SAD

Chapter one

Case Study: Online Tutoring System

- **Project Title:**
  Online Tutoring System (OTS)

Objective:

1.1 Problem Statement

- ✓ Adults have no time to go institutions and they prefer to study online anywhere and anytime
- ✓ Its take cost people who is far away from us such as those in other country.
- ✓ Offline classes are expensive than online classes.

1.2 Proposed Solution

- ➢ Available anywhere and anytime
- ➢ Online classes are cheaper than offline classes
- ➢ Online tutoring offers personalization.
- ➢ Self-discipline and responsibility
- ➢ Teachers have ability to share their knowledge anywhere

1.3 Scope of the project

- ✓ The scope of this project is limited to building a web based system for online tutoring education where anyone, anywhere can transform their lives.
- ✓ A platform for students to find and book sessions with qualified tutors.
- ✓ manage courses, schedules, and payments

1.4 Project Goals:

- ✓ Develop an efficient, scalable platform to facilitate tutoring sessions.
- ✓ Deliver the project within the defined timeline and budget.

Key Activities

1. **Feasibility Study:**

    - ✓ **Technical Feasibility:**
      Assess the use of modern web and desktop to ensure platform accessibility and scalability. The system will use open-source technologies like Html,css and Javascript for the front end, Mysql for the back end, PHP for server side and cloud hosting solutions for deployment.
    - ✓ **Economic Feasibility:**
      Implement the system with open cost-efficient resources, leveraging software, hardware and open-source tools to minimize expenses.
    - ✓ **Legal Feasibility:**
      Ensure compliance with data protection regulations (e.g., GDPR) to safeguard user information.
    - ✓ **Operational Feasibility:**
      Confirm that tutors and students can efficiently use the platform with minimal onboarding and support.
    - ✓ **Schedule Feasibility:**
      the project is covered within one week.

2. **Identify Project Stakeholders:**

    - ✓ Students: Primary users who will schedule and attend sessions.
    - ✓ Tutors: Experts who will conduct the tutoring sessions.
    - ✓ Platform Administrators: Responsible for managing users, courses, and system performance.

3. **Develop a Project Plan:**

    - ✓ Define timelines for each project phase, including requirements gathering, design, development, testing, and deployment.
    - ✓ Allocate resources such as development team members, tools, and hardware.

4. **Create Initial Risk Assessments:**

   - ✓ Identify risks such as platform downtime, data breaches, or insufficient user adoption.
   - ✓ Develop mitigation strategies, such as regular system monitoring, secure coding practices, and marketing efforts to attract users.

Chapter two

The goal of this phase is to identify, gather, and analyze the functional and non-functional requirements of the **Online Tutoring System** to ensure it aligns with user needs and business objectives.

Key Activities:

1. **Requirement Gathering:**

To ensure the success of the Online Tutoring System, we need to gather detailed requirements from all stakeholders using interviews, surveys, and workshops. Here's some collected data.

1. Objectives of Requirement Gathering

A. Conducting Interviews

Objective: Gain detailed, qualitative insights from key stakeholders.

Steps to Execute:

1. **Identify Stakeholders:**
   - **Students:** Understand learning needs and preferred platform features.
   - **Tutors:** Gather input on teaching methodologies, scheduling, and payment options.
   - **Administrators:** Learn about system management, reporting, and scalability needs.
2. **Prepare Questions:**

- Categorize into general questions and user-specific queries.
- Focus on pain points, must-have features, and expectations.

3. **Sample Questions:**
   - Students:
     - "What challenges do you face in accessing quality tutoring?"
     - "Would you prefer live classes, recorded sessions, or both?"
   - Tutors:
     - "What tools do you need to deliver effective lessons online?"
     - "How should payments and scheduling be managed on the platform?"
   - Administrators:
     - "What features do you need for user management and reporting?"
     - "What security concerns do you have for user data?"
4. **Documentation:**
   - Record sessions with permission.
   - Summarize findings in an interview report.

By aligning the development of the tutor management system with the specific needs and preferences of students, tutors, and administrators, we aim to create a platform that is user-centric, efficient, and secure. This approach ensures that the system not only addresses current challenges but also adapts to future requirements, providing a sustainable solution for all stakeholders involved

B. Surveys and Questionnaires

Objective: Gather quantitative data from a larger audience.

To inform the development of the tutor management system, I conducted surveys and interviews with key stakeholders, including students, tutors, and administrators. The detailed responses and data collected from these engagements are compiled in the appendix of this book

By integrating these survey-derived insights, we can enhance our tutor management system to better meet the needs of its users.

User Needs and Expectations:

1. **Students:**
    - Simple and intuitive platform to access learning materials and schedule sessions.
    - Progress tracking to measure learning outcomes.
2. **Tutors:**
    - Ability to manage multiple courses and track student performance.
    - Tools for uploading and sharing resources, such as PDFs, videos, and quizzes.
    - Automated scheduling to reduce administrative work.
3. **Administrators:**
    - Comprehensive dashboard to manage users, courses, and financial transactions.
    - Tools for monitoring platform usage and generating reports.
    - Secure data storage and compliance with data protection regulations.

Functional Requirements

1. **User Management:**
    - **Registration and Authentication:**
        - Allow users (students, tutors, administrators) to create accounts and securely log in.
2. **Course and Content Management:**
    - **Course Creation:**
        - Allow tutors to create and manage course materials, including documents, videos, and quizzes.
3. **Scheduling**
    - **Session Booking:**
        - Enable students to view tutor availability and book sessions accordingly.
4. **Payment Processing:**
        - Facilitate secure payments for tutoring sessions through trusted payment gateways.

Non-Functional Requirements

1. **Performance:**
   - Ensure the system can handle concurrent users without performance degradation.
2. **Scalability:**
   - Design the system to accommodate growth in user base and feature expansion.
3. **Security:**
   - **Data Protection:**
     - Implement encryption protocols to protect user data.
   - **Compliance:**
     - Adhere to relevant data protection regulations (e.g., GDPR).
4. **Reliability:**
   - Ensure high system uptime and implement backup solutions.
5. **Maintainability:**
   - Develop the system with clean code practices to facilitate easy maintenance and updates.

System Requirement Specification(SRS)

To ensure a seamless online tutoring experience, tutors should invest in proper equipment and software. Some essential items include:

- A reliable computer or laptop with a stable internet connection.
- A webcam and microphone for clear video and audio communication.
- A drawing tablet or touchscreen device with a stylus for better interaction during lessons.

In terms of software, tutors should familiarize themselves with various tools and platforms that can enhance the online learning experience. Some popular options include:

- Web conferencing platforms (e.g., Zoom, Google Meet, Skype) for facilitating live online sessions.
- Online Tutoring System (OTS) for organizing course Management ,tutor and student management.
- Interactive whiteboard applications for real-time collaboration

In conclusion, understanding the role and importance of online tutoring is crucial for tutors to adapt to this educational format and provide effective learning experiences for their students. Investing in the right equipment and software will improve the lessons' overall quality and contribute to the success of this increasingly popular learning method.

Chapter 3

Design System Architecture

1.1 Hardware Architecture

- **Client Devices**: Students and tutors use smartphones, tablets, laptops, or desktops.
- **Server Requirements**: A cloud-based server to host the platform.
  - **Specifications**:
    - **CPU**: Quad-core or higher.
    - **RