**SOFTWARE REQUIREMENT SPECIFICATION**

**Project Title: Tutor Application System**

## ****PROBLEM STATEMENT:****

The absence of a centralized, reliable system for connecting students with qualified tutors poses significant challenges in the academic support ecosystem. Students face difficulties in identifying suitable tutors, while educators struggle to reach learners effectively through traditional or unstructured channels.

The proposed **Tutor Application System** aims to resolve these inefficiencies by delivering a secure, scalable, and intuitive digital platform. It facilitates seamless tutor discovery, session management, and administrative oversight, thereby enhancing accessibility, transparency, and effectiveness in personalized learning experiences.

1.INTRODUCTION:

**1.1 Purpose**

This Software Requirements Specification (SRS) defines the scope, functionality, and performance of the Tutor Application System, a web-based platform that connects students with qualified tutors. The purpose is to streamline the process of tutor discovery, application management, and administration oversight. This document will serve as a foundation for system design, development, and testing.

**1.2 Intended Audience**

* **Clients & Stakeholders** – to validate that the solution meets business goals.
* **Development Team** – to understand system design and implement requirements.
* **Testers** – to derive test cases based on the functional requirements.
* **UI/UX Designers** – to understand the expected user experience.

### 1.3 Scope

The Tutor Application System enables three types of users:

* **Students** can register, search tutors based on subjects, and apply.
* **Tutors** can register, create a teaching profile, and manage student applications.
* **Admins** can monitor the platform, approve tutors, and manage subjects and user data.

The final deliverable includes:

* A **responsive web application**
* RESTful **APIs** using **Java Spring Boot**
* A **MySQL database**
* Basic **security** and **role-based access control**

**1.4 Definitions and Acronyms**

**SRS –** Software Requirements Specification

**CRUD –** Create, Read, Update, Delete

**UI –** User Interface

**API –** Application Programming Interface

**2. OVERALL DESCRIPTION**

### 2.1 Product Perspective

This is a **standalone full-stack web application** developed using modular architecture. The system is extensible to integrate features like in-app chat, rating systems, and mobile access.

### 2.2 Product Functions

The key functions of the system include:

* **Secure user authentication and authorization.**
* **Tutor profile creation and subject assignment.**
* **Tutor search with advanced filters (subject, location).**
* **Application tracking and management for both students and tutors.**
* **Administrative control for user and subject management**.

### 2.3 User Classes and Characteristics

|  |  |  |
| --- | --- | --- |
| User Role | Description | Technical Expertise |
| **Student** | Learner seeking tutoring services. Can apply to tutors. | Basic |
| **Tutor** | A registered user who offers teaching services. | Moderate |
| **Admin** | Platform controller who manages tutors, students, and subjects. | Advanced |

### 2.4 Assumptions and Dependencies

* Users will have stable internet access and modern browsers.
* The backend server is hosted and accessible over HTTPS.
* The system will not handle payments in version 1.0.
* Future integrations may include email/SMS notifications or calendar sync.

**3. SPECIFIC REQUIREMENTS**

### 3.1 Functional Requirements

#### Student Module

* **FR1**: Student registration and login.
* **FR2**: Search tutors by subject, name, or location.
* **FR3**: View tutor profile and availability.
* **FR4**: Apply to a tutor and track application status.

#### Tutor Module

* **FR5**: Tutor registration and login.
* **FR6**: Create and manage tutor profile with subjects and availability.
* **FR7**: View, accept, or reject student applications.

#### Admin Module

* **FR8**: View all users (students and tutors).
* **FR9**: Approve or reject tutor applications.
* **FR10**: Manage subject categories (CRUD operations).

### 3.2 Non-Functional Requirements

* The system shall ensure password encryption using a secure hashing algorithm to protect user credentials.
* The system shall provide a **responsive user interface** accessible on both desktop and mobile devices.
* The system shall ensure **API response times are ≤ 2 seconds** under normal load conditions.
* The system shall implement **role-based access control (RBAC)** at both the frontend and backend for secure feature access.
* The application shall follow a **modular and scalable architecture**, allowing for future feature additions and system enhancements.
* The system shall maintain **99% uptime** in a production environment to ensure high availability and reliability.

