Software Requirements Specification (SRS)

Project: EduMentor - Mentor-Student Connection and Issue Escalation System

1. Introduction

1.1 Purpose

To develop a web-based platform connecting students and mentors, enabling students to submit queries/issues for resolution. When a mentor cannot resolve an issue, it escalates to the Head of Department (HOD) for further action.

1.2 Scope

The EduMentor system includes:

- Student registration and query submission
- Mentor assignment and issue resolution tracking
- Real-time chat
- HOD dashboard for escalations
- Admin controls

1.3 Overview

The system allows students to interact with mentors and raise issues. Mentors manage students and resolve queries. Unresolved issues escalate to the HOD, who can intervene or re-assign mentors.

2. General Description

2.1 Functions

- User registration and authentication for students, mentors, and HODs
- Submission and management of issues
- Mentor assignment (automatic and manual)
- Real-time chat between students, mentors, and HODs
- Issue escalation to HOD if unresolved by mentors

- Admin management for all user roles

2.2 User Community

- Students: Submit issues, chat with mentors and HOD, track progress
- Mentors: View mentees, resolve queries, track open/closed cases
- HOD: View all mentors, intervene in unresolved cases, resolve/escalate issues
- Admin: Oversee users and issues, platform analytics

---

3. Functional Requirements

3.1 Possible Outcomes

- Issue resolved by mentor
- Issue escalated to HOD (if unsolved within time/criteria)
- Issue resolved/closed by HOD
- Feedback collected from student

3.2 Ranked Order

Issues escalate based on:

- Time outstanding
- Priority set by student/mentor
- Mentor activity/inactivity

3.3 Input-Output Relationship

- Input: User registrations, issue submissions, chat messages
- Processing: Assignment to mentors, escalation logic, activity logging
- Output: Status updates, notifications, dashboard analytics

---

4. User Interface Requirements

4.1 Software Interfaces

- Student dashboard
- Mentor dashboard
- HOD dashboard (shows all mentors, flagged/unresolved issues, options for intervention)
- Admin dashboard

4.2 Interface Examples

- Query submission form
- Mentor's list of assigned issues (with status)
- HOD window displaying all mentors and any escalated issues
- Status notifications to users

5. Performance Requirements

5.1 Response Time

Core actions (login, message, status update) under 2 seconds.

5.2 Throughput

Support at least 500 concurrent active users.

5.3 Scalability

System can scale to universities/large institutions with 10,000+ users.

6. Non-Functional Attributes

6.1 Usability

Simple, modern, mobile-friendly design for all roles.

6.2 Reliability

99%+ uptime with robust database backup.

6.3 Security

- Role-based access (student, mentor, HOD, admin)
- Encrypted passwords and sensitive data
- Secure session and communication protocols

7. Schedule and Budget

7.1 Timeline

- Total duration: 6 months
  - Month 1: Requirements finalized, designs created

- Month 2: Student/mentor core flows implemented
- Month 3: Chat and dashboard integration
- Month 4: HOD dashboard and escalation features
- Month 5: Testing and optimization
- Month 6: Deployment and monitoring

7.2 Cost Estimate

Based on:

- Developer/QA costs
- Cloud hosting and maintenance
- Third-party chat/notification tools (if necessary)

---

8. Appendices

8.1 Supplementary Information

- UML Use Case: Issue escalation
- ERD for user, issue, chat, escalation entities
- Sequence diagram: Issue escalation workflow

8.2 Glossary

- Issue: Student-submitted query/problem
- Mentor: Faculty or experienced student assigned to guide/resolve issues
- HOD: Head of Department; authority to resolve escalated cases
- Admin: System superuser with full access