American College of Technology (ACT)

Department of Computer Science

Undergraduate Capstone Project

Title: Tutoring Platform

Group Members:

No	**Name**	**ID**
1	Albastros Kebede	008/BSC-B4/2022
2	Hermella Wondwossen	036/BSC-B4/2022
3	Melat Belay	049/BSC-B4/2022

Advisor Name: Instructor Mr.Abiy Signature_____________

24/07/2025

Acknowledgment

We would like to express our heartfelt gratitude to everyone who supported us throughout our capstone project journey. First and foremost, we thank God for His guidance, strength, and blessings that made this project possible. We are deeply grateful to our project advisor, Mr. Abiy, for his exceptional guidance, insightful feedback, and unwavering support, which were invaluable to our project's success. We also sincerely appreciate our instructors Ms. Sofanit and Ms. Marta and friends who generously shared their knowledge and answered our questions to help refine our work. Special thanks go to our families for their constant support, patience, and encouragement, which motivated us throughout this endeavor. Finally, we acknowledge all those who contributed directly or indirectly to classmates, and professionals whose advice and feedback helped shape this project. This achievement would not have been possible without each of you, and we are truly thankful for your invaluable contributions.

Abstract

This project addresses the significant challenges within Ethiopia's informal and fragmented tutoring landscape, where students struggle to find qualified tutors and existing systems often lack essential digital support, evaluation mechanisms, and broad course coverage. To overcome these issues, a comprehensive web-based Tutoring Platform has been developed to centralize and enhance educational support nationwide. Designed to serve learners from Grades 1-12, university, and college levels, as well as individuals seeking language courses like English, the platform provides flexible options for both online and in-person tutor scheduling. Key features include a robust filtering system for tutor selection, and integrated tools for scheduling, chat, video conferencing (Jitsi Meet), and assessment quizzes. Furthermore, it incorporates a secure payment system (Chapa) and a built-in rating and review system to enhance user trust and accountability. Built using a modern technology stack comprising Next.js for the backend, React for the frontend, and MongoDB as the database. This platform aims to bridge the existing gaps, improve learning outcomes, and foster a more efficient, transparent, and accessible tutoring ecosystem across Ethiopia.

ይዘት

ይህ ፕሮጀክት በኢትዮጵያ መደበኛ ባልሆነ እና በተበታተነ የማጠናከሪያ ትምህርት ገጽታ ውስጥ ያሉ ጉልህ ተግዳሮቶችን የሚፈታ ሲሆን ተማሪዎች ብቁ አስተማሪዎች ለማግኘት የሚታገሉበት እና ነባር ስርዓቶች ብዙ ጊዜ አስፈላጊ የዲጂታል ድጋፍ፣ የግምገማ ዘዴዎች እና ሰፊ የኮርስ ሽፋን የላቸውም። እነዚህን ጉዳዮች ለመቅረፍ፣ በአገር አቀፍ ደረጃ የትምህርት ድጋፍን ማእከላዊ ለማድረግ እና ለማጎልበት አጠቃላይ በድረ-ገጽ ላይ የተመሰረተ የማጠናከሪያ ትምህርት መድረክ አዘጋጅተናል። ከ1-12ኛ ክፍል ተማሪዎችን ፣የዩኒቨርሲቲ እና የኮሌጅ ደረጃዎች እንዲሁም እንደ እንግሊዘኛ ያሉ የቋንቋ ኮርሶች ለሚፈልጉ ግለሰቦች ድህረ ገጹ ኢንተርኔት ላይ በበይነመረቡ በኩል በኦንላይን ስብሰባ ወይም በአካል ተገኝቶ የማጥናት ክፍለ ጊዜ አማራጮችን ለተማሪዎች እና አስተማሪዎች ይሰጣል። የድረ-ገጹ ቁልፍ ገፅታዎች ለተጠቃሚ ምቹ የሆኑ ሁለት ቋንቋዎችን (እንግሊዝኛ እና አማርኛ) የሚያካትቱ፣ መምህራንን ለመምረጥ የሚያስችል ጠንካራ የማጣሪያ ዘዴ እና ለክፍለ-ጊዜ መርሃ ግብር፣ ለቻት፣ ለቪዲዮ ኮንፈረንስ (ጂትሲ ስብሰባ) እና ለተማሪዎች ራሳቸውን እንዲገመግሙ የፈተና ጥያቄን ያካትታል። በተጨማሪም ደህንነቱ የተጠበቀ የክፍያ ስርዓት (ቻፖ) እና በድረ-ገጹ ውስጥ የተካተተ የደረጃ አሰጣጥ እና የግምገማ ስርዓት ተማሪዎቹ የሚያስተምራቸውን መምህር ደረጃ መስጠት እና ሌሎች ተማሪዎች ደረጃውን ማየት ይችላሉ። በአጠቃላይ ዘመናዊ ቴክኖሎጂ በመጠቀም የተገነባነው ይህ ድረ-ገጽ አሁን ያሉትን ክፍተቶች ለማስተካከል፣ የትምህርት ውጤቶችን ለማሻሻል እና ቀልጣፋ፣ግልጽ እና ተደራሽ የማስተማሪያ ድህረገፅ በመላ ኢትዮጵያ ለመፍጠር ያለመ ነው።

Contents

List of figures

List of tables

List of Abbreviations

EUEE	Ethiopian University Entrance Examination
UML	Unified Modeling Language
MongoDB	Mongo Data Base
Next.js	Next java script
APIs	Application programming interface
NoSQL database	not only Structured Query Language
JWT-based authentication	JSON Web Token
HTTPS	Hypertext Transfer Protocol Secure
N/A	not applicable
e.g	example
ETN	(external verifier) user that verify submitted tutor document

Chapter One: Introduction

1.1 Background

Ethiopia's education system follows a structured 8-4-4 framework (8 years of primary, 4 years of secondary, and 4 years of university education) with national assessments at key academic milestones. The system emphasizes teacher-centered instruction in most classrooms, with large student-to-teacher ratios that often limit individualized attention. The curriculum is standardized nationwide, with instruction primarily delivered in local languages at primary levels and transitioning to English in secondary and higher education creating language barriers for many students. At critical junctures, students face standardized exams including the Grade 6 regional exam, Grade 8 regional exam, and the Grade 12 Ethiopian University Entrance Examination (EUEE) which determines higher education access. University students must pass a national exit exam before graduating, while graduate students take comprehensive master's exams. While these national evaluations reflect a strong push by the Ministry of Education toward standardizing and enhancing the quality of education across the country. However, this structure also places significant academic pressure on students at all levels.

Despite these advancements, many learners across Ethiopia face ongoing challenges in accessing the academic support needed to succeed in these exams. Traditional tutoring systems are often limited by geographical constraints, as both students and tutors struggle to find each other, whether in rural areas where qualified tutors are scarce or in urban centers where connections remain fragmented. The process of matching learners with appropriate tutors is difficult nationwide, with no centralized system to facilitate these connections. Moreover, the absence of structured feedback, flexible learning options, and communication further diminishes the effectiveness of traditional tutoring arrangements that do manage to form. This disconnects affects all parties, students can't find the right tutors, while qualified tutors can't reliably reach students who need their help.

To address these gaps, the tutoring platform has been developed to connect students with verified tutors nationwide. Designed to serve learners from Grades 1–12, university, and college levels, as well as individuals seeking language courses like English. The platform also offers both online and in-person tutoring within an integrated system. Key features include flexible scheduling, secure local payment options, assessment quizzes, chat, and a robust rating and feedback system. Students can monitor their progress, receive personalized academic support, and choose between private and group sessions tailored to their learning preferences. Meanwhile, tutors can expand their reach, manage sessions efficiently, and grow through transparent performance feedback. By digitizing and centralizing tutoring services, this platform enhances the tutoring experience in Ethiopia, supporting both academic success and long-term educational growth.

1.2 Statement of the problem

1.2.1 Existing System

In Ethiopia, the current system for connecting students with tutors is highly informal and fragmented. Most students and parents rely on personal networks and word-of-mouth recommendations. In some cases, they refer to paper-based advertisements, such as flyers or school bulletin boards to find available tutors.

Tutors face similar challenges. They also depend on traditional methods like referrals or printed ads to find students, which makes it hard to reach a wider audience or establish a reliable income stream. This informal system results in inefficient matching and limited access to qualified instructors.

While a few digital platforms have emerged to modernize this process, most are underdeveloped or dysfunctional. The existing systems considered to be functioning properly mainly focus on national exam preparation for Grades 1–12, with little support for university-level courses or language learners. Some of these websites still lack integrated core features, and users are redirected to telegram for key actions like booking sessions. Relying on telegram for main operations reduces the effectiveness of having a dedicated web platform.

Additionally, most platforms only offer one-on-one tutoring, with no support for group sessions, limiting their usability for many students and tutors.

1.2.2 Major problem of existing system

- **Fragmented Tutor-Student Matching:** Students struggle to find qualified tutors, while tutors lack a centralized platform to reach learners efficiently. Most connections happen through informal networks or paper-based methods, limiting opportunities.
- **Inefficient Session Management:** Current tutoring relies on manual scheduling (phone calls, texts), leading to mismatched availability, no-shows, and wasted time for both tutors and students.
- **Limited Digital Learning Support:** Many existing solutions do not integrate essential tools like flexible scheduling, attendance tracking, student assessments, quizzes, or video conferencing, making learning less effective.
- **No Standardized Tutor Evaluation:** Many existing solutions do not have a rating or feedback system, which enables students to evaluate their tutors and provides an additional way to choose the most suitable tutor, beyond just qualifications.
- **Rigid Learning Options:** Many platforms only offer either online or in-person tutoring, but not both, restricting flexibility for students who may need hybrid solutions.
- **Narrow Course Coverage:** Most existing platforms focus mainly on grade 1–12 exam preparation, neglecting the needs of higher education students and language learners.
- **Difficulty in Finding Qualified Tutors**: Lack of proper filtering or qualification verification makes it hard to find skilled tutors through informal networks or paper-based methods.

1.2.3 Proposed system

The proposed system is a tutoring platform designed to address the key challenges of the current fragmented system in Ethiopia by offering a modern, web-based solution that improves accessibility, efficiency, and learning outcomes. To resolve the difficulty students face in finding qualified tutors, the platform includes a comprehensive filtering system based on subject, language, tutor type, experience, rating , gender and price and ensuring more effective tutor-student matching. By verifying tutor qualifications and prioritizing experienced applicants, it addresses the problem of unreliable tutor credentials.

To overcome inefficient session management, the system integrates scheduling, booking, attendance tracking, and automated notifications that alert users to changes such as rescheduled sessions or new messages, reducing no-shows and miscommunication. The inclusion of chat and video conferencing tackles the lack of digital learning support, while offering both online and in-person tutoring options provides the flexibility that existing platforms lack. Supporting both group and one-on-one sessions further caters to diverse learning preferences.

Furthermore, built-in exams and assessments enable monitoring of student progress, directly addressing the absence of standardized evaluation tools. A secure payment system simplifies transactions, improving user experience. Finally, the rating and review system enhances tutor accountability and helps students make informed decisions, solving the problem of missing tutor evaluation mechanisms.

Overall, by directly targeting these previously identified issues, this intelligent and inclusive platform seeks to establish a reliable, organized, and equitable tutoring ecosystem for Ethiopian learners.

1.2.4 Significance of the proposed system

- **Bridges the Gap between Students and Qualified Tutors:** Provides a centralized, efficient system to connect students with verified and experienced tutors across various subjects and academic levels.
- **Improve Learning Outcomes**: Features like progress assessments, quizzes, and personalized matching help ensure more effective and personalized learning.
- **Increases Accountability and Quality:** Rating and feedback systems allow students to review tutors, promoting transparency and continuous improvement in teaching quality.
- **Reduce Paper-Based Systems**: The platform eliminates manual paperwork by digitizing tasks like scheduling, payments, attendance, quizzes and assessments, making the process faster and more user friendly.
- **Provides Secure and Seamless Payments:** Integrated payment options streamline transactions for both parties, eliminating the need for manual or cash-based payments.
- **Supports Flexible Learning:** enables both online and in-person tutoring, allowing users to choose based on their convenience and learning preferences.

1.3. Motivation

The motivation for this project arises from persistent challenges in Ethiopia's tutoring landscape. Currently, the system is fragmented, making it difficult for students to find qualified tutors and for educators to connect with learners. This lack of a centralized platform leads to inefficiencies, missed opportunities, and limited access to quality education.

Moreover, existing tutoring options lack flexibility and comprehensive digital support, which limits the overall effectiveness and user experience. Additionally, the absence of proper verification mechanisms affects the reliability and trustworthiness of the system.

This project is necessary to overcome these obstacles by creating a unified, user-friendly platform that simplifies tutor-student matching, supports multiple learning modes, and integrates essential digital tools. It aims to improve educational outcomes by making tutoring more accessible, reliable, and efficient for everyone involved. The solution’s potential to enhance learning quality and accessibility in Ethiopia justifies the commitment of resources and effort to its development.

1.4 Scope and limitation of the project

1.4.1 Scope of the project

The scope of this project encompasses the design and development of a comprehensive web-based tutoring platform aimed at connecting students with qualified tutors across Ethiopia. The platform will cater to learners from grade 1 through university, as well as individuals seeking language and foundational courses like English, offering both online and in-person learning options to accommodate diverse educational needs. It will feature core functionalities such as session scheduling, attendance tracking, video conferencing, assessment tools, and secure payment integration. The system is designed to ensure a seamless, interactive, and flexible learning experience while fostering accountability and transparency through tools like tutor ratings and automated notifications. By centralizing tutoring services in one accessible platform, the project aims to enhance educational accessibility, improve learning outcomes, and support the professional growth of tutors.

1.4.2 Limitation of the project

Despite the platform’s comprehensive approach, certain challenges remain that require long-term strategies to mitigate:

- **Internet Dependency:** Reliable connectivity is essential for online sessions, which may pose accessibility issues in rural areas.
- **Tutor Verification Challenges:** Ensuring accurate and trustworthy verification of tutor qualifications can be complex and time-consuming.
- **Limited Course Coverage:** Initially, the platform may offer only a limited selection of common language courses.
- **Language Support Limitations:** translation into other Ethiopian languages is currently limited.
- **Privacy & Security Risks:** Ensuring secure data management and preventing unauthorized access is critical for user trust.

1.5 Objective

1.5.1 General objective

The general objective of this project is to develop online and in person Tutoring Platform.

1.5.2 Specific objective

- To study the current state of tutoring services in Ethiopia, focusing on accessibility, tutor verification, and digital literacy challenges.
- To analyze existing local and international tutoring platforms and extract best practices applicable to the Ethiopian context.
- To identify key challenges faced by students and tutors, such as scheduling system, limited regional access, and poor communication.
- To gather functional and non-functional requirements from stakeholders, including students, tutors, and educational institutions.
- To design a responsive and user-friendly platform interface.
- To implement core features such as user registration/login, session booking, chat, video conferencing, secure payments, exam and progress tracking, tutor scheduling, and admin monitoring.
- To test the platform thoroughly to ensure usability, performance, security, and reliability across devices and regions.

1.6 Methodology

The methodology for this project involves a systematic approach to understanding, designing, and developing a web-based tutoring platform. It includes three key phases: data collection methodology, system design and analysis tools, and system development tools.

1.6.1 Data Collection Methodology

To ensure the platform addresses real-world educational challenges and user needs, a combination of research and stakeholder engagement methods were used during the data collection phase. The following approaches were applied:

1. Online Research

We conducted comprehensive online research to evaluate the current state of tutoring services in Ethiopia and to explore international best practices.

More than 10 Ethiopian platforms including A+ Online Tutors, TeachOn, Teach Ethiopia, Fidel Tutorial, Ethio IQ, Kelem Tutors, Addis Tutor, Apperntus, Ethio College Prep, and Lesson Pal were examined to identify common challenges.

In addition, leading international private tutoring marketplaces such as Preply, Wyzant, Superprof, MyTutor, TutorMe, iTalki, HeyTutor, Brainfuse, and Tutor.com were analyzed for effective design strategies. The insights gained from this research directly influenced the design of our platform, ensuring it addresses local needs while aligning with globally successful solutions.

2. Observation

During the research phase, direct observations were conducted by reviewing how students and tutors interact with existing tutoring services in Ethiopia. It was noted that many students rely heavily on informal networks and paper-based advertisements to find tutors, resulting in limited reach and inconsistent quality. Tutors often depend on word-of-mouth referrals or school bulletin boards, which restricts their ability to connect with a wider student base.

Additionally, the use of digital platforms is minimal and, where present, core functionalities are fragmented or shifted to external apps like Telegram. There is a lack of integrated tools for scheduling, communication, and payment, leading to inefficiencies and missed opportunities for both students and tutors. These observations underscore the need for a centralized, user-friendly, and digitally integrated platform that can streamline the tutoring process and enhance accessibility and reliability.

3. Interview

Information was gathered through interviews conducted with representatives from 7 educational institutions, including schools, colleges, and universities. to understand their needs and challenges with tutoring services. These insights helped shape the platform to align with educational standards and institutional workflows.

