CSE-309

SRS Document

Done by:

Name : Tasdid Ahmed Ahan

ID : 2222019

COURSE: CSE-309

SECTION: 05

SEMESTER: Summer-25

Software Requirements Specification (SRS) for TutorBridge

1. Introduction

1.1 Purpose

The purpose of TutorBridge is to provide a reliable tuition matchmaking platform that connects students with qualified tutors. The system facilitates tutor applications, student tuition requests, messaging, reporting, payments, and admin moderation. This SRS defines the functional and non-functional requirements of the system, serving as a reference for developers, testers, and stakeholders.

1.2 Document Conventions

- **Roles**: Student, Tutor, Admin
- **Status terms**: *Pending, Approved, Active, Blocked, Open, Closed*
- **UI terms**: Tabs, Buttons, Modals
- **Database references**: Written in lowercase (e.g., `profiles`, `requests`, `reports`).

1.3 Intended Audience and Reading Suggestions

- **Developers**: Focus on Sections 2–4 for technical design and implementation.
- **Testers**: Review Section 3 (System Features) and Section 5 (Nonfunctional Requirements).
- **Project Managers**: Review Sections 1 and 2 for scope and assumptions.
- **End Users**: Not a target audience, but may refer to Section 2.3 (User Classes).

1.4 Project Scope

TutorBridge is a web-based system that enables:

- Students to post tuition requests and hire tutors.
- Tutors to browse tuition requests and apply.
- Admins to oversee users, requests, payments, and reports.
  The system ensures transparency, security, and efficiency in private tutoring arrangements.

1.5 References

- Supabase documentation (https://supabase.com/docs)
- Project schema SQL design
- HTML/CSS/JS frontend implementation

2. Overall Description

2.1 Product Perspective

TutorBridge is a standalone web application with backend integration via Supabase (PostgreSQL + authentication + storage). It operates as a cloud-hosted service and supports role-based access.

2.2 Product Features

- Student Dashboard (manage requests, applications, messages, payments)
- Tutor Dashboard (browse requests, apply, manage applications, messages)
- Admin Dashboard (user management, request moderation, report resolution, payments & analytics)
- Authentication with role-based redirection (student, tutor, admin)
- Messaging between students and tutors

- Payment tracking and commission collection

2.3 User Classes and Characteristics

- **Students**: Post tuition requests, hire tutors, make payments.
- **Tutors**: Apply for tuition requests, communicate with students, receive payments.
- **Admins**: Approve tutors, manage users, resolve reports, oversee payments.

2.4 Operating Environment

- **Frontend**: HTML, CSS, JavaScript (vanilla, no framework).
- **Backend**: Supabase (PostgreSQL, Auth, Storage).
- **Browsers**: Chrome, Firefox, Edge.
- **Devices**: Desktop/laptop primarily (mobile optimization optional).

2.5 Design and Implementation Constraints

- Hosted entirely on web (no native app).
- Supabase service availability is critical.
- Role-based access must be enforced for security.

2.6 User Documentation

- Quick Start Guide (to be provided).
- FAQ and Help page on the website.

2.7 Assumptions and Dependencies

- Users have stable internet access.
- Payments are handled manually or through integrated gateways (future expansion).
- Supabase availability is assumed.

3. System Features

3.1 Authentication & User Roles

- **Description**: Login/signup with role selection (Student, Tutor, Admin).
- **Priority**: High.
- **Stimulus/Response**: User logs in → system validates → redirects to dashboard.
- **Functional Requirements**:
    - Users must select role during signup.
    - Admin role assigned manually in DB.
    - Login redirects based on role.

3.2 Student Features

- Post tuition requests (subject, class, location, salary, type).
- Manage requests (edit, close, delete).
- Review tutor applications.
- Accept/reject tutors.
- Messaging with tutors.
- Track payments.

3.3 Tutor Features

- Browse open requests.
- Apply to requests (with proposed salary).
- Manage applications (withdraw, check status).
- Messaging with students.
- View payments and commission.

3.4 Admin Features

- Manage users (approve tutors, block/unblock).
- Manage requests (edit, assign tutors, close/delete).
- Moderate reports (resolve, block users).
- Track payments and commissions (mark collected).
- Analytics dashboard (users, requests, payments, reports).

3.5 Messaging

- Real-time messaging between tutors and students.
- Messages stored in `messages` table.
- Notifications on new messages.

3.6 Reports

- Students/tutors can report inappropriate behavior.
- Admins review reports, resolve them, and take action.

3.7 Payments & Commission

- Students make payments to tutors.
- Admins collect 10% commission.
- Commission status tracked (pending/collected).

4. External Interface Requirements

4.1 User Interfaces

- Responsive web pages for Student, Tutor, and Admin dashboards.
- Standardized button styles (primary, secondary, danger, success).
- Tab-based navigation for admin panel.

4.2 Hardware Interfaces

- No special hardware required beyond standard devices (PC/laptop).

4.3 Software Interfaces

- Supabase APIs for authentication, database, and storage.

4.4 Communications Interfaces

- HTTPS protocol for all communication.
- WebSockets (optional) for real-time messaging.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- Page load within 3 seconds on stable internet.

- Handle 100+ concurrent users initially.

5.2 Safety Requirements

- Prevent system crashes by validating all inputs.
- Graceful fallback to mock data if backend unavailable.

5.3 Security Requirements

- Role-based access control (RBAC).
- Passwords stored securely (via Supabase).
- Reports and payments accessible only by admins.

5.4 Software Quality Attributes

- **Maintainability**: Modular JS files per feature.
- **Portability**: Runs on all modern browsers.
- **Usability**: Intuitive dashboards with consistent UI.

6. Other Requirements

- Future expansion to mobile app.
- Payment gateway integration (Stripe/SSLCommerz).
- Tutor rating/review system enhancement.

Appendices

A. Glossary

- **Request**: A tuition post created by a student.

- **Application**: A tutor’s response to a request.
- **Commission**: 10% fee collected by the platform.

B. Analysis Models

- Use case diagrams (not included here but can be added).

C. Issues List

- Mobile optimization (pending).
- Real-time notifications (future).