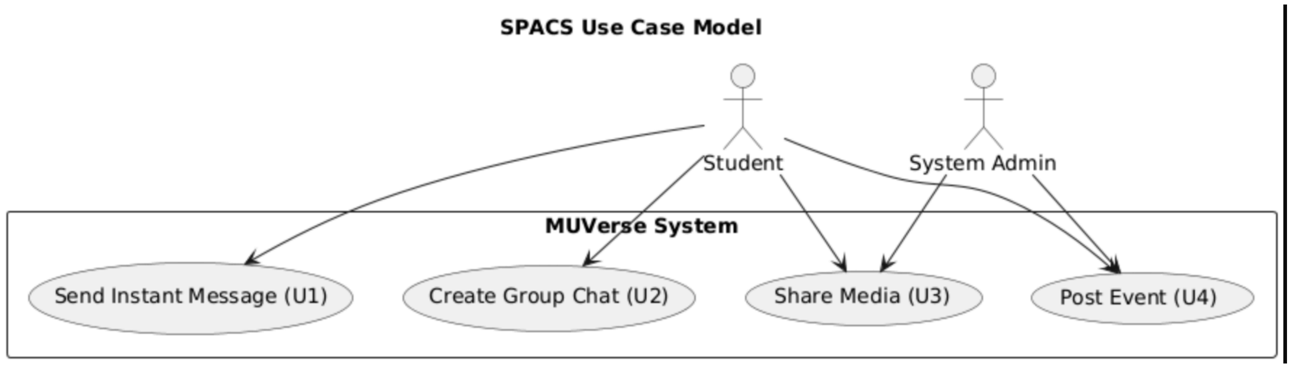
- ***Description**: Access requires a valid MU student ID (e.g., SE22UCSE085) and password, verified against the MU Student Database API. Successful login grants a session token valid for 24 hours.*
- ***Behavior**: Authentication requests are encrypted, with failed attempts tracked to enforce security policies.*

3.2.7 F7: The system shall support push notifications for user engagement.

- ***Description**: Users receive real-time alerts for new messages, group invites, event RSVPs, and news feed updates, customizable via settings.*
- ***Behavior**: Firebase Cloud Messaging (FCM) delivers notifications, with an opt-out option for non-critical alerts.*

3.3 Use Case Model

The Use Case Model encapsulates MUVerse's functionality, illustrating how actors interact with the system to achieve its objectives. Designed by SPACS, this model leverages UML to provide a clear, professional visualization of the app's core operations, ensuring traceability to functional requirements. Below, a use case diagram is described, followed by detailed specifications for key use cases.



3.3.1 Use Case #1: Send Instant Message (U1)

Author – D. Safdar Hussain, K. Pranvith, Alekhya
Purpose - Enable a student to send a real-time text message to another student, fostering instant communication.
Requirements Traceability – I F1, F7
Priority - High - Core to social networking functionality.
Preconditions - User is authenticated, has selected a contact, and has network connectivity.
Post conditions - Message is delivered to the recipient's device and stored in the chat history.
Actors – Student (sender), Student (recipient)

Extends – None
Flow of Events
1. Basic Flow -
   - Student opens the Chat Screen.

- Student types a message (e.g., “Hey, lab at 3?”) in the text input.
- Students tap the “Send” button.
- System transmits the message via WebSocket, displays it in the sender’s chat, and notifies the recipient with a push notification.

2. Alternative Flow -
   - If the recipient is offline, the system queues the message and delivers it upon their next login.
3. Exceptions:
   - Network failure: System displays “Message failed to send” and offers a retry option.

Includes: None
Notes/Issues - Ensure < 2s latency for a responsive experience; test edge cases like special characters.

3.3.2 Use Case #2: Create Group Chat (U2)

Author - P. Chaitra, B. Shivani, K. Pavithra
Purpose - Allow a student to create a group chat for collaborative discussions, enhancing community engagement.
Requirements Traceability - F2
Priority - Medium - Essential for group coordination.
Preconditions - User is authenticated and has network access.
Postconditions - A new group chat is created, visible to invited members, with the creator as admin.
Actors - Student (creator), Students (members)
Extends - None
Flow of Events -

1. Basic Flow:
   - Student navigates to the Group Screen and selects “Create Group.”
   - Student enters a group name (e.g., “SPACS Study Crew”).
   - Student adds members from their contact list (e.g., SE22UCSE193).
   - Student confirms creation; system initializes the group and notifies members.
2. Alternative Flow:
   - A selected member declines the invite; they are excluded from the group.
3. Exceptions:
   - Member limit (100) exceeded: System displays “Group full” error and prevents addition.

Includes: None
Notes/Issues: Test scalability with large groups; consider admin role delegation.

4 Other Non-functional Requirements

Non-functional requirements elevate MUVerse beyond mere functionality, defining its performance, security, and quality attributes to deliver an exceptional user experience for Mahindra University (MU) students. Crafted by the SPACS team, these requirements ensure the app is fast, secure, and robust, reflecting the innovative spirit of our academic community.

4.1 Performance Requirements

Performance requirements dictate the speed, responsiveness, and capacity of MUVerse, ensuring it meets the demands of a real-time social networking platform. These metrics are designed to support the functional requirements (Section 3.2) and are grounded in the need for seamless student interaction, even under heavy usage. The rationale is to maintain engagement and trust, critical for a university-exclusive app.

- ***P1**: The system shall deliver instant messages to recipients within 2 seconds under normal network conditions (4G/5G or Wi-Fi with >5 Mbps).*
  - *Rationale: Instant messaging (F1) is the app's heartbeat; delays beyond 2 seconds disrupt real-time conversation flow, risking user frustration. Developers must optimize WebSocket connections and Firebase latency.*
- ***P2**: The system shall load the news feed within 3 seconds upon app startup, assuming a stable network connection.*
  - *Rationale: A swift news feed (F5) ensures students access updates instantly, enhancing daily engagement. This requires efficient caching and MongoDB query optimization.*
- ***P3**: The system shall support 10,000 concurrent users without degradation in message delivery or app responsiveness.*
  - *Rationale: MU's student body could grow, and group chats/events (F2, F4) demand scalability. Firebase's load balancing must handle peak usage (e.g., during exams or events).*
- ***P4**: Media uploads (up to 50MB) shall complete within 10 seconds on a 10 Mbps connection.*
  - *Rationale: Quick media sharing (F3) keeps interactions fluid; longer waits deter usage. Client-side compression and Firebase Storage efficiency are key.*

4.2 Safety and Security Requirements

Safety and security requirements protect MUVerse users from data breaches, privacy violations, and misuse, ensuring a trustworthy platform. Given the app's handling of sensitive student data and its mobile nature, SPACS has prioritized robust safeguards, compliance with regulations, and proactive defenses, creatively tailored to MU's academic context.

- ***S1**: The system shall encrypt all user data (messages, media, profiles) using AES-256 standards in transit and at rest.*
  - *Safeguard: Prevents unauthorized access if data is intercepted or storage is compromised.*
  - *Rationale: Protects student privacy, aligning with India's IT Act, 2000, and MU policies.*

- ***S2**: The system shall lock user accounts after 5 consecutive failed login attempts, requiring a 10-minute cooldown before retry.*
    - *Action: Deters brute-force attacks on MU credentials (F6).*
    - *Rationale: Balances security with usability, preventing account hijacking while allowing genuine retries.*
- ***S3**: The system shall comply with MU privacy policies and India's Information Technology Act, 2000, for data handling and retention.*
    - *Requirement: Student data (e.g., IDs, messages) must not be shared externally without consent; retention limited to 1 year post-graduation.*
    - *Rationale: Legal and ethical necessity, ensuring trust in MUVerse as an MU-exclusive platform.*
- ***S4**: The system shall secure mobile connections using TLS 1.3 for all API and WebSocket interactions.*
    - *Safeguard: Encrypts communication between the app and Firebase/MU servers.*
    - *Rationale: Prevents eavesdropping on student chats, a must for a mobile app in public Wi-Fi settings.*
- ***S5**: The system shall log all admin actions (e.g., event postings, news updates) with timestamps and user IDs for auditability.*
    - *Action: Ensures accountability for authorized users (F4, F5).*
    - *Rationale: Mitigates misuse risks, providing a traceable record for MU oversight.*

4.3 Software Quality Attributes

*Software quality attributes define MUVerse's excellence beyond functionality, ensuring it meets both student expectations and developer needs. SPACS has selected three key attributes—**Usability**, **Scalability**, and **Reliability**—to reflect the app's role as a student-centric, future-proof platform. Each is specific, quantitative, and verifiable, with actionable strategies to achieve them, showcasing our Software Engineering coursework mastery.*

4.3.1 Usability

- ***U1**: The system shall enable new users to master core features (messaging, group creation, event RSVP) within 5 minutes, verified via beta testing feedback from 50 MU students.*
    - *How Achieved: Intuitive UI design via Figma prototypes, with tooltips on first use and a 3-tap maximum to access any feature (e.g., Home → Chat → Send). Consistency in teal-and-gold icons and layouts reduces the learning curve.*
    - *Rationale: Ease of use trumps complexity for student adoption; a short onboarding time ensures MUVerse becomes their go-to app.*

4.3.2 Scalability

- ***S1**: The system shall scale to support 20,000 concurrent users by 2026 without requiring major architectural overhaul, tested via simulated load on Firebase.*
    - *How Achieved: Leverages Firebase's auto-scaling Realtime Database and Cloud Functions, with sharding for group chats and events. Modular MERN stack design allows easy backend expansion (e.g., adding servers).*

- *Rationale: Prepares for MU's growth and potential inter-university expansion, ensuring MUVerse remains viable long-term. Design for change supports adding features like polls or live streams.*

4.3.3 Reliability

- ***R1**: The system shall maintain 99.9% uptime for messaging and news feed services, measured monthly via Firebase analytics, with downtime not exceeding 43 minutes.*
  - *How Achieved: Implements redundant Firebase backups and failover mechanisms. Offline caching ensures partial functionality during outages. Regular stress testing during beta phase (May 2025) identifies weak points.*
  - *Rationale: Reliability builds trust; students rely on MUVerse for critical updates and communication, especially during exams or events.*

5 Other Requirements

While optional, this section enriches MUVerse by addressing supplementary requirements that elevate its utility, compliance, and adaptability within Mahindra University's dynamic ecosystem. Designed by the SPACS team, these requirements reflect our commitment to delivering a holistic, forward-thinking solution that transcends basic expectations. They encompass database specifics, internationalization, legal considerations, accessibility, and analytics—each crafted to ensure MUVerse thrives as a student-centric platform.

5.1 Database Requirements

- ***D1**: The system shall utilize MongoDB as the primary database for storing user profiles, event details, and news feed entries, with a schema supporting rapid queries and scalability.*
  - *Details: User profiles include fields like student ID, name, course, and profile picture URL. Events store title, date, time, location, and RSVP list. News feed entries include title, content, timestamp, and optional media links.*

- *Rationale: MongoDB's NoSQL flexibility supports unstructured data (e.g., media metadata) and scales with MU's growing student base, complementing Firebase's real-time capabilities.*

- ***D2**: The system shall implement a data retention policy, archiving inactive user data (e.g., graduated students) after 12 months and deleting it after 24 months.*
  - *Rationale: Balances storage efficiency with compliance (see 5.3), ensuring MUVerse remains lean and legally sound.*

5.2 Internationalization Requirements

- ***I1**: The system shall support dual-language interfaces in English and Telugu, with seamless switching via user settings.*
  - *Details: All UI text (e.g., "Send," "Create Group") and error messages (e.g., "Network Error") are stored in language files, dynamically loaded based on user preference.*
  - *Rationale: Reflects Hyderabad's linguistic diversity, making MUVerse inclusive for Telugu-speaking students while maintaining English as the academic default.*
- ***I2**: The system shall format dates and times according to user locale (e.g., DD-MM-YYYY for India, 12-hour clock).*
  - *Rationale: Enhances usability (U1) by aligning with regional conventions, critical for event postings (F4).*

5.3 Legal Requirements

- ***L1**: The system shall comply with India's Information Technology Act, 2000, and the Digital Personal Data Protection Act, 2023 (if enacted), regarding user consent, data storage, and breach notification.*
  - *Details: Users must consent to data collection (e.g., profile, messages) via a pop-up on first login. Data breaches must be reported to MU authorities within 72 hours.*
  - *Rationale: Ensures legal integrity and student trust, aligning with S3 (Section 4.2).*
- ***L2**: The system shall include a terms-of-use agreement prohibiting misuse (e.g., spam, harassment), enforceable by account suspension.*
  - *Rationale: Protects the MU community, reinforcing MUVerse as a safe space.*