1.6.2 System Design and Analysis Tools

Architecture Design

The system architecture was modeled using UML diagrams, including use case, activity, and sequence diagrams to represent key workflows such as session booking, conducting sessions, and login/registration.

Agile development approach

An agile approach was followed, with user stories prioritized in Jira and weekly sprint backlogs refined based on stakeholder feedback. This ensured flexibility, continuous improvement, and user-centered development.

1.6.3 System Development Tools

The system was developed using a modern and robust technology stack, ensuring scalability, performance, and ease of maintenance. The main tools and technologies used include:

Backend: Next.js

- Next.js was chosen for its hybrid rendering capabilities (both server-side and client-side), built-in routing, and excellent performance optimization. It also integrates smoothly with React and supports API routes, reducing the need for a separate backend framework.

Frontend: React

- React was selected for its component-based architecture, reusability, and large community. It enables fast and interactive user interfaces, which is ideal for building a responsive tutoring platform.

Database: MongoDB

- MongoDB, a NoSQL database, was chosen for its flexibility in handling unstructured and semi-structured data, such as user profiles, messages, and session records. It allows for rapid development and scaling, making it suitable for evolving systems like this platform.

Third-Party Services: The project integrated third-party service:

- **Payment Integration:** Chapa payment gateways were integrated to ensure secure, reliable, and seamless financial transactions for both students and tutors.

Development Methodology

The project followed the Agile Scrum framework, working in short iterative sprints to incrementally design, implement, test, and refine features based on feedback and evolving requirements.

1.7 Feasibility of the project

1.7.1 Technical Feasibility

The project is technically feasible due to the use of proven technologies such as the Next.js for backend development and React for the frontend, which together support building a robust and scalable tutoring platform. Integration with payment gateways like Chapa is supported by existing APIs, while tools like Git facilitate efficient version control and collaboration. The system's secure authentication mechanisms further ensure technical reliability and accessibility.

1.7.2 Operational Feasibility

Operationally, the platform is designed to meet the needs of students and tutors by providing an intuitive and efficient management tools such as scheduling, session tracking, and communication. Its scalable architecture allows for handling increasing user demands, while role-based access controls maintain security and smooth workflow. Institutional feedback incorporated during design enhances its alignment with educational practices, supporting long-term usability and effective operation.

1.7.3 Economic Feasibility

The economic feasibility is promising, with initial development costs minimized by leveraging open-source technologies like Next.js, React, and MongoDB. The platform’s revenue model includes commissions from tutoring sessions, providing sustainable income. Supporting both private and group sessions caters to various budgets, expanding the user base. These factors combined suggest the platform can maintain economic sustainability while offering affordable access to quality education services across Ethiopia.

Chapter Two: System Requirement Specification

2.1 Background Overview

This tutoring platform aims to provide accessible, flexible, and high-quality educational support for students from grade one to university level in Ethiopia by integrating both online and in-person tutoring services. It is designed around well-defined functional requirements tailored to four core user roles i.e. ETN, admin, student, and tutor enabling operations such as account management, session booking, video conferencing, progress tracking, and secure payments. The system also meets essential non-functional requirements, ensuring data security, high performance, and scalability. The platform is technically feasible using technologies like Next.js, React, and MongoDB operationally viable with scheduling, and role-based access and economically sustainable due to low development costs and a commission-based revenue model. Overall, the platform seeks to transform tutoring in Ethiopia by offering interactive and reliable learning experience.

2.2 Functional Requirements

Functional requirements define the specific behaviors and capabilities that the Tutoring Platform must support. These requirements outline the core features that allow ETN, students, tutors, and administrators to interact with the system effectively and achieve their goals.

The system functionalities are listed as follow:

- The system should allow students and tutors to register by providing personal information.
- The system should allow all users to log in and log out securely.
- The system should allow users to manage their profile
- The system should allow students to search for courses or tutor based on subject
- The system should allow students to book tutoring sessions with available tutors.

- The system should enable students to make secure payments through the platform.
- The system should allow students to take exams and view their results.
- The system should allow students to view their attendance records.
- The system should enable students to access shared learning materials uploaded by tutors.
- The system should allow students to rate tutors
- The system should allow students and tutors to communicate via chat.
- The system should allow students to attend both online and in-person sessions.
- The system should provide notifications to students and tutors about upcoming sessions and important updates.
- The system should allow tutors to set their availability schedule and reschedule sessions when necessary.
- The system should enable tutors to track student attendance during sessions.
- The system should allow tutors to view their session bookings and student feedback (ratings).
- The system should enable tutors to conduct online sessions using integrated video conferencing tools.
- The system should allow tutors to share learning materials with students.
- The system should enable tutors to prepare exams for students and monitor their performance.
- The system should allow the admin to manage users by adding, view or deactivating user accounts.
- The system should allow the admin to view attendances, material uploaded and ratings submitted by students.
- The system should allow the admin to receive and accept reports submitted by student.
- The system should enable ETN (external verifier) to verify submitted tutor documents.
- The system should enable the admin to make payments to ETN after successful verification.
- The system should provide students, admin and tutors with access to view transaction history.

2.3 Non-Functional Requirements

Security

- The system blocks unauthorized users from accessing secured pages.
- Login authentication is required, with strong password rules (uppercase, lowercase, numbers, special characters).
- Passwords are hashed using the bcrypt algorithm, which is preferred for its built-in salting and resistance to brute-force attacks, offering stronger protection than general-purpose algorithms like SHA or MD5.

Performance

- The system should minimize errors and display clear error messages that guide users.
- There shall be various ways of retrieving data, and it shall take less time.
- The system should handle multiple users simultaneously without significant delays.
- The performance of our website will be measured using key metrics such as uptime, responsiveness across devices, user engagement (including session duration and bounce rate), and error rates.

Portability: The software shall work properly in any browser.

Usability

- The application interface should be user-friendly, intuitive, and accessible.
- The end user should be able to access any page quickly, depending on internet connection speed.

Availability: the software should be available to anyone with an internet connection.

- The system should be available 24 hours a day, 7 days a week.

Maintainability: After deployment, if any error occurs, it should be easily maintained by the software developer.

Scalability: The platform must efficiently support increasing users, expand course offerings, and serve multiple institutions across Ethiopia without performance decline.

Reliability: The system must be reliable, with minimal downtime and robust error handling.

Chapter Three: Requirement Analysis and modeling

3.1 Overview

This chapter presents the requirement analysis and system modeling of the Tutoring Platform, an online system designed to support interactive learning between students and tutors. Using scenario-based, behavioral, and class-based modeling techniques, it explores how users and system components interact in various real-world scenarios. Key use cases and actors are identified to define user roles and system functionality, while activity, sequence, and state diagrams illustrate system workflows and dynamic behavior. Class modeling captures the core structural components and their relationships. Together, these models offer a comprehensive understanding of the system's architecture, data flow, and interactions, forming a solid foundation for designing a scalable and efficient tutoring platform.

3.2. Scenario Based Modeling

3.2.1 Use case identification

In the use case identification phase, we focus on identifying the key functionalities the Tutoring Platform must support to meet the needs of its primary users: students, tutors, administrators, and the ETN. For students, the system should allow them to register and log in, search for tutors, book tutoring sessions, make payments, attend sessions, and access shared learning materials. Additionally, they should be able to take exams, view their results, track attendance, manage their profiles (including updating or deleting their accounts), rate tutors, receive notifications and communicate through chat.

Tutors, on the other hand, should be able to register, log in, and submit documents for verification. Once verified, they can set their schedules, view session bookings, conduct tutoring sessions, and share learning materials. They are also expected to track student attendance, prepare exams, participate in chats, and receive notifications. Tutors should be able to manage their profiles by viewing or updating their accounts, as well as they can view the ratings they receive.

Administrators are responsible for maintaining system integrity. Their use cases include logging in and out, managing user accounts (students, tutors, ETN), viewing tutor ratings, accepting reports, making payments to the ETN and overseeing general platform operations to ensure compliance and quality. The ETN's primary responsibility is verifying tutor documents submitted through the platform to ensure authenticity and quality of instruction.

Identifying these use cases ensures that the platform addresses all critical interactions required to support structured, effective, and user-centered educational engagement among students, tutors, administrators, and external verifier.

3.2.2 Actor identification

Understanding the actors in the Tutoring Platform is critical to grasping how users interact with the system. Each actor represents a distinct role with specific responsibilities that define their engagement with the platform.

Primary Actors:

- **Student**: A learner who engages with the platform to enhance their academic experience.
- **Tutor**: An educator responsible for delivering academic sessions and supporting students.

Secondary Actors:

- **Admin**: The system administrator overseeing platform integrity and managing all user-related operations.
- **ETN:** Assists with tutor document verification.

3.2.3 Use case description

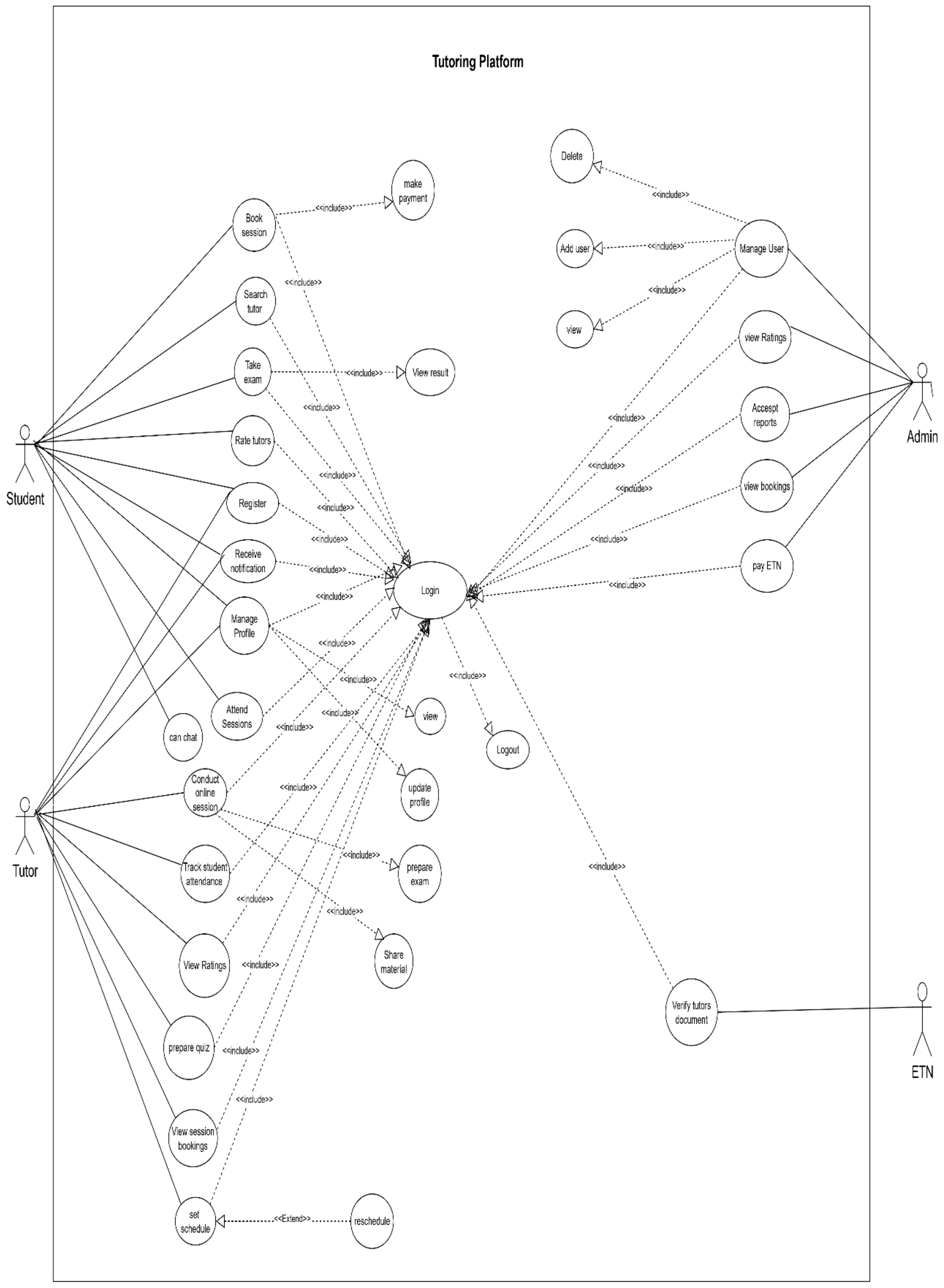


Figure 3.1 Use case Diagram

Table 3.1: System Use Case for Register as Student

Use case Name	Register
Participating Actors	Student
Flow of events	1. User clicks on “Sign Up” or “Register.” 2. The System presents the registration form. 3. The student fills in required details (full name, email, password, etc.). 4. The System validates the input data. 5. The student submits the registration form. 6. Then a new student account is created and confirmation email is sent to the user. 7. The student confirms the email and logs into the platform.
Entry condition	User accesses the platform and is not logged in.
Exit Condition	• Student account is successfully created and activated. • Student can now log in and access student features.

Table 3.2: System Use Case for Book Tutoring Session

Use case Name	Book Tutoring Session
Participating Actors	Student
Flow of events	1. Student searches for tutors by subject, availability, experience, or rating. 2. The System displays a list of available tutors matching the criteria. 3. Student selects a tutor and views their profile and clicks “Book Session.” 4. Student selects a preferred timeslot and continue to make payment. 5. The System presents a payment form. 6. Student completes payment. 7. The student receives a notification confirming the session enrollment.
Entry condition	Student is logged in.
Exit Condition	Tutoring session is successfully booked and scheduled.

Table 3.3: System Use Case for Conduct Online Tutoring Session

Use case Name	Conduct Online Tutoring Session
Participating Actors	Tutor Student
Flow of events	1. Tutor and Student access the online session link on the platform. 2. The System establishes a secure video connection. 3. Tutor conducts the tutoring session, sharing materials or screen as needed. 4. Student interacts, asks questions, and participates in the session and chat. 5. The System tracks session duration and attendance. 6. After the session ends, the tutor store student attendance records.
Entry condition	Student and tutor have a scheduled session. Both are logged in and available at the session time.
Exit Condition	• Session is completed and attendance is recorded. • Progress and feedback are saved in the system.

Table 3.4: System Use Case for Manage Tutor Profile

Use case Name	Manage Tutor Profile
Participating Actors	Tutor
Flow of events	1. Tutor navigates to “Profile page” 2. The System displays the tutor’s current profile information. 3. Tutor updates name, bio, tutoring type and profile photo. 4. Tutor submits the updated information. 5. The System validates inputs and saves the updates. 6. The System confirms successful profile update.
Entry condition	Tutor is logged in.
Exit Condition	• Tutor’s profile is updated and visible to students. • Tutor receives confirmation of the update.

Table 3.5: System Use Case for Manage Users

Use case Name	Manage Users
Participating Actors	Admin
Flow of events	1. Admin navigates to “User” section. 2. The System displays a list of all platform users (students, tutors, ETN). 3. Admin selects a user to view, deactivate or delete. 4. Admin views user details or delete the account if needed. 5. The System validates and updates the database accordingly. 6. Admin confirms the changes.
Entry condition	Admin is logged in and has access to user management features.
Exit Condition	• User information and statuses are updated as intended. • System confirms successful changes.

Table 3.6: System Use Case for Search Tutors

Use case Name	Search Tutors
Participating Actors	Student
Flow of events	1. Student accesses the “Tutors” section. 2. The System presents available tutors 3. System provides filters (subject, gender, language, experience, availability, ratings). 4. Student applies filters and initiates the search. 5. The System retrieves and displays a list of tutors matching the criteria. 6. Student browses tutor profiles and selects one or more details.
Entry condition	Student is logged in.
Exit Condition	• Student views detailed tutor profiles. • Student can proceed to book a session if desired.

Table 3.7: System Use Case for Ratings

Use case Name	Ratings
Participating Actors	Student
Flow of events	1. Student books a session with a tutor 2. Student attends the session and submits ratings. 3. The System stores the rating and updates the overall rating given by the student. 4. The System displays a confirmation of successful submission.
Entry condition	Student has completed a tutoring session and is logged in.
Exit Condition	• Rating is saved and reflected on student dashboard. • Student receives confirmation of submission.

3.2.4 Activity diagram

Activity diagram describes the behavior of a system in terms of activities. These activities represent the execution of a set of operations. The completion of these operations then triggers a transition to another activity. Essentially, an activity diagram is a type of state chart diagram where the states are considered "action states".

Activity diagrams also show alternative transitions based on conditions, which are depicted using decision nodes (a diamond shape) with labeled outgoing arrows representing different branches in the control flow. They are primarily used to model workflows, business processes, or the internal operations of a system.