**4. EXTERNAL INTERFACE REQUIREMENTS**

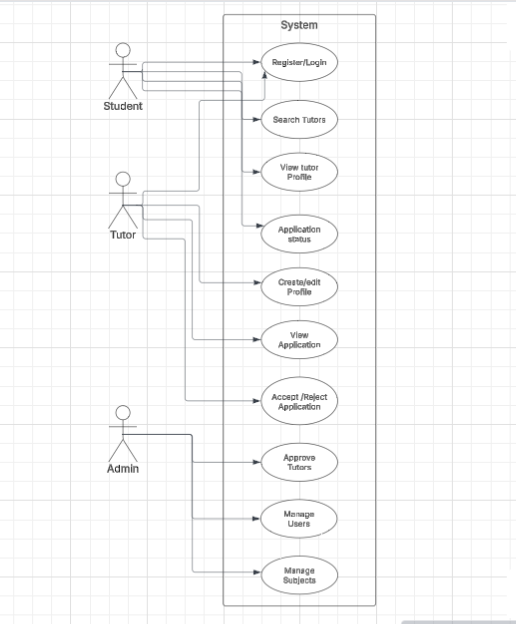
**4.1 User Interfaces**

* Responsive UI for all user roles.
* Login/Register Pages for all users.
* Dashboard tailored to each user type.
* Tutor Search Page with filters and result listing.
* Application Management interfaces**.**

**5. System Models**

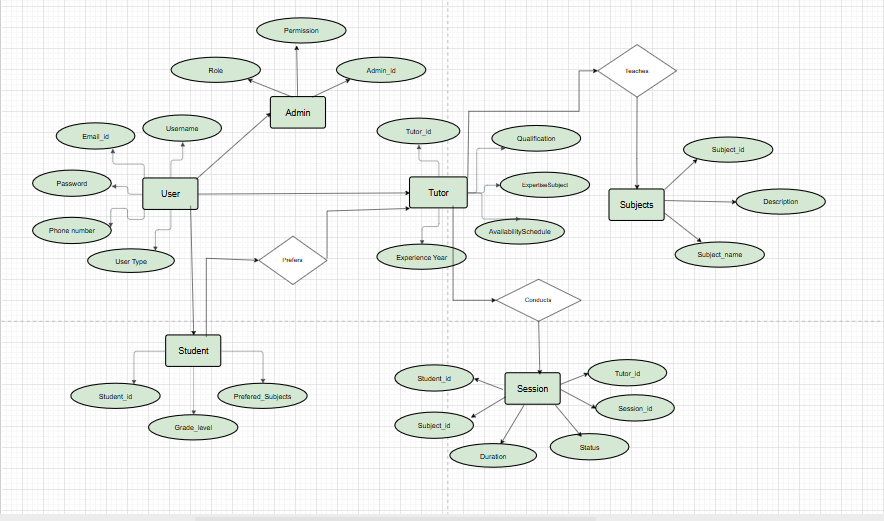
**5.1 Use Case** **Diagram**

* Actors: Student, Tutor, Admin
* Use cases: Register/Login, Search Tutors, Apply, Create Profile, Approve Tutors



**5.2 Entity Diagram (High-level)**

* User (id, name, email, password, role)
* TutorProfile (id, userId, subjects, availability, location)
* Subject (id, name)
* Application (id, studentId, tutorId, status)



# Conclusion

The Tutor Application System is a comprehensive solution aimed at modernizing and streamlining the tutoring process. By providing a centralized platform for students, tutors, and administrators, it enhances accessibility, efficiency, and user experience. This system not only simplifies tutor-student interactions but also ensures transparency, scalability, and reliability in delivering personalized education support.