5.4 Accessibility Requirements

- ***A1**: The system shall adhere to WCAG 2.1 Level AA standards, ensuring accessibility for students with disabilities.*
  - *Details: Features include screen reader compatibility (e.g., VoiceOver, TalkBack), high-contrast mode (teal-on-white text), and scalable font sizes (up to 200%).*
  - *Rationale: Promotes inclusivity, ensuring all MU students can engage with MUVerse, enhancing usability (U1).*
- ***A2**: The system shall support voice input for messaging and event creation, activated via a microphone icon.*
  - *Rationale: Aids students with motor impairments and adds convenience, aligning with modern app trends.*

Appendix A – Data Dictionary

The Data Dictionary serves as the backbone of MUVerse, cataloging every critical element—variables, states, constants, inputs, and outputs—that defines the system's behavior and structure. Crafted by the SPACS team, this table ensures traceability across the SRS, linking each item to its corresponding requirements and operations. It reflects the app's role as a dynamic, student-centric social networking platform for Mahindra University (MU), providing developers and evaluators with a clear, concise reference to build and assess MUVerse with precision.

Item Name	Type	Description	Possible States/Values	Related Operations	Related Requirements
Message	Variable	A text string sent between users in real-time chats. Includes metadata like sender and timestamp.	String (max 500 chars); Sent, Delivered, Read	Send, Receive, Display, Store	F1, U1, P1, S1, R1

Appendix B - Group Log

Date	Activity/Meeting Details	Participants	Effort Contribution
07-02-2025	**Project Kickoff**: Meeting to assign roles and brainstorm *MUVerse* vision. Defined scope and objectives via SOW.	All Members	Led discussions, drafted SOW (8 hrs). Team suggested initial ideas (e.g., chat focus) (~1 hr each, 5 hrs total).
10-02-2025	**Requirements Gathering**: Research on MU student needs and technical feasibility; team feedback session planned.	Safdar Hussain, Chaitra, Pranvith	Safdar compiled user stories and tech stack (3 hrs). Chaitra and Pranvith offered verbal feedback (~2 hrs each, 4 hrs total).
15-02-2025	**UI/UX Design Session**: Designed Figma prototypes; team meeting scheduled but sparsely attended.	Chaitra, Safdar	Safdar created teal-and-gold UI mockups (2 hrs). Chaitra suggested color scheme (~1 hrs).
20-02-2025	**Functional Requirements Drafting**: Defined F1-F7; group call planned, limited participation.	Pavithra, Pranvith, Safdar	Safdar wrote Section 3.2 (10 hrs). Pavithra and Pranvith reviewed draft, suggested RSVP feature (~2 hrs).
25-02-2025	**Use Case Development**: Modeled use cases (U1-U4) and UML diagram; team input sought but minimal.	Shivani, Pavithra, Alekhya	Pavithra developed use cases and diagram (4 hr). Alekhya and Shivani proposed event notification idea (~2 hrs). Others missed call.
01-03-2025	**Non-functional Requirements**: Drafted performance, security, and quality attributes; team meeting canceled.	Safdar Hussain, Pranvith	Safdar crafted Section 4 (4 hrs). Pranvith emailed latency concern (~4 hr).
05-03-2025	**Additional Requirements**: Added database, internationalization, and legal requirements (Section 5).	Safdar Hussain, Chaitra	Safdar wrote Section 5 (8 hrs). Chaitra suggested other languages support (~4 hr).
08-03-2025	**SRS Compilation & Polish**: Reviewed and formatted entire SRS, integrating all sections.	Chaitra, Alekhya, Shivani, Pavithra	Chaitra and Alekhya finalized document (4 hrs). Shivani and Pavithra proofread minor sections (~2 hrs each, 4 hrs total).
09-03-2025	**Submission Preparation**: Verified SRS completeness and prepared for submission.	All Members	Safdar ensured quality and submitted (5 hrs). Team notified of final draft; no further input.

5.1.1 Total Effort Summary:

100 hours

- **D. Safdar Hussain**: ~20 hours – Led planning, design, drafting, and finalization
- **P. Chaitra**: ~20 hours – Provided early feedback and language suggestion.
- **K. Pavithra**: ~16 hours – Contributed RSVP idea and proofreading.
- **K. Pranvith**: ~16 hours – Offered tech feedback and latency note.
- **B. Shivani**: ~16 hours – Suggested UI colors and proofread.
- **Alekhya Raavi**: ~16 hours – Proposed notification feature.