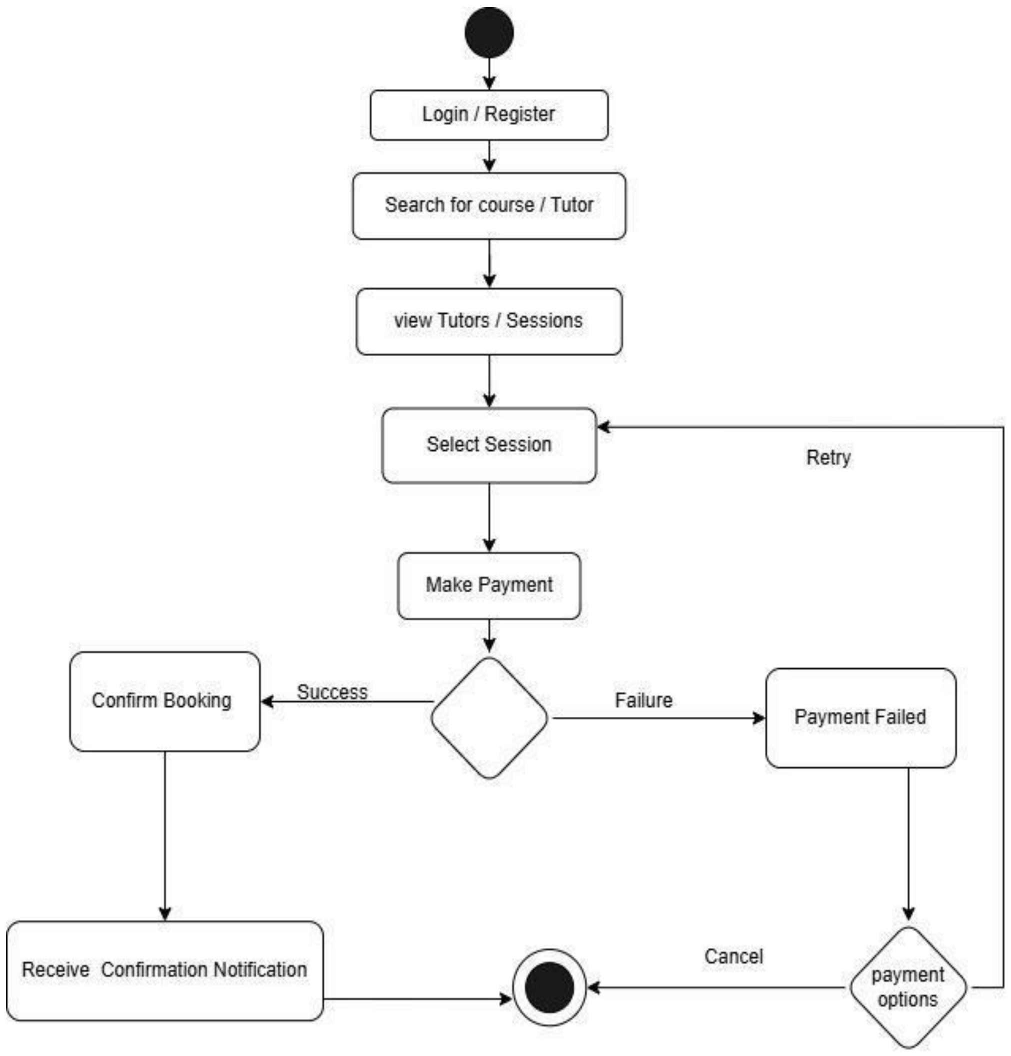


Figure 3.2: Activity Diagram for Student book a session

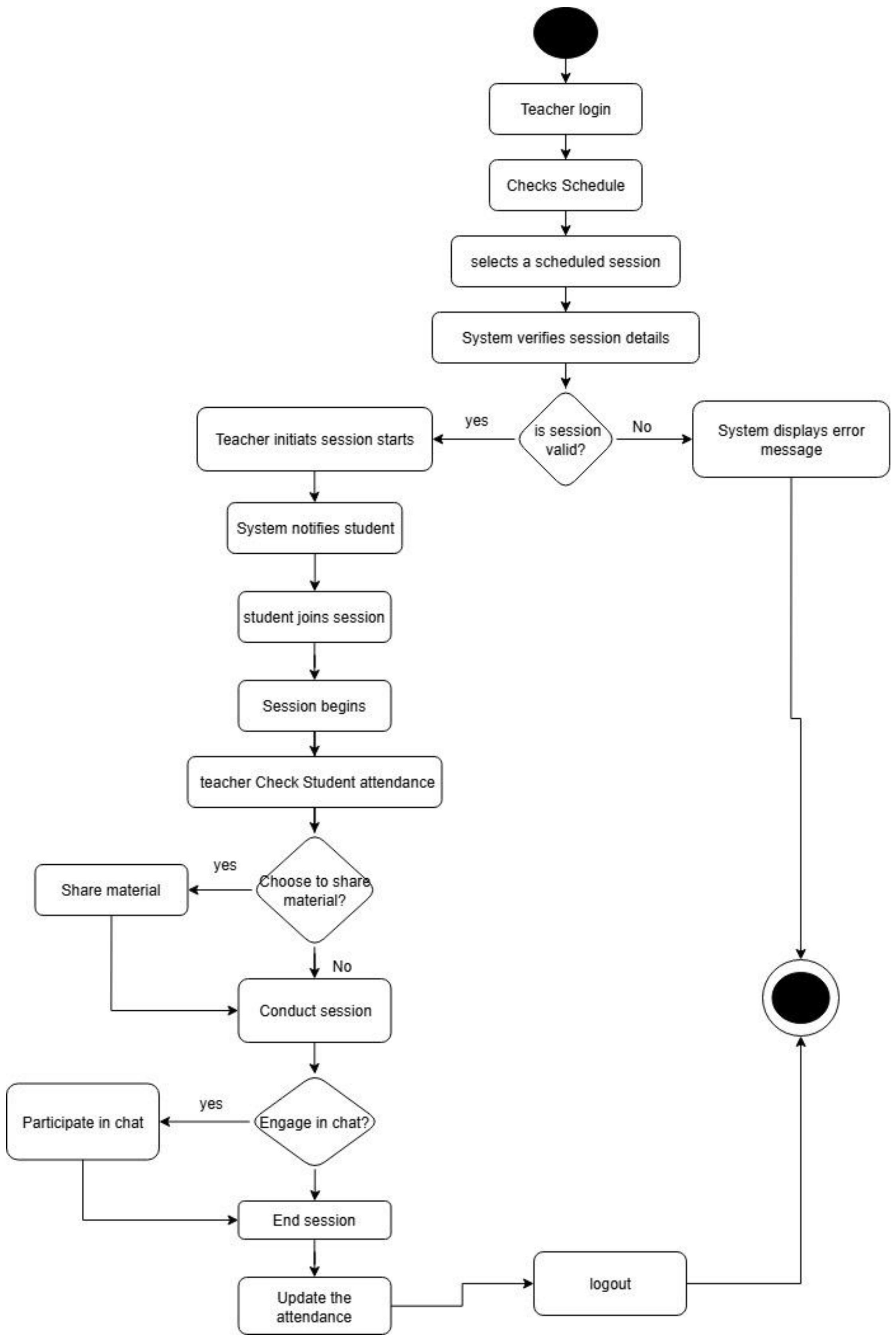


Figure 3.3: Activity diagram for tutor conduct online session

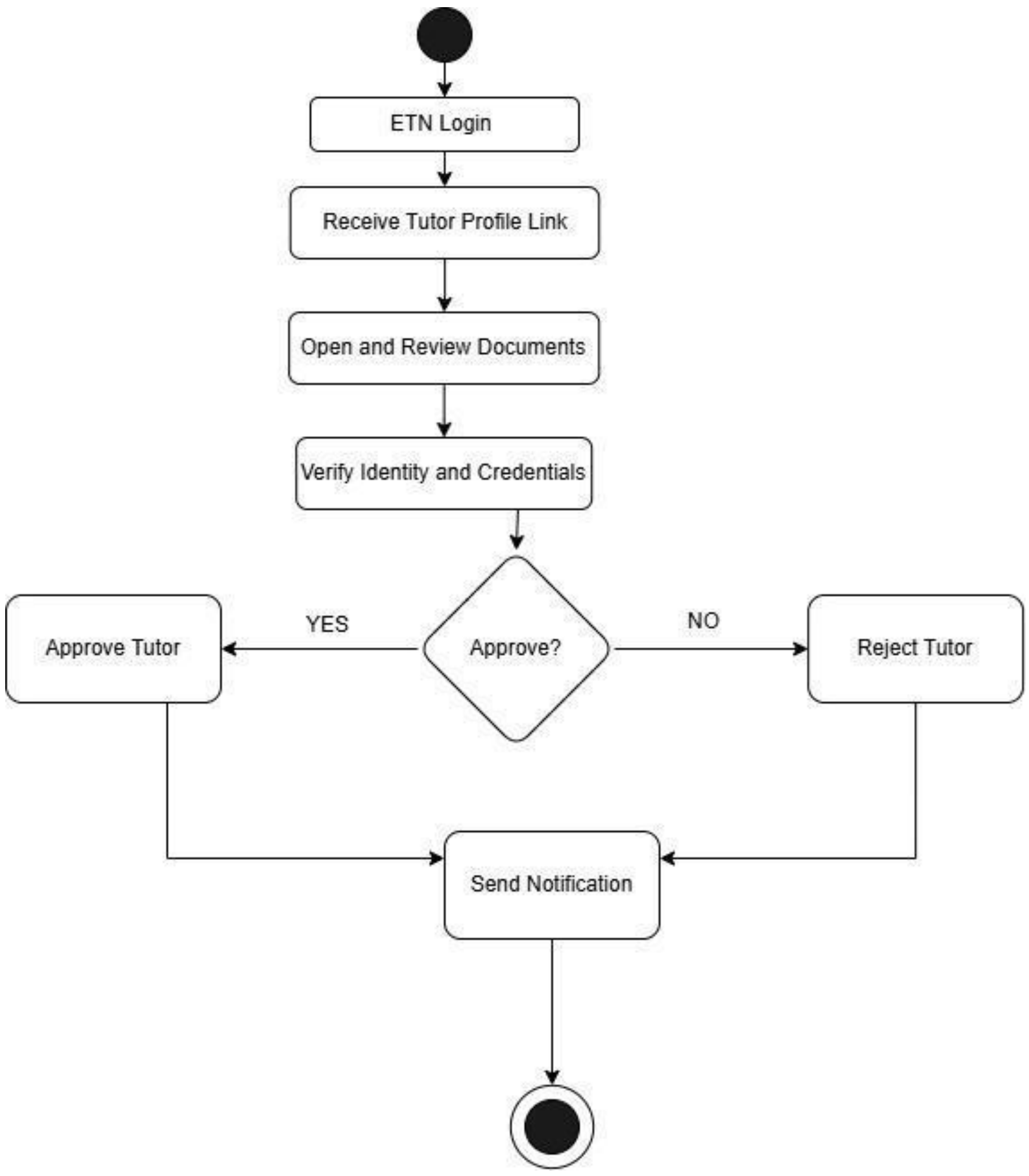

ETN Login
Receive Tutor Profile Link
Open and Review Documents
Verify Identity and Credentials
Approve?
YES
NO
Approve Tutor
Reject Tutor
Send Notification

Figure3.4: Activity diagram for ETN tutor verification

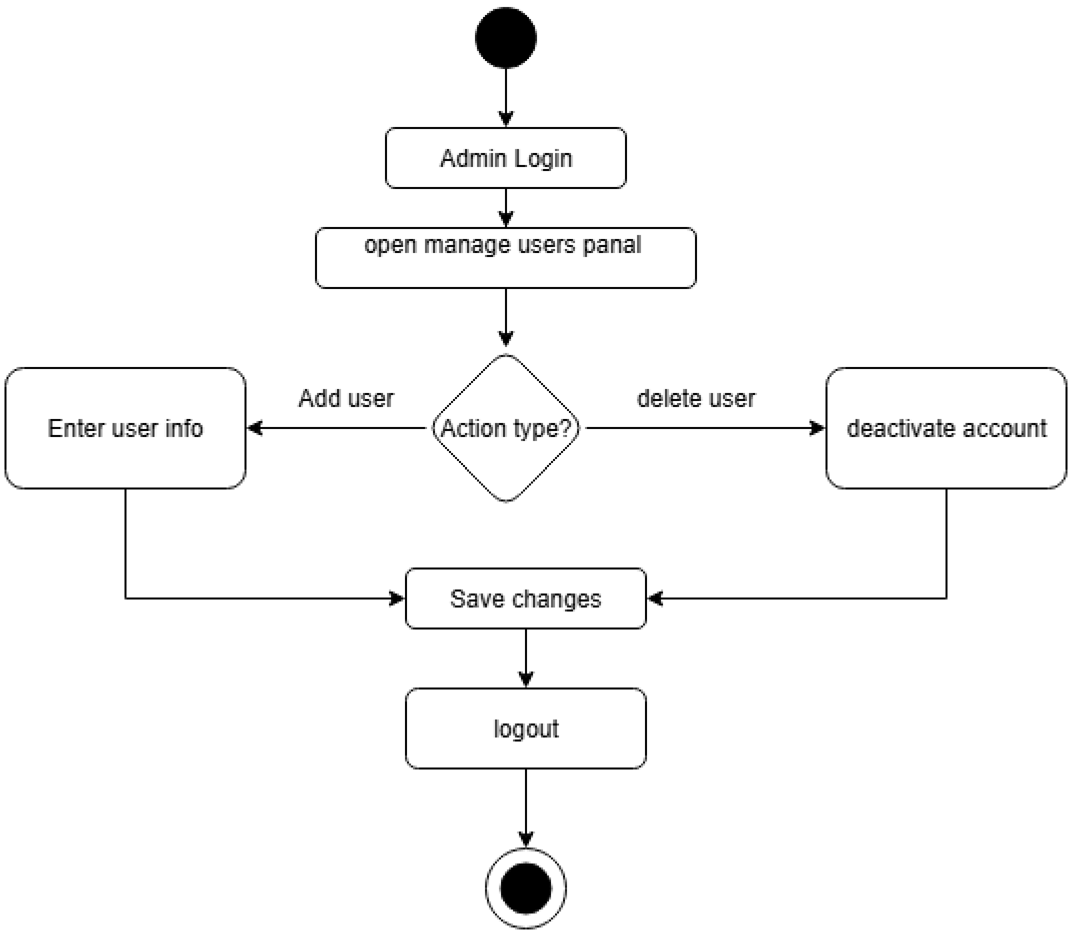


Figure 3.5: Activity diagram Admin manages user

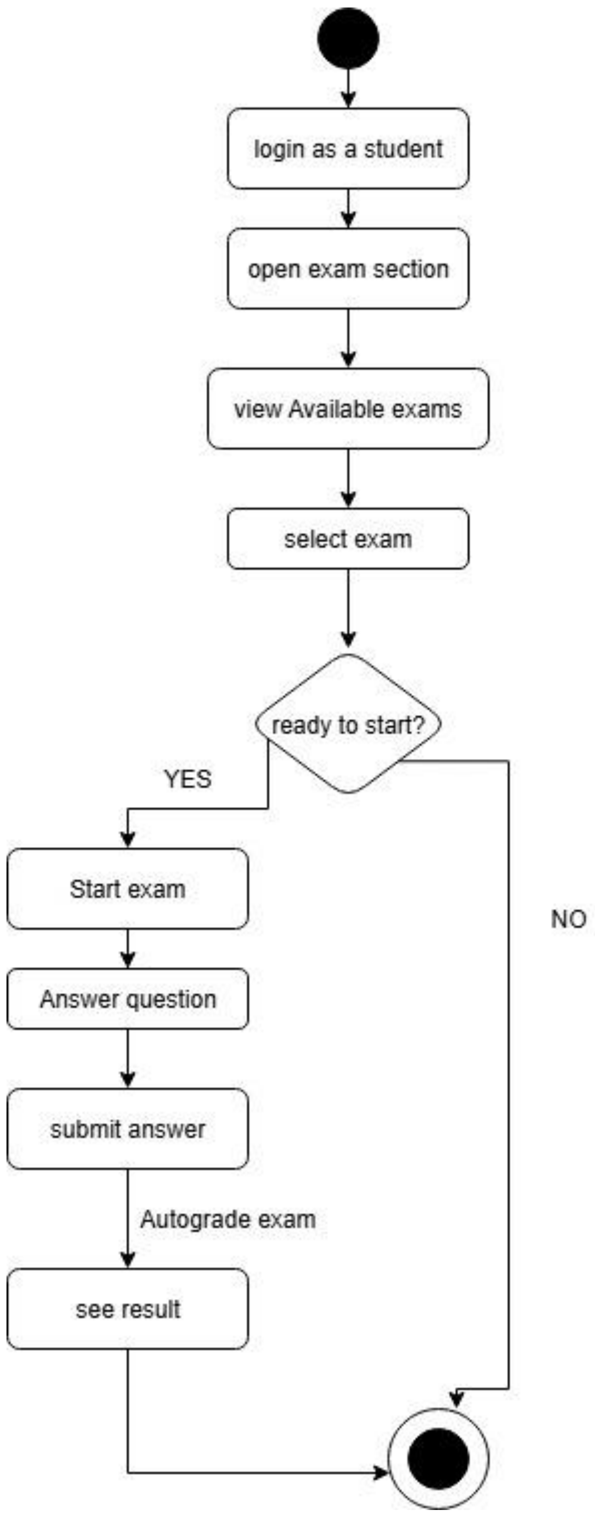


Figure 3.6: Activity diagram for Student takes an exam

3.3. Behavioral/Dynamic Modeling

Unlike structural modeling, which focuses on the static aspects of a system (like classes and objects), behavioral modeling captures how the system responds to various inputs, how different components interact, and how the system's state changes over time. Its main purpose is to help understand, design and refine the system behavior, ensuring it performs correctly, interacts efficiently and meets the requirements.

3.3.1. Sequence diagram

Sequence diagrams help visualize the dynamic behavior and interactions within the software system. They are particularly useful for detailing how objects communicate with each other over time to achieve a specific functionality.

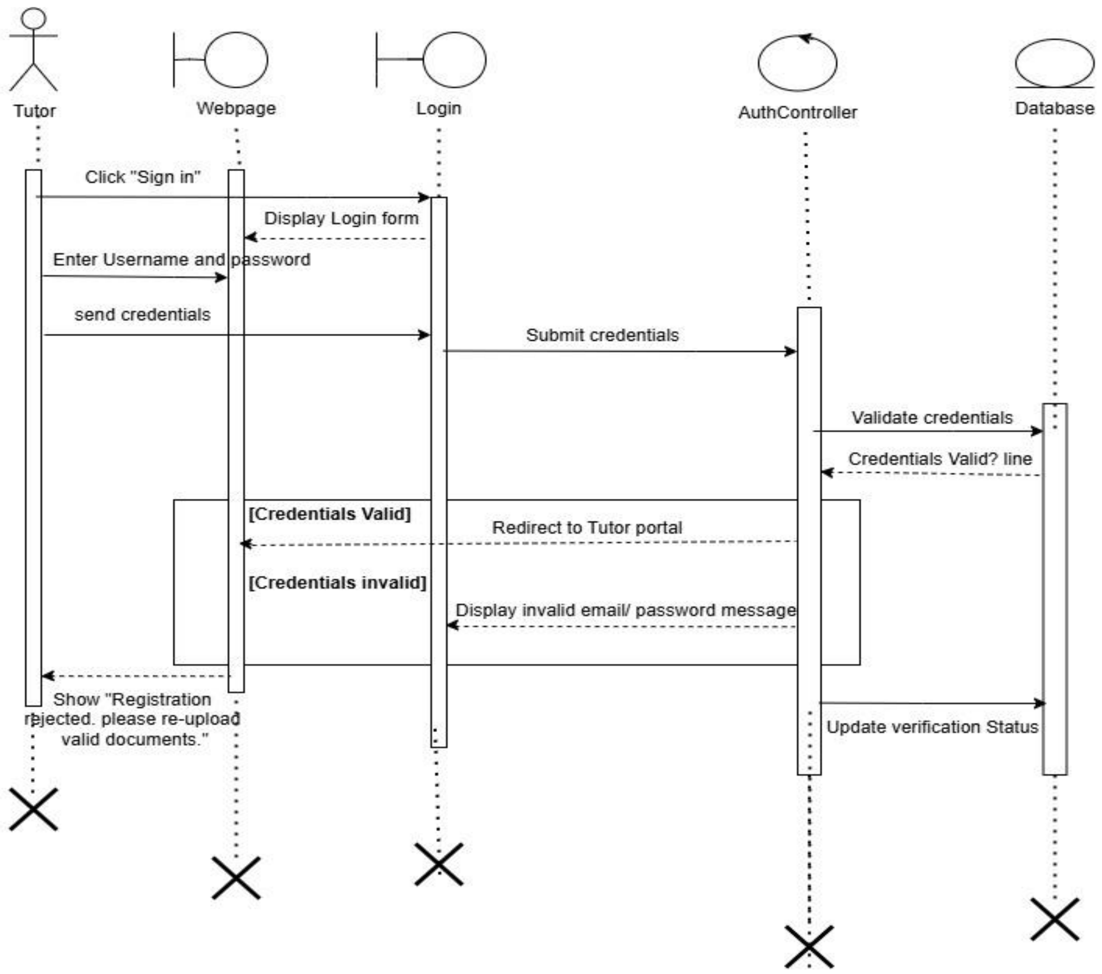


Figure 3.7: Sequence diagram Tutor login

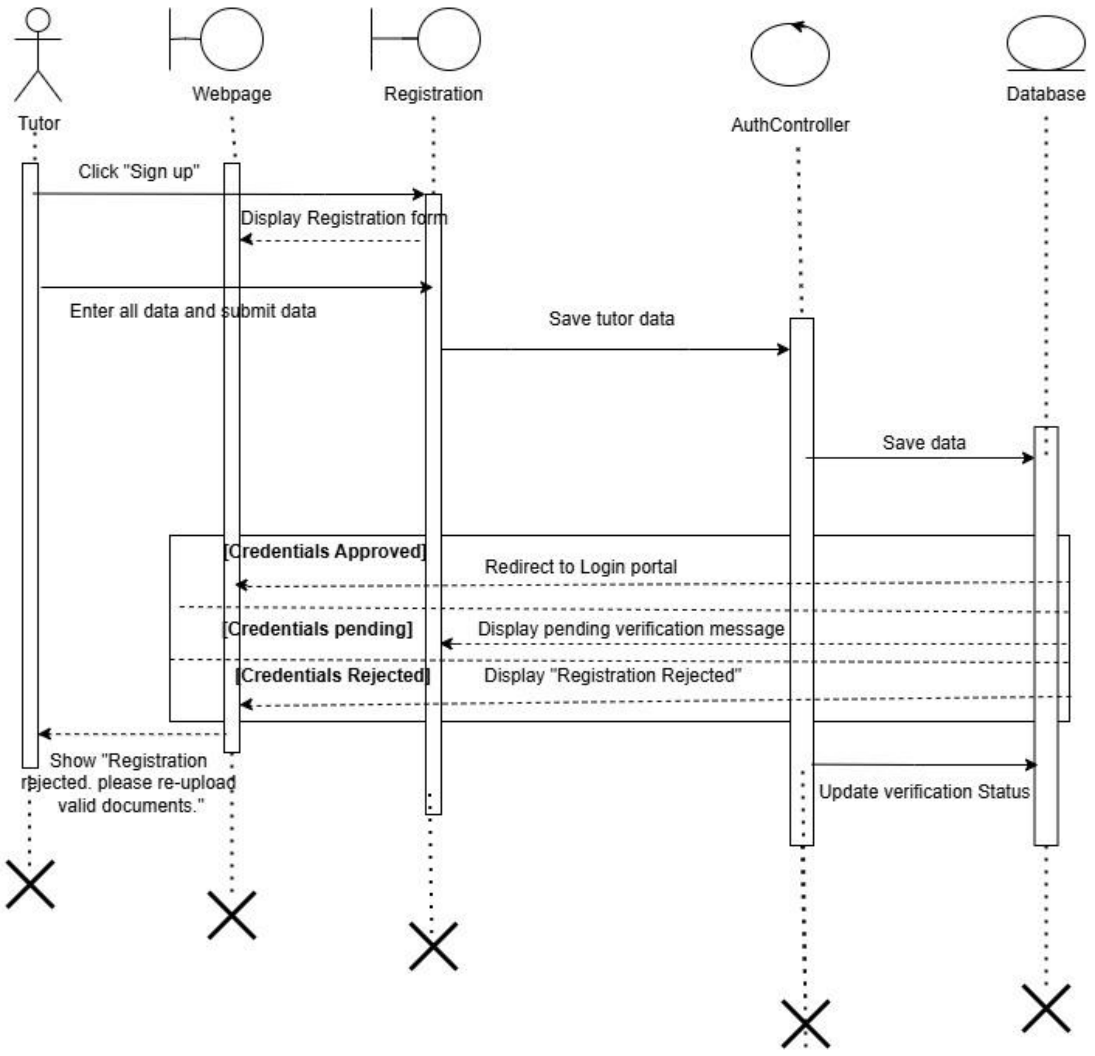

Tutor
Webpage
Registration
AuthController
Database
Click "Sign up"
Display Registration form
Enter all data and submit data
Save tutor data
Save data
[Credentials Approved]
Redirect to Login portal
[Credentials pending]
Display pending verification message
[Credentials Rejected]
Display "Registration Rejected"
Show "Registration rejected. please re-upload valid documents."
Update verification Status

Figure 3.8: Sequence diagram for Tutor registration and verification

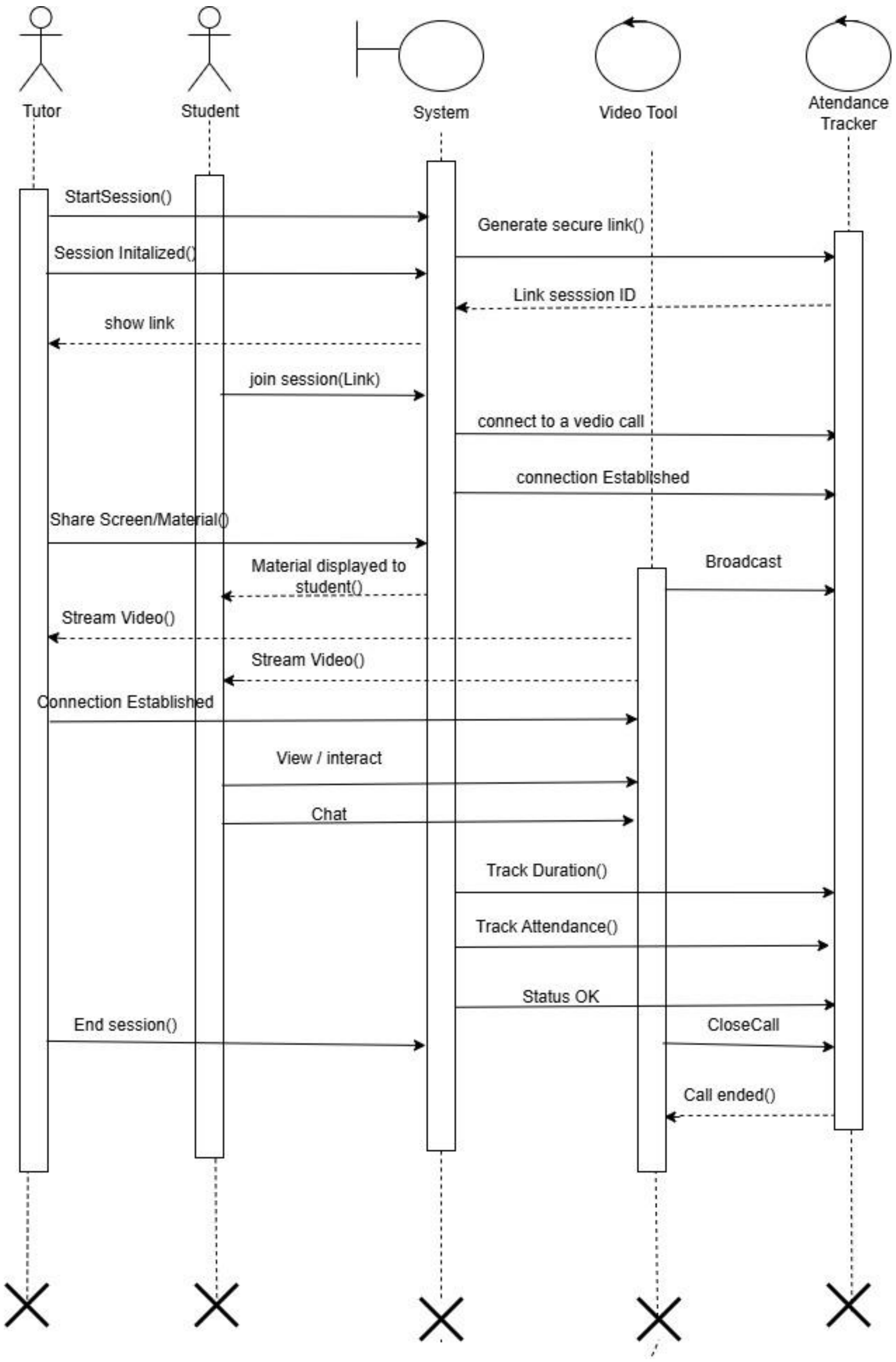


Figure 3.9: Sequence diagram for online session tutor and student

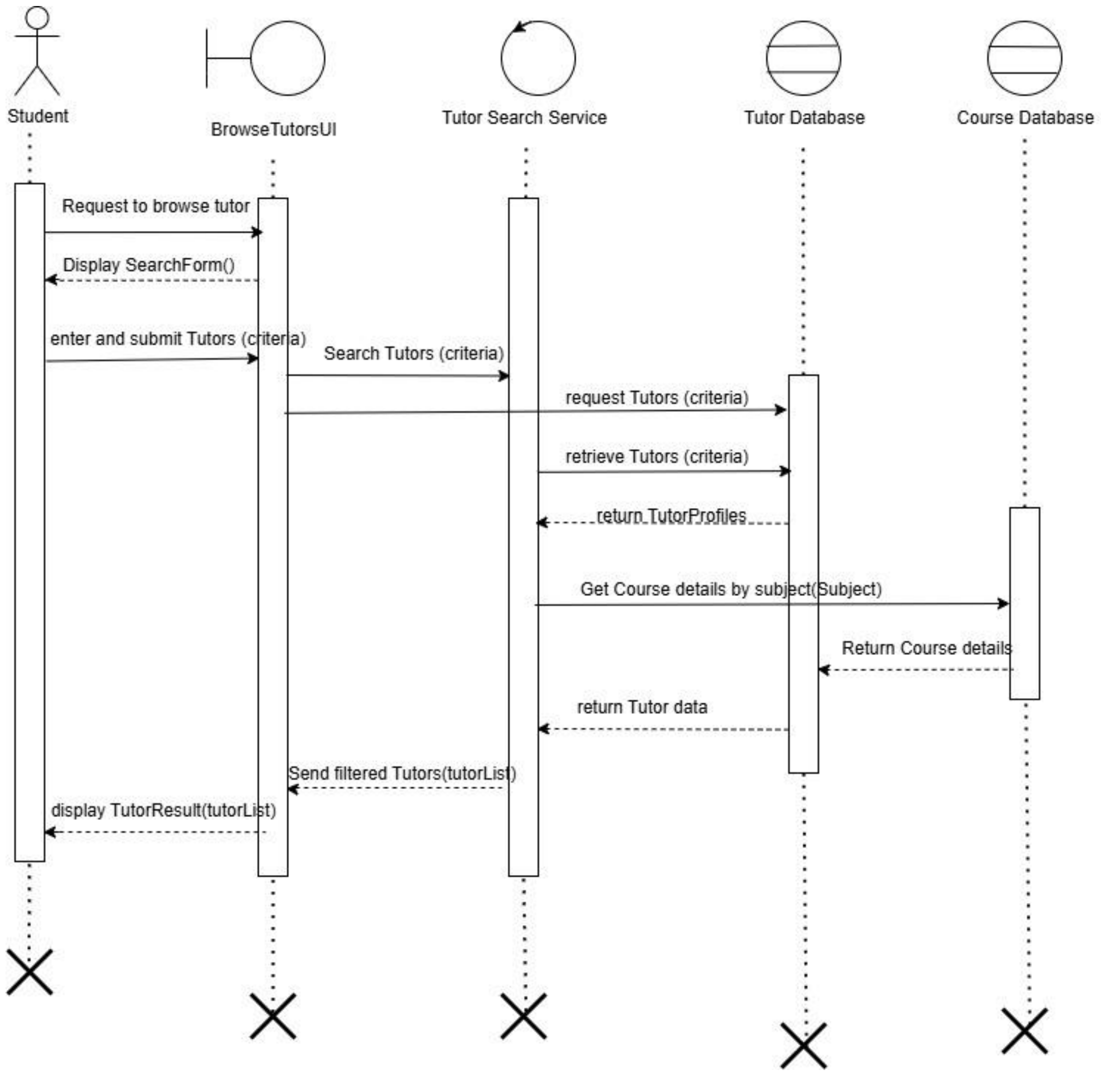


Figure 3.10: Sequence diagram for student searches for tutor

3.3.2. State Diagram

State diagram is a UML behavioral diagram that is used to describe the dynamic behavior of an individual object as it transitions through different states. It illustrates how the system behaves and responds by showing "a particular set of values for an object" (a state) and the conditions associated with the change of state.

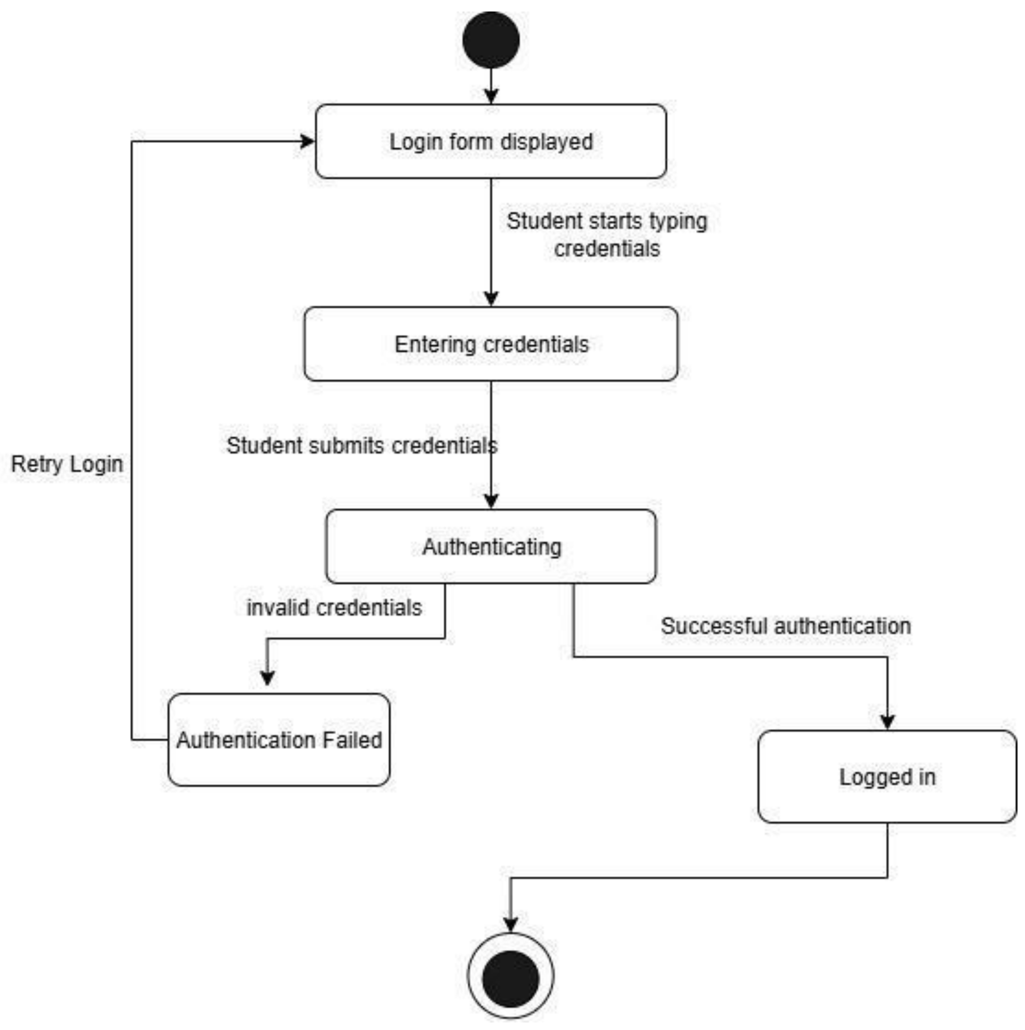


Figure 3.11: student login state diagram

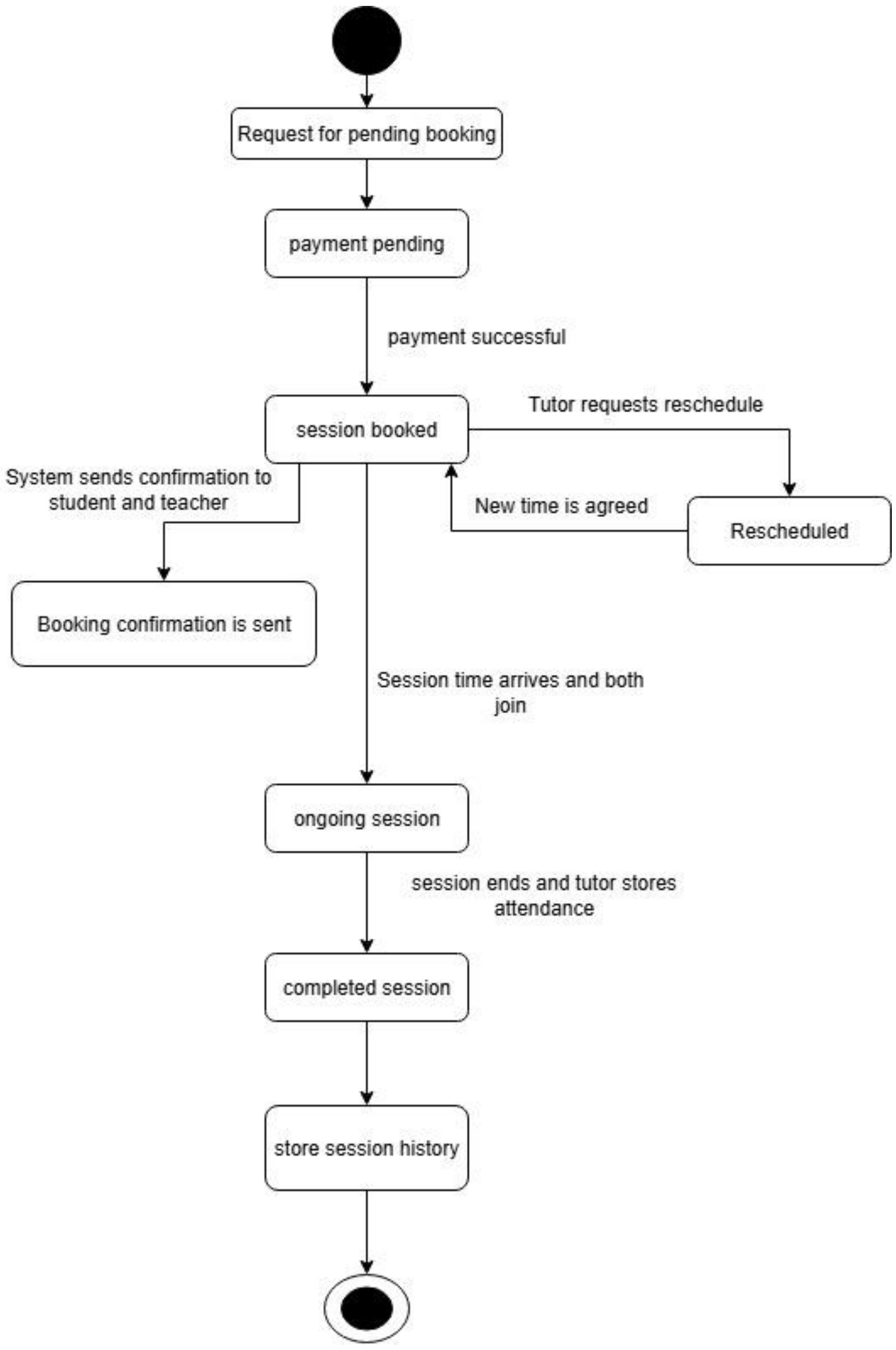

Request for pending booking
payment pending
payment successful
session booked
Tutor requests reschedule
Rescheduled
New time is agreed
System sends confirmation to student and teacher
Booking confirmation is sent
Session time arrives and both join
ongoing session
session ends and tutor stores attendance
completed session
store session history

Figure 3.12: Tutoring session state diagram

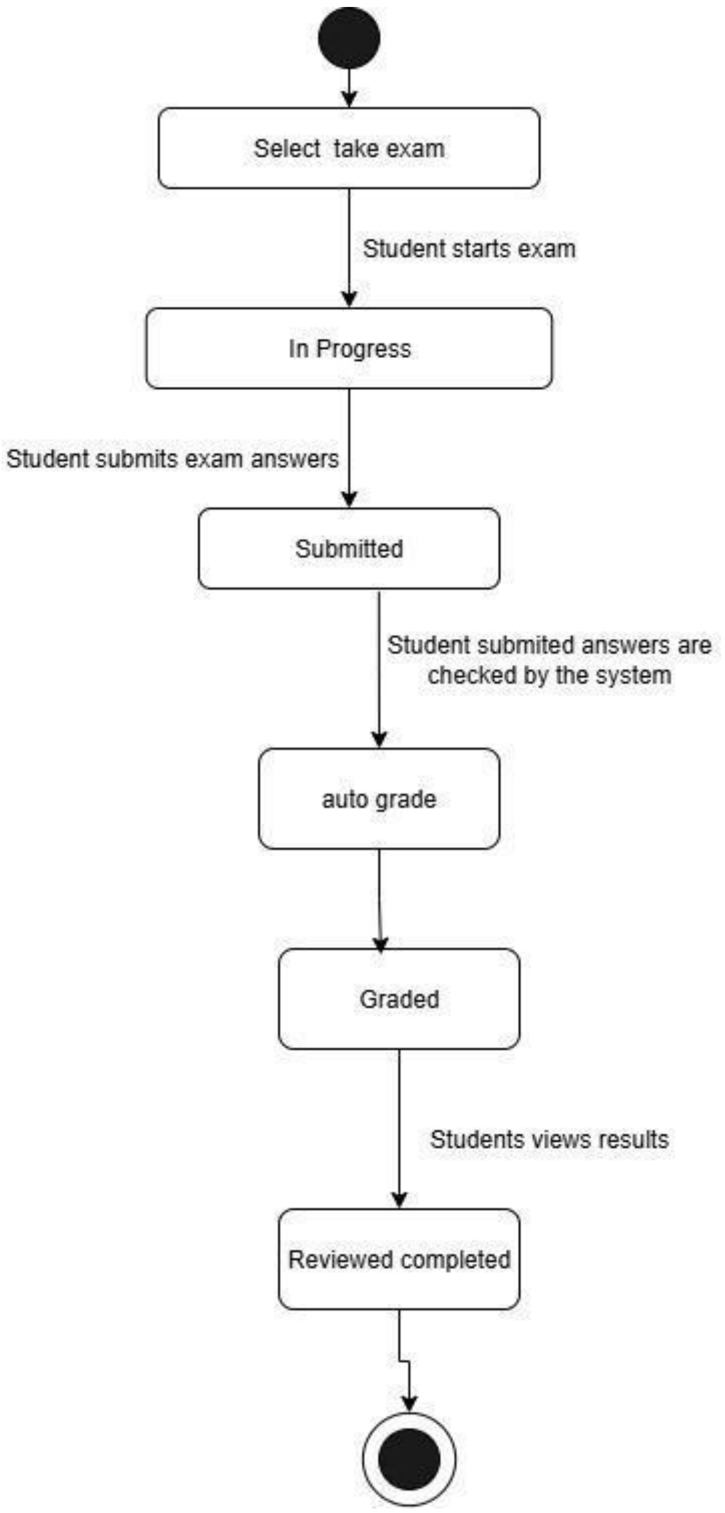


Figure 3.13: students taking a quiz state diagram

3.4. Class-Based Modeling

Class-based modeling is a core approach in object-oriented design that defines the system’s structure and behavior through classes. In the tutoring platform, it captures key entities such as users, courses, sessions, and communication features, with each class encapsulating its own attributes and methods. The model also represents various relationships among these classes, ranging from tightly bound dependencies to more flexible associations. This structured representation supports a modular and well-organized system architecture, making the platform scalable, maintainable, and adaptable to future enhancements.

3.4.1. Identifying classes

User

General class for all types of users (students, tutors, admins)

- **Attributes:** id, first_name, last_name, email, password, phone_no, gender, profile_pic, address
- **Methods:** register(), login(), logout(), manageProfile()

Student

Represents users looking for tutoring services

- **Attributes**: preferredLearningMode()
- **Methods**:bookSession(),searchTutor(),viewAttendance(), receiveNotification(),viewMaterial(), takeQuiz(), makePayment(), rateTutor(), attendSession(),viewresult(),participateInChat()

Tutor

Represents users providing tutoring services

- **Attributes**: cv, educationLevel, subjectTaught, availability, teachingMode
- **Methods**: conductSession(), viewProfile(), updateProfile(),

uploadMaterial(),trackStudentAttendance(),,receiveNotification(), viewratings(),viewSessionBookings(),participateInChat()

Admin

System administrator with highest privileges

- **Methods**: addUser(), viewProfiles(), deactivateAccounts(), viewRatings(), view analysis(), view bookings(), accept reports(), pay ETN()

Session

Represents a tutoring session

- **Attributes**: session_id, tutor_id, student_id, session_time, subject
- **Methods**: startSession(), endSession()

Rating

Student feedback and review of a tutor

- **Attributes**: rating_id, ratedby, score
- **Methods**: submit()

Attendance

Tracks student presence in a session

- **Attributes**: attendance_id, session_id, student_id

Notification

System messages sent to users

- **Attributes:** notification_id, recipient_id,
- **Methods**: send(), view()

Exam

Career test or topic evaluation for students

- **Attributes:** exam_id, student_id, time, result
- **Methods**: take(), viewresult(), retake()

Payment

Handles payment transactions between student and platform/tutor

- **Attributes:** payment_id, student_id, amount, method, date, tutor_id
- **Methods**: process(), cancel()

Material

Learning resources shared by tutors

- **Attributes**: material_id, title, file
- **Methods**: upload(), download(), delete()

Chat

Direct messaging between student and tutor

- **Attributes:** chat_id, sender_id, recipient_id, session_id
- **Methods**: sendMessage(), deleteMessage()

ETN

- **Attributes:** etn_id, name
- **Methods:** verify()

3.4.2. Class Diagram

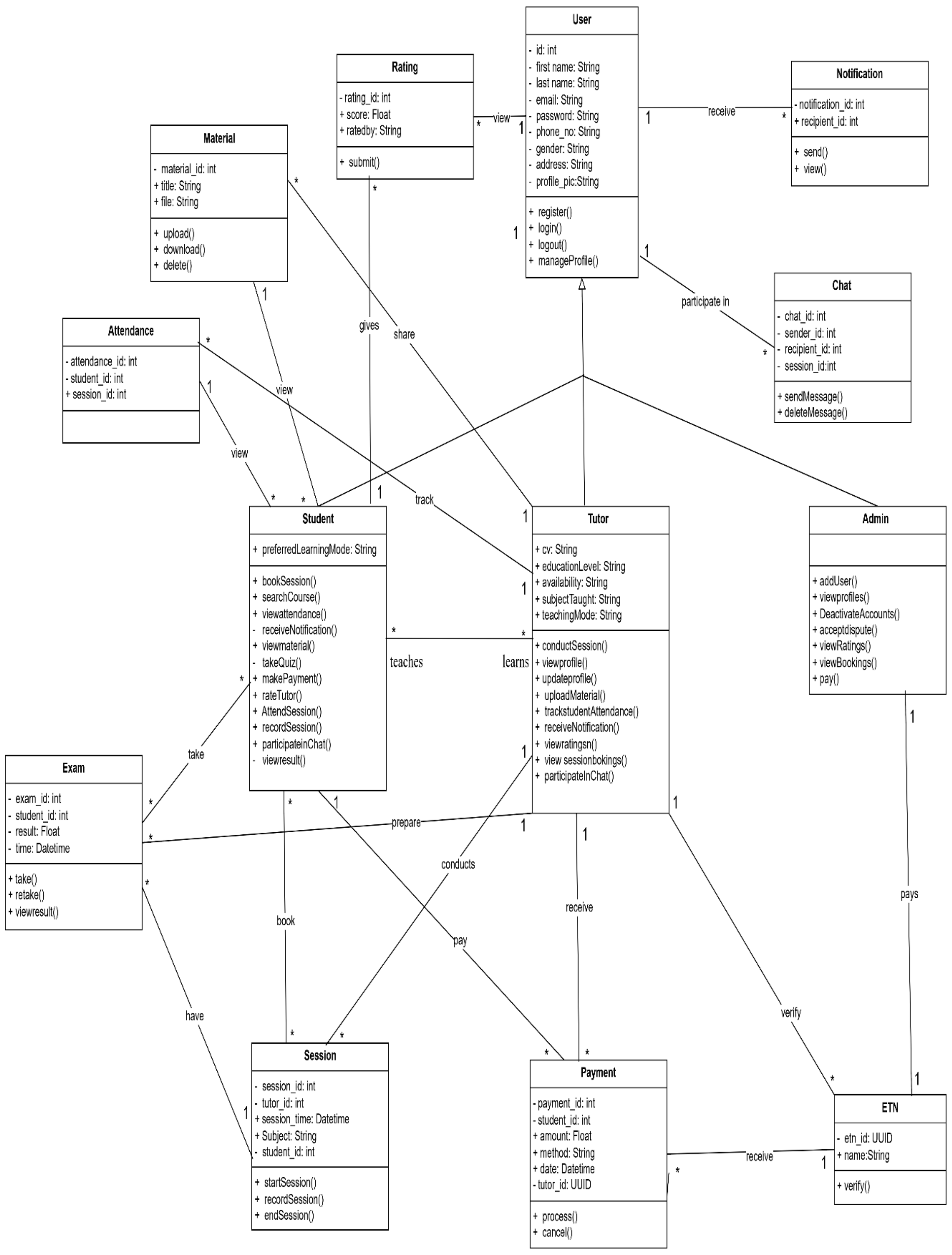


Figure 3.14 Class Diagram

Chapter Four: System Design

4.1 Overview

The system design phase of our tutoring platform plays a crucial role in translating user and functional requirements into actionable plans. This stage establishes how the system's components such as users, features, data flow, and architecture are structured and integrated to support core functions such as tutor-student interaction, session scheduling, learning materials distribution, and online tutoring.
Our goal is to provide an inclusive educational platform that offers personalized, observable tutoring for both online and in-person sessions, from primary school to university level. This section outlines the design considerations and choices made to ensure accessible, secure, and responsive system architecture.

4.2 System Design

System design plans how a system will function, covering architecture, components, and interfaces to meet requirements. It ensures scalability (handling growth), performance (speed/efficiency), maintainability (easy updates), security (data protection), user experience (intuitive design), and compliance (standards/integration). This phase converts analysis into actionable specifications for developers.

Core Architectural Components

Authentication & Authorization

- Secure Login System using passwords
- Email Verification is required before account activation, promoting trust and security.
- Role-Based Redirection: After logging in, users are redirected to role-specific dashboards with their permitted features.

Student Dashboard Features

- **Tutor Search & Booking**: Students browse tutors and reserve slots based on availability.
- **Attend Sessions**: Students can attend tutoring sessions.
- **Exam & Result Tracking**: Students take online assessments and view scores with analysis.
- **Material Access**: use study guides or notes uploaded by tutors.
- **Manage profile:** have access to update the information in their profile.
- **Tutor Ratings**: Rate tutors after a session to improve the learning mechanism.
- **View attendance**

Tutor Dashboard Features

- **Schedule Management**: Tutors define when they are available for sessions.
- **Session Bookings**: View bookings and student information.
- **Live Sessions**: Launch or join live classes through integrated video services.
- **Exam Preparation**: Upload and assign quizzes or evaluations.
- **Material Sharing**: Tutors upload resources for students.
- **Manage profile:** have access to update the information in their profile.
- **Student Monitoring**: View attendance and performance metrics.

Admin Dashboard

- **User Management**: Add and deactivate accounts (students, tutors, ETN officers)
- **Platform Metrics**: View usage statistics, tutor ratings, bookings and session information/ summaries
- Manage payments (pay ETN and view transactions)
- **Report Handling**: Accept or reject student/tutor reports.

ETN Officer Portal

- **Document Verification**: Review and validate tutor qualification documents

Technology Stack

Frontend	React
Backend	Next.js
Database	MongoDB
Video Conferencing	Jitsi Meet integration
Payment Gateway	Chapa
Development Tools	Visual Studio Code, GitHub

4.3. Architecture of the System

Our chosen architecture is based on a client-server model, designed specifically for a modern tutoring platform that facilitates both online and in-person learning for students across different education levels. The architecture supports dynamic interaction between students, tutors, admins, and ETN officers while ensuring scalability, security, and smooth performance. Key components include:

Client Interface:
This is the user-facing component developed using React, optimized for web access. It provides a friendly and responsive interface for students to search tutors, book sessions, attend classes, and view materials. Tutors can manage schedules, share learning resources, and monitor attendance. Admins and ETN officers access dashboards with administrative tools and validation features.

Application Server (Backend):

The backend is built using Next.js, which handles the core backend logic. It manages like user authentication, sessions, exam processing, payment, chat, video conferences and notifications. The backend is also responsible for routing role-based functionality and ensuring secure, modular communication between the client and the database.

Database Server:

The system uses MongoDB, a flexible NoSQL database, for storing all essential data including user profiles, session records, learning materials, exam results, payments, and feedback. MongoDB's document-oriented structure is well-suited for handling dynamic, user-generated content and scalable workloads across multiple roles and interactions.

Third-Party Services Integration:

The platform integrates with third-party services to enhance functionality. Notably, Chapa is used as the payment gateway to securely handle tutoring session payments. For communication and virtual sessions, APIs such as Jitsi are incorporated to enable video conferencing, live classes, and chat features.

Security Layer:

Robust security mechanisms are embedded throughout the architecture. These include JWT-based authentication, role-based access control, HTTPS data transmission, and MongoDB access restrictions. Additional protections like rate-limiting, input validation and server-side encryption help prevent unauthorized access and ensure the integrity of sensitive user data and financial transactions.

4.4 Access Control and Database Design

4.4.1 User Roles & Access Control

The platform defines four user roles, each with unique permissions and workflows:

- **Student**
  - Register/login
  - Search available tutors
  - Book sessions (online or in-person)
  - Take quizzes and receive results.
  - View shared materials
  - Rate tutors
  - receive session notifications
  - Manage their profile
  - Make payments for tutoring services
- **Tutor**
  - Register/login
  - Set their schedule.
  - Conduct online sessions or register in-person meetings
  - Prepare quizzes
  - View session bookings
  - Upload and share learning materials.
  - Track student attendance.
  - View feedback/ratings from students.
  - Chat with students, notify students about session updates and receive session notifications
  - Manage their profile
- **Admin**
  - Manage users (add, view and deactivate accounts)
  - View ratings
  - Pay for ETN
  - view bookings
  - Accept and review session reports.
- **ETN Officer**
  - Validate tutor credentials/documents.

4.4.2 Data base diagram

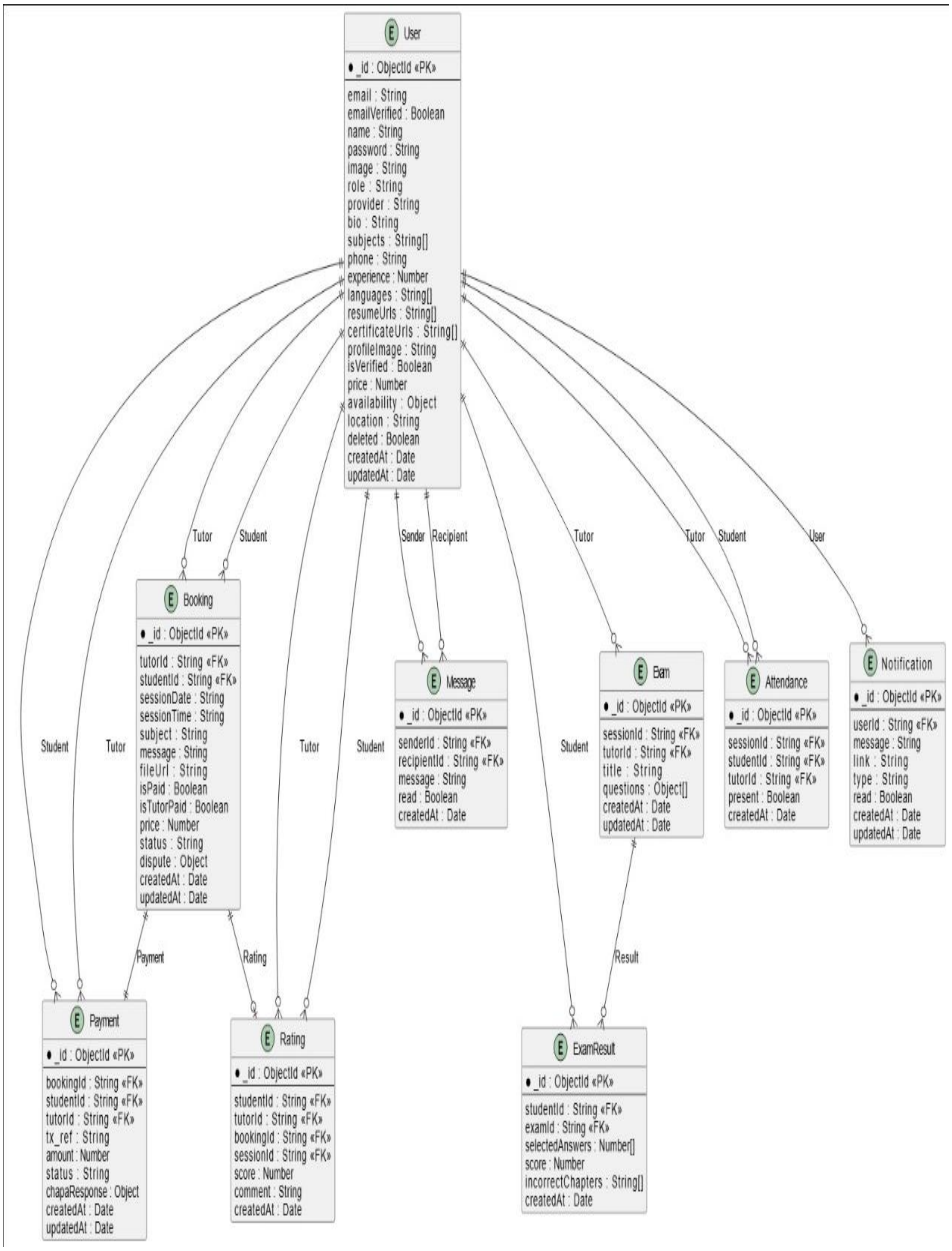
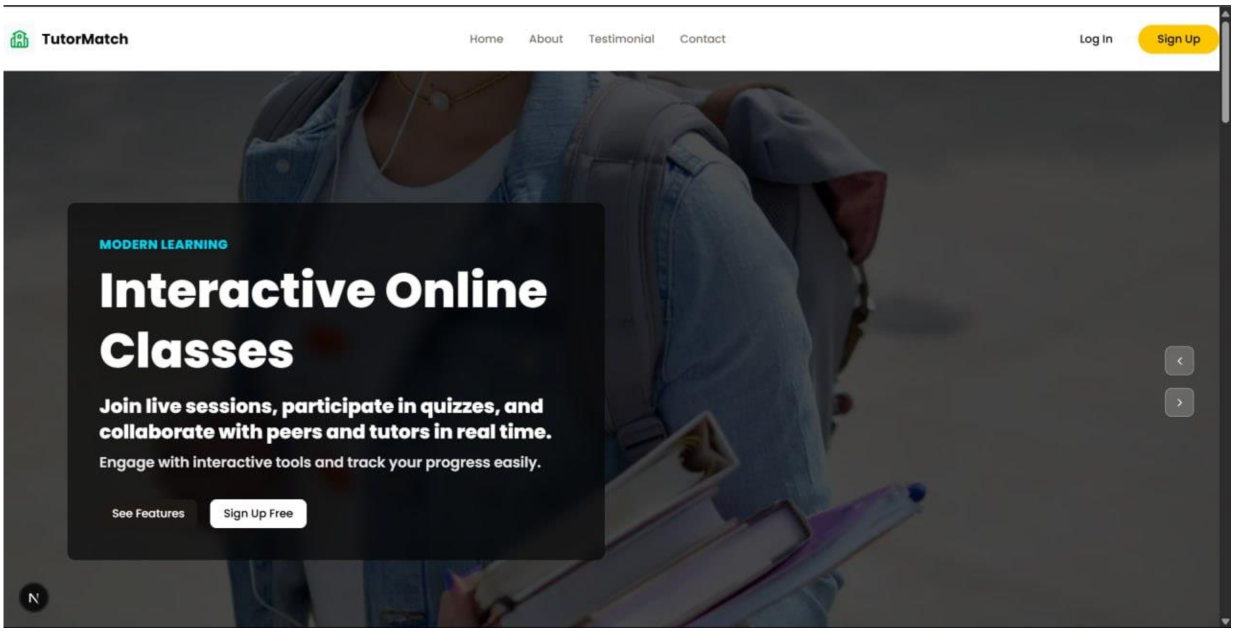


Figure 4.1 Database Design

4.5 User Interface Design

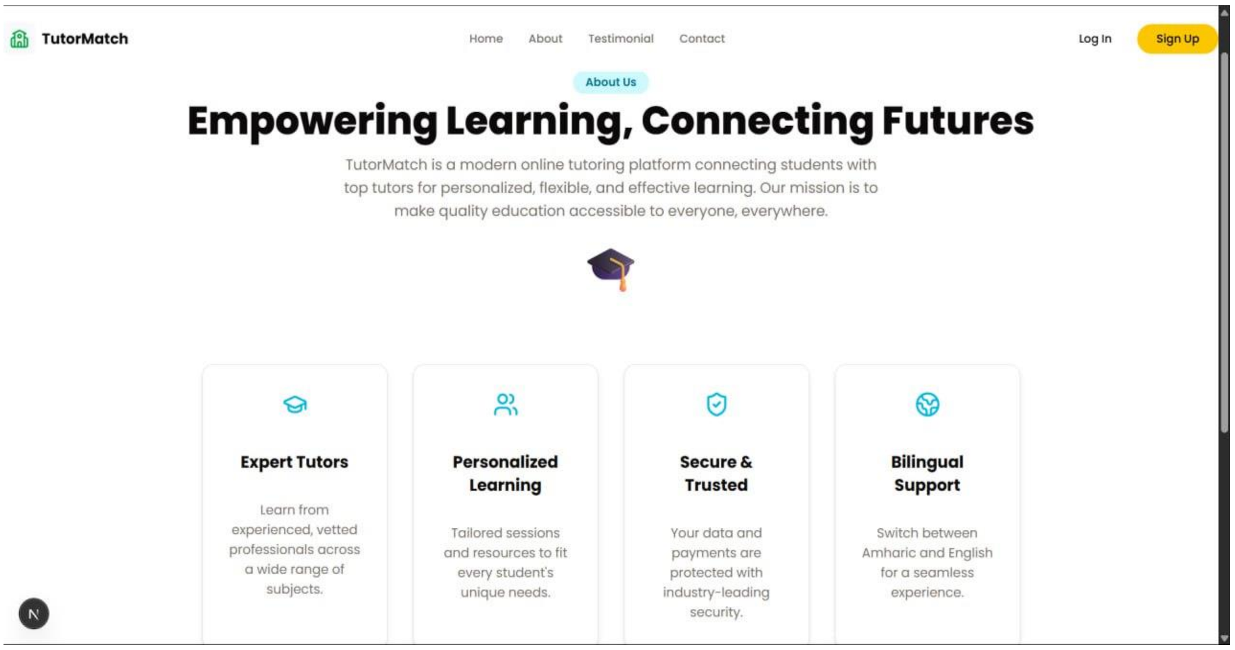


MODERN LEARNING

Interactive Online Classes

Join live sessions, participate in quizzes, and collaborate with peers and tutors in real time.

Engage with interactive tools and track your progress easily.



Empowering Learning, Connecting Futures

TutorMatch is a modern online tutoring platform connecting students with top tutors for personalized, flexible, and effective learning. Our mission is to make quality education accessible to everyone, everywhere.

Expert Tutors

Learn from experienced, vetted professionals across a wide range of subjects.

Personalized Learning

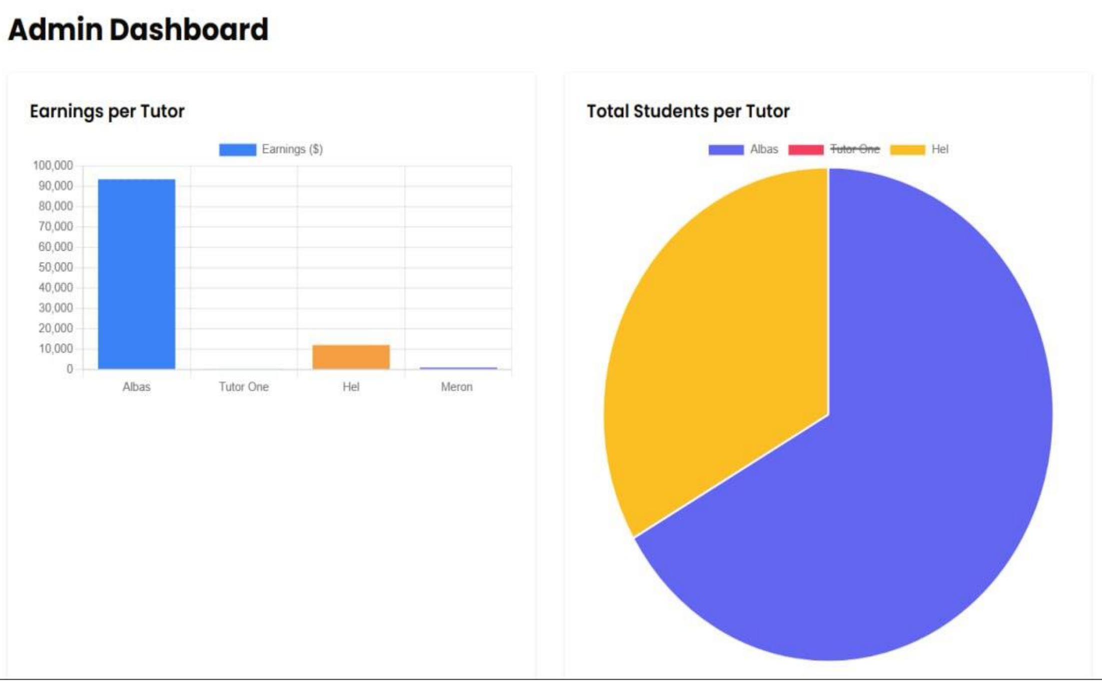
Tailored sessions and resources to fit every student's unique needs.

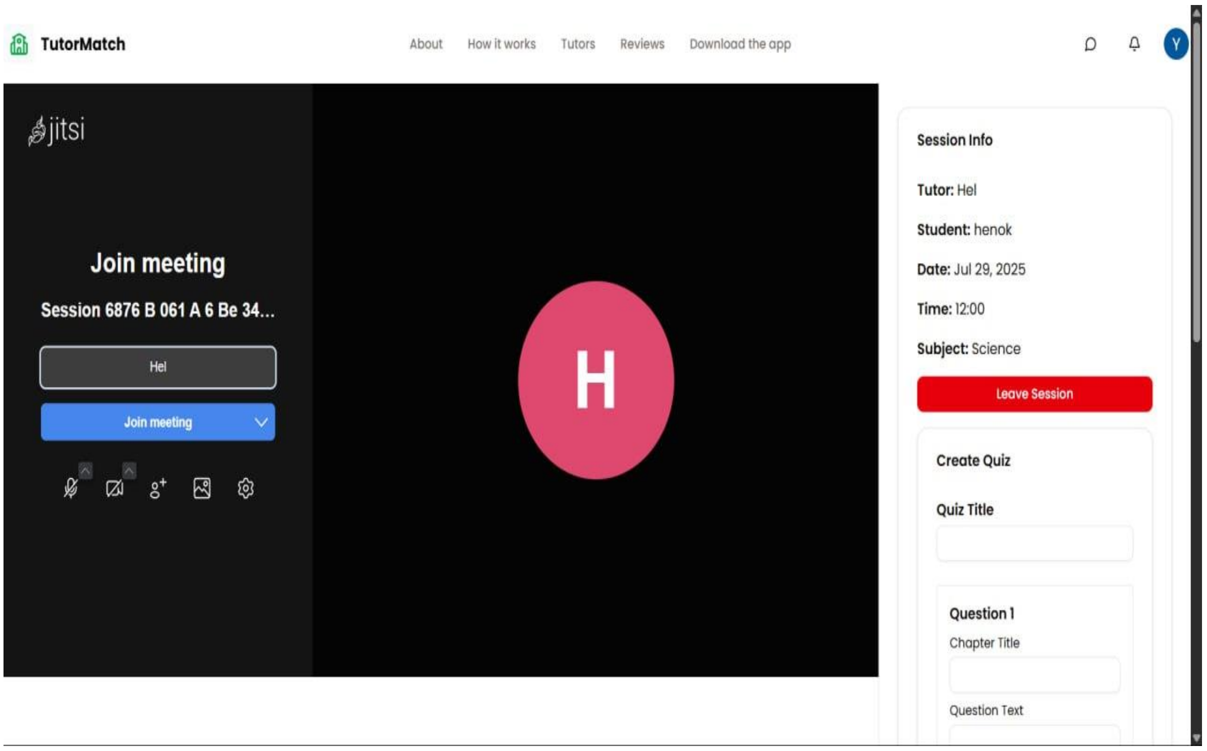
Secure & Trusted

Your data and payments are protected with industry-leading security.

Bilingual Support

Switch between Amharic and English for a seamless experience.





Session Info

Tutor: Hel

Student: henok

Date: Jul 29, 2025

Time: 12:00

Subject: Science

Create Quiz

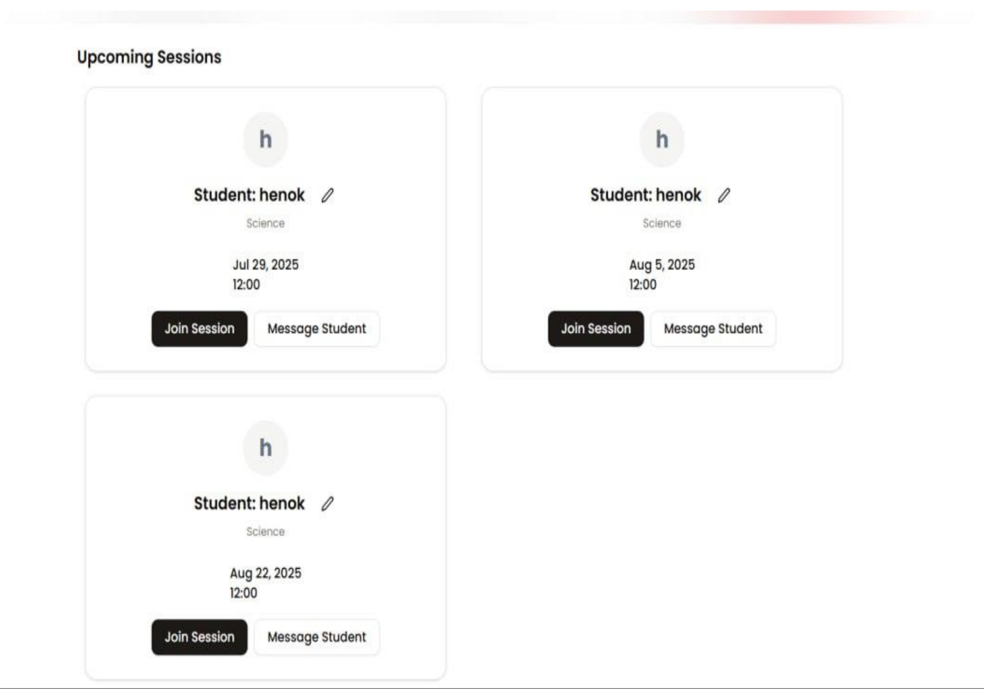
Quiz Title

Question 1

Chapter Title

Question Text

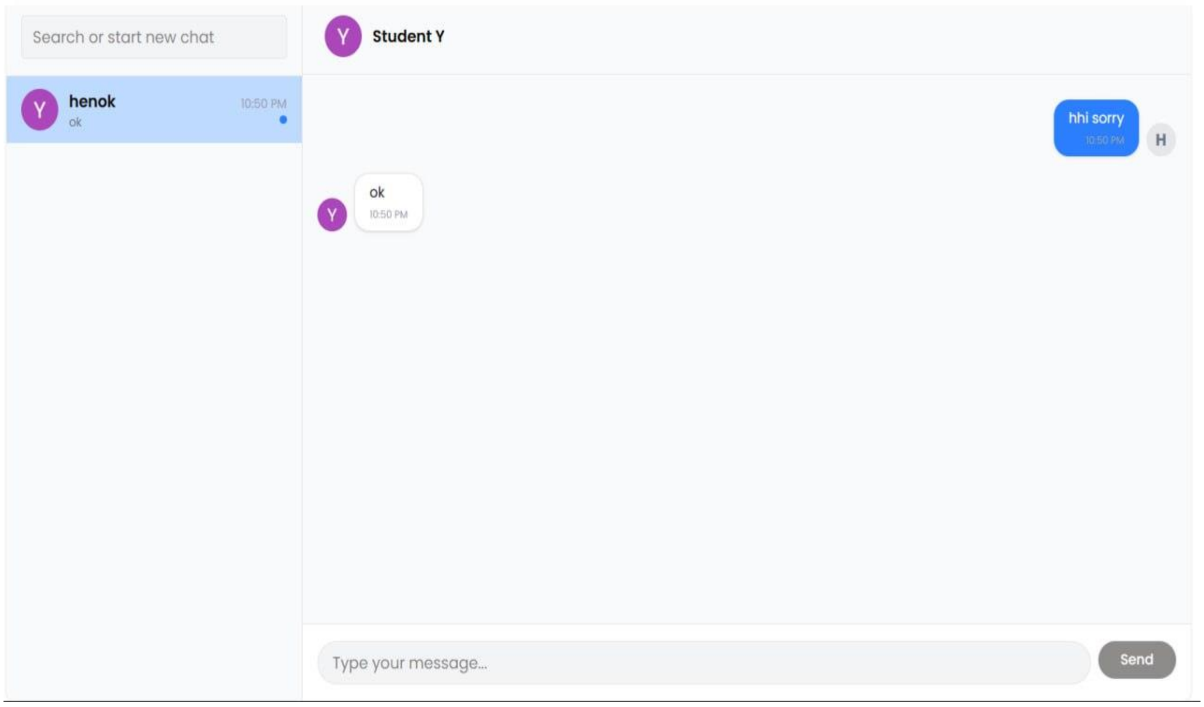
Session 6876 B 061 A 6 Be 34...


Upcoming Sessions

h
Student: henok
Science
Jul 29, 2025
12:00

h
Student: henok
Science
Aug 5, 2025
12:00

h
Student: henok
Science
Aug 22, 2025
12:00



henok 10:50 PM
ok

Student Y

hhi sorry 10:50 PM

ok 10:50 PM

Chapter Five: Implementation and Testing

5.1 Introduction

For the Tutoring Platform project, this Chapter is dedicated to the practical realization of the system's design and ensuring its functionality. This chapter is crucial for describing the behavior of the implemented system and justifying how it meets the initial design specifications. While it typically doesn't include all code, it presents sample code that highlights particular approaches or represents important components. This part of the document also emphasizes providing a walk-through of the system to explain its behavior and its relation to the design, along with producing evidence and justification from various sources.

5.2 Implementation

The implementation of the tutoring platform was conducted in several well-organized stages to ensure that the final system aligns with the design specifications and meets user expectations. The following outlines each key implementation phase:

1. System Analysis and Design:

The project began with analyzing the specific functional and non-functional requirements of a modern online tutoring platform. The goal was to enable secure, scalable interactions between students, tutors, and administrators. System diagrams including Use Case, Entity Relationship Diagrams (ERD), and user flowcharts were developed to visualize the journey and interaction across roles. This architectural planning shaped our schema design, page routing, and API structure.

2. Backend and Database Development:

The backend was developed using Next.js App Router with API routes, leveraging its server-side capabilities for robust system operations. MongoDB served as the primary database, with Mongoose providing schema modeling for data collections including Users with role differentiation. This combination delivered a scalable and maintainable foundation for the platform's data management needs.

3. Frontend Development

The frontend was developed using Next.js (React), Tailwind CSS, and Shadcn UI to deliver a responsive and intuitive user experience across all devices. This modern tech stack enabled the creation of modular, role-specific interfaces that maintained consistent performance on both desktop and mobile platforms. The component-based architecture with Tailwind's utility-first CSS and Shadcn's accessible UI patterns ensured efficient development while meeting high standards of usability and design consistency.

4. Third-Party Integrations

To extend functionality, several third-party services were integrated:

- **Chapa API**: For local Ethiopian payments, integrated at the point of session booking.
- **Jitsi**: For video conferencing inside /session/:id/join with dynamic room generation and role-based panels.
- **Better-auth**: For secure and minimal email/password and social auth.
- **Resend Email API**: To send account activation emails and session notifications.

5. Chat Feature Implementation

The platform implements a basic CRUD (Create, Read, Update, and Delete) chat system where messages are stored in a dedicated messages collection within the database, accessible either through a dedicated chat interface or via "Message Tutor/Student" buttons embedded in user dashboards. Conversations are logically grouped by senderId and recipientId pairs to maintain message threads, with the system employing a straightforward fetch-and-render approach rather than real-time socket connections for message delivery, ensuring simplicity and reliability in communication between users while minimizing technical complexity. This design choice prioritizes stability and ease of maintenance while still providing essential messaging functionality for tutor-student interactions.

6. Report System & Payment Logic

The platform implements a secure payment flow where students pay when booking a session, ensuring smooth payment mechanism between students and tutors. To maintain quality and accountability, a reporting mechanism allows students to notify administrators if tutors fail to join sessions or provide unsatisfactory instruction. Administrators can proactively remove a tutor profile if the tutor keeps getting bad reviews and 1 or more students report an issue. This dual-layer oversight combining student rating and student reports ensures fair transactions and maintains educational standards across all tutoring sessions.

7. Notification System

Every major action sends notifications including new bookings, Tutor approval, student enrollment, payment and changes on schedule. They are stored in the notifications collection and accessed via a bell icon in the navigation bar.

8. Testing and Debugging

We implemented defensive programming strategies, with backend validation on every endpoint and user role. Testing was performed at three levels: Unit testing for key utilities, Manual integration testing across flows (booking, payment, session join) and User acceptance testing (UAT) with real tutors and students.

9. Development Environment

The development environment was designed for collaboration, productivity, and version control. Key components included:

- **OS**: Windows 10/11
- **Editor**: VS Code with extensions for Tailwind, React, MongoDB
- **Git/GitHub**: Branch-based feature development, commit conventions followed
- **Package Manager**: npm

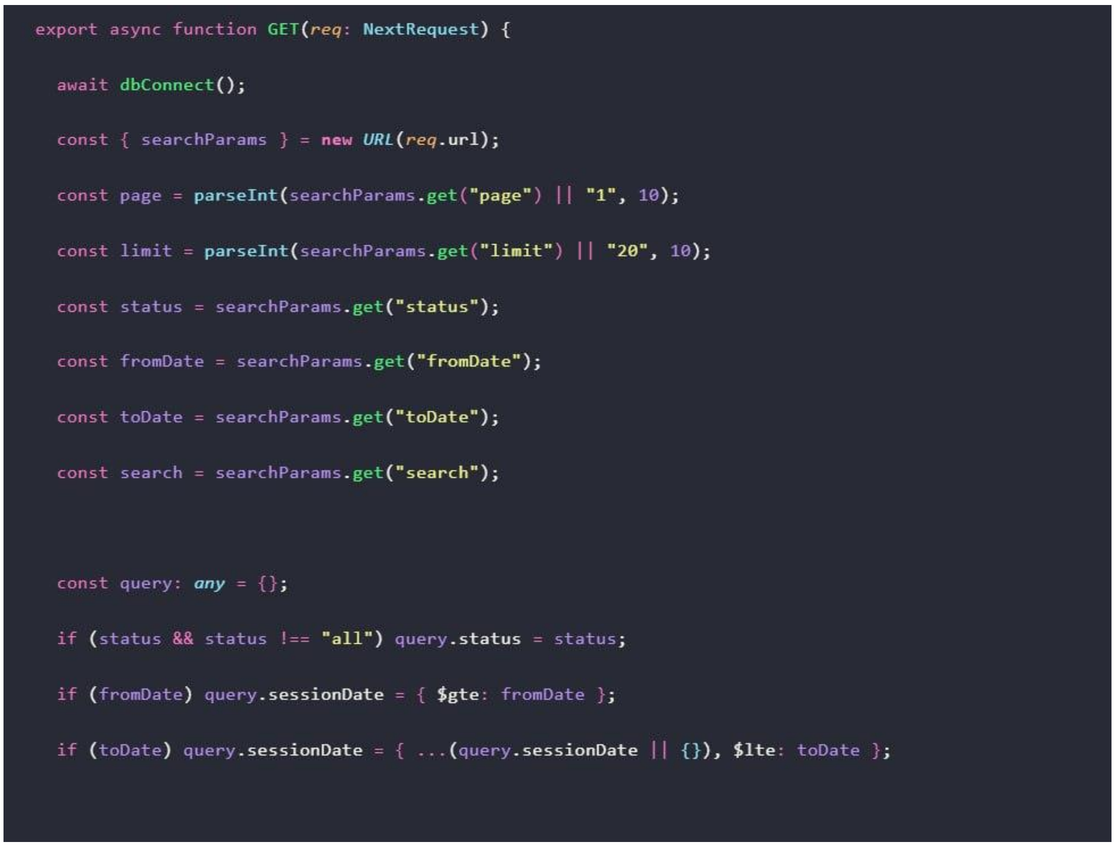
10. Coding Standards and Practices

- Folder structure follows domain-based modularity
- Auth protected routes via middleware and better auth session
- ESLint + Prettier used for consistent formatting
- API error handling with custom responses

11. Sample Codes

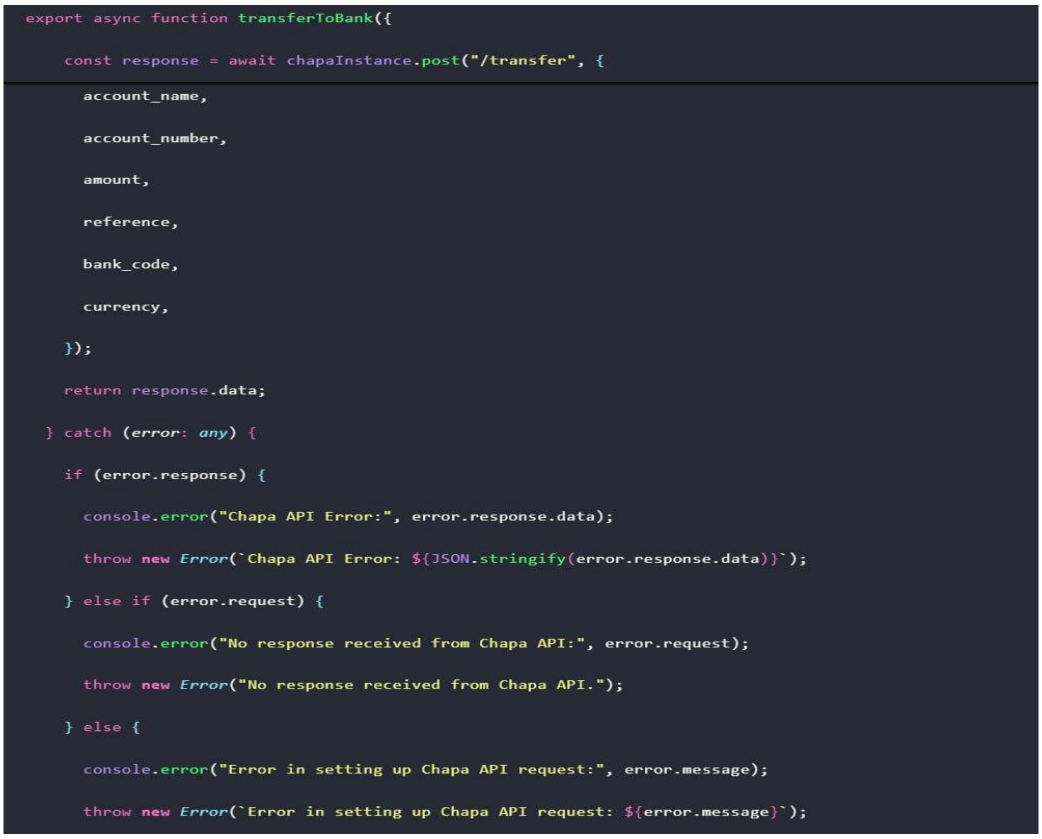
The following provides examples of sample code from different parts of the system:

Route.ts



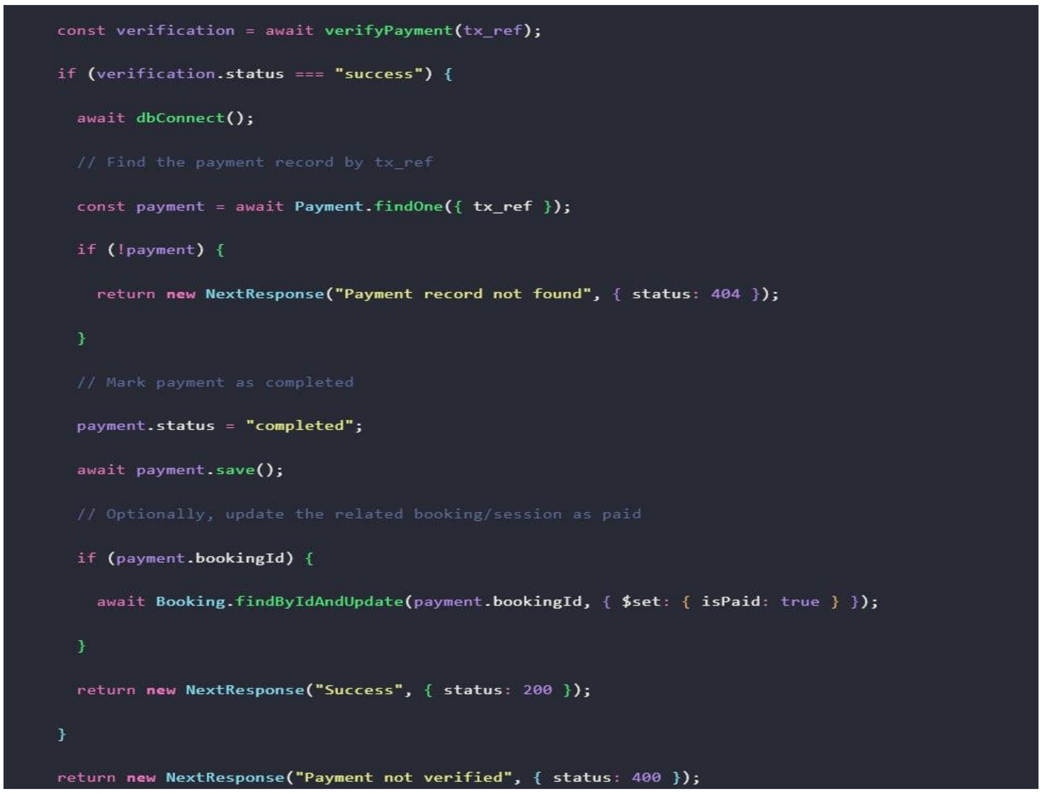
```
export async function GET(req: NextRequest) {

  await dbConnect();

  const { searchParams } = new URL(req.url);

  const page = parseInt(searchParams.get("page") || "1", 10);

  const limit = parseInt(searchParams.get("limit") || "20", 10);

  const status = searchParams.get("status");

  const fromDate = searchParams.get("fromDate");

  const toDate = searchParams.get("toDate");

  const search = searchParams.get("search");


  const query: any = {};

  if (status && status !== "all") query.status = status;

  if (fromDate) query.sessionDate = { $gte: fromDate };

  if (toDate) query.sessionDate = { ...(query.sessionDate || {}), $lte: toDate };
```

Chapa.ts



```
export async function transferToBank({

    const response = await chapaInstance.post("/transfer", {
      account_name,
      account_number,
      amount,
      reference,
      bank_code,
      currency,
    });
    return response.data;
  } catch (error: any) {
    if (error.response) {
      console.error("Chapa API Error:", error.response.data);
      throw new Error(`Chapa API Error: ${JSON.stringify(error.response.data)}`);
    } else if (error.request) {
      console.error("No response received from Chapa API:", error.request);
      throw new Error("No response received from Chapa API.");
    } else {
      console.error("Error in setting up Chapa API request:", error.message);
      throw new Error(`Error in setting up Chapa API request: ${error.message}`);
```

Webhook/route.ts



```
    const verification = await verifyPayment(tx_ref);
    if (verification.status === "success") {
      await dbConnect();
      // Find the payment record by tx_ref
      const payment = await Payment.findOne({ tx_ref });
      if (!payment) {
        return new NextResponse("Payment record not found", { status: 404 });
      }
      // Mark payment as completed
      payment.status = "completed";
      await payment.save();
      // Optionally, update the related booking/session as paid
      if (payment.bookingId) {
        await Booking.findByIdAndUpdate(payment.bookingId, { $set: { isPaid: true } });
      }
      return new NextResponse("Success", { status: 200 });
    }
    return new NextResponse("Payment not verified", { status: 400 });
```

5.3 Testing

Testing is a critical phase in software development that ensures the software meets the specified requirements and is free of defects. The following types of testing were conducted:

Test Case 1: Student Browses Tutors and Books a Session

This test case verifies the end-to-end process of a student finding and booking a session with a tutor, incorporating elements from the "Browse Available Tutors" and "Book Tutoring Sessions" functional requirements.

Test Step	**Test Data**	**Expected Result**	**Pass Criteria**	**Fail Criteria**
1.Student login	Valid Student Credentials (student@example.com, Pass123!)	Student is redirected to their dashboard/home page.	User is successfully logged in and redirected to the expected landing page.	Login fails, or an error message is displayed, or the user remains on the login page.
2. Navigate to "Tutors." Section	N/A	The "Tutors" page is displayed with search filters and a list of tutors.	"Tutors" page loads successfully, showing filters (subject, price, criteria) and initial tutor listings.	Page fails to load, or essential filters/tutor listings are missing.
3. Apply filters.	Subject: Mathematics, Availability: Tomorrow	The system displays a filtered list of tutors matching the criteria.	Filtered list of tutors is displayed accurately, showing only those matching the selected criteria.	Filters are not functional, or results do not reflect the chosen criteria, or loading time.

4. Select a tutor and view profile.	Click on Tutor X's profile.	Tutor X's detailed profile page is displayed with availability.	Tutor's profile page loads successfully, showing qualifications, experience, teaching style, and available time slots.	Profile page fails to load, or crucial tutor details/availability are missing.
5. Select a preferred timeslot.	Tomorrow, 10:00 AM	The selected timeslot is highlighted, and a "Book Session" button appears.	Timeslot selection is functional, and the system prepares for booking.	Timeslot cannot be selected, or booking option is unavailable.
6. Click "Book Session."	N/A	A payment form/confirmation page is presented.	Payment form loads successfully.	Payment form fails to load, or an error occurs during the booking initiation.
7. Complete payment.	Valid Payment Details (e.g., Chapa)	Payment is processed successfully, and a confirmation message is displayed.	Payment process completes and a confirmation message is shown. Notification sent after booking.	Payment fails, an error message is displayed (e.g., "Insufficient funds", "Invalid payment information"), or confirmation is not displayed.
8. Verify session confirmation.	Check student dashboard/ notifications.	The booked session appears in the student's upcoming sessions.	Session is successfully booked, scheduled, and appears in the student's booked session. Notification of booking receipt received.	Booked session does not appear, or an incorrect session is listed.

Table 5.1 Tutor Conducts Online Session and Student Joins

Test Case 2: Tutor Conducts Online Session and Student Joins

Test Step	**Test Data**	**Expected Result**	**Pass Criteria**	**Fail Criteria**
1. Tutor logs in and initiates session.	Valid Tutor Credentials (tutor@example.com, Pass123!)	Tutor's dashboard loads, and an option to start the session is available.	Tutor is logged in and can access the session management interface. The "Join Session" button/link is present and functional.	Login fails or session initiation option is missing/non-functional.
2. System generates secure link.	N/A	A secure video conferencing link and session ID are generated.	System generates a unique, secure link and session ID.	Link generation fails or link is not secure.
3. Student logs in and joins session.	Valid Student Credentials (student@example.com, Pass123!), Session Link	Student is directed to the online session interface.	Student is logged in and can access the session link. Video/audio connection is established for both tutor and student.	Student cannot join, connection fails to establish, or audio/video issues prevent effective communication.
4. Tutor shares screen/material.	Sample Presentation/ Document	The shared content is displayed to the student.	Tutor can successfully initiate screen sharing or upload/share learning materials, and the student can view them clearly.	Shared content is not visible to the student, or sharing functionality is broken.

5. Student interacts and asks questions.	Student uses chat or microphone.	Student's messages/audio are received by the tutor.	Student can use the integrated chat system to communicate with the tutor, and their audio input is clear.	Chat messages are not sent/received, or audio communication is not functional.
6. System tracks session duration/attendan ce.	N/A	Session duration and student attendance are recorded by the system.	The system accurately tracks the duration of the session and records student attendance.	Session duration or attendance is not recorded, or records are inaccurate.
7. Session ends, tutor stores attendance.	Tutor clicks "Leave Session."	The session concludes, and attendance records are saved.	Tutor can end the session, and the system saves attendance records immediately after the session ends.	Session cannot be ended, or attendance records are not saved or are corrupted.

Table 5.2 Tutor Conducts Online Session and Student Joins

Test Case 5.3: Student Rates and Reviews a Tutor

This test case focuses on the feedback mechanism, based on the "Rate and Review Tutors" functional requirement

Test Step	Test Data	Expected Result	Pass Criteria	Fail Criteria
1. Student logs in.	Valid Student Credentials (student@exa mple.com, Pass123!)	Student is redirected to their dashboard/homep age.	User is successfully logged in and redirected to the expected landing page.	Login fails, or an error message is displayed, or the user remains on the login page.

2. Navigate to "Booked Session"	N/A	A list of booked sessions is displayed.	The page loads successfully, listing booked sessions and after joining a session it shows a clear option to rate tutors	Page fails to load, or booked sessions are not listed, or rating option is missing.
3. Join a session	Select the amount of stars you rate the tutor and click "Rate session" button next to Session with Tutor Y.	The rating interface (e.g., star rating, text box for review) for Tutor Y is displayed.	The rating interface loads, allowing the student to select a rating score and enter a review.	Rating interface fails to load or is not interactive.
4. Submit rating and review.	Rating: 5 Stars, Review: "Excellent session, very helpful!"	The rating and review are successfully submitted and stored. A confirmation message is displayed.	The rating is saved and reflected on the tutors' profile promptly. A confirmation of successful submission is displayed.	Rating/review submission fails, no confirmation is displayed, or the rating is not updated on the tutor's profile.
5. Verify updated tutor rating.	Access Tutor Y's public profile.	Tutor Y's overall rating reflects the newly submitted rating, and the review is visible	The tutor's profile displays the updated average rating, and the submitted review is visible to the student and potentially other users, as per system design.	Tutor's rating does not update, or the review is not visible, or an incorrect rating is displayed.

Table 5.3 Student Rates and Reviews a Tutor

Test Case 4: User Profile Update

This test case verifies that users (e.g., Students, Tutors, Admin, and ETN) can successfully update their personal or professional profiles on the platform, reflecting the non-functional requirement of usability by ensuring a "user-friendly, intuitive, and accessible" interface for such actions.

Test Step	Test Data	Expected Result	Pass Criteria	Fail Criteria
1. Log in to the system with an existing user account.	Username: "johndoe@example.com" Password: "ValidPassword123"	User is successfully logged in and redirected to their dashboard or homepage.	User is logged in and can access their role-specific functionalities.	Login fails due to incorrect credentials or technical issues.
2. Navigate to the "Profile" then "Edit Profile" section.	N/A	The system displays the user's current profile information in editable fields.	Edit profile page loads successfully with current details pre-populated.	Page fails to load, or profile information is missing/incorrect.
3. Update desired profile fields.	**For Student:** Learning Mode: "Online" **For Tutor:** tutoring type: "online" Updated **For Admin:** New Name: "Dr. kebede"	The system accepts the updated information in the respective fields.	Fields allow new input without errors.	Fields are not editable, or input is rejected for valid data.

4. Submit the updated profile information.	N/A	The system validates the input, saves the updates, and displays a success message. The changes are immediately reflected in the user's profile and visible where applicable.	**Profile updates are saved successfully.** A confirmation message is displayed. Profile updates reflect (**For Admin:** New Name: "Dr. kebede")	Profile update fails due to invalid data or technical issues. No confirmation message is displayed, or changes are not saved/reflected.
5. Verify the updated information.	N/A	The updated information is visible on the user's profile page and anywhere else it is displayed within the system.	The new profile details are correctly displayed and accessible across the system. Validation errors (if any) are displayed clearly.	Old information persists, or incorrect data is displayed after the update. Error messages are unclear or unhelpful.

Table 5.4 user updates their profile test case

Chapter Six: Conclusion and Future Work

6.1. Conclusion

The proposed tutoring platform presents an innovative digital solution to Ethiopia's education challenges by creating a centralized system that effectively connects students with qualified tutors. Addressing critical gaps in the current fragmented model, the platform introduces key features including flexible learning options (both online and in-person sessions), secure payment integration, communication tools, and comprehensive performance tracking. Together, these elements enhance accessibility, accountability, and learning outcomes across all academic levels, from primary and secondary education to university courses and language courses.

The platform's systematic methodology elevates traditional tutoring practices by incorporating verified tutor profiles, comprehensive assessment tools, and a transparent evaluation system. Its additional features, including attendance monitoring, resource sharing capabilities, and progress tracking, significantly bolster its potential to revolutionize academic support services throughout Ethiopia's diverse communities, encompassing both urban centers and rural areas. By digitizing and optimizing the connection between educators and learners, this innovative solution establishes a crucial framework for democratizing access to quality educational support across the nation.

6.2. Future Work

- **AI-Powered Tutor-Student Matching**: Implement machine learning algorithms to suggest the best tutors based on student learning styles, past performance, and subject needs.
- **Mobile Application Development**: Expand accessibility by launching Android and iOS apps, particularly for users in rural areas with limited desktop access.
- **Gamification & Interactive Learning**: Introduce badges, leaderboards, and interactive exercises to increase student engagement and motivation.
- **Expanded Language Support**: Add more local languages (e.g., Oromo, Tigrinya) to improve inclusivity beyond English and Amharic.
- **Offline Mode for Low-Internet Areas**: Allow downloadable learning materials and pre-recorded sessions for students with unreliable internet access.
- **Advanced Analytics & Reporting**: Provide data-driven insights for tutors, students, and administrators to track long-term progress.
- **Scholarship & Financial Aid Features**: Introduce discounts or subsidized tutoring for underprivileged students to promote equal access.
- **Virtual Classroom Enhancements**: Incorporate whiteboard tools, breakout rooms, and AI-based live transcription for better online learning.
- **Scalability to Other African Markets**: Adapt the platform for neighboring countries with similar educational challenges (e.g., Kenya, Uganda, Rwanda).

By continuously evolving with technological advancements and user feedback, this tutoring platform has the potential to revolutionize education accessibility not only in Ethiopia but across Africa, fostering a smarter, more connected, and equitable learning future.

References

- Sommerville, I. (2016). Software engineering (10th ed.). Pearson.
- Pressman, R. S., & Maxim, B. R. (2020). Software engineering: A practitioner's approach (9th ed.). McGraw-Hill.
- Alshammari, M., Alwan, A. A., & Qureshi, M. R. J. (2020). Secure user authentication in web applications: A comparative study. Journal of Information Security and Cybercrimes Research
- MongoDB,Inc.(2023). MongoDB documentation. https://www.mongodb.com/docs/Next.js.(2023).
  Next.js official documentation: https://nextjs.org/docs
- Kebede, A., & Wondwossen, H. (2023). Digital tutoring platforms in developing economies: Challenges and opportunities. Proceedings of the 5th International Conference on Education Technology (pp. 112–125). IEEE.
- Ministry of Education, Ethiopia. (2022). National education sector report. http://www.moe.gov.et/reports
- Chapa. (2023). Chapa payment API integration guide. https://developer.chapa.co/docs
- World Wide Web Consortium (W3C). (2021). Web Content Accessibility Guidelines (WCAG) 2.1. https://www.w3.org/TR/WCAG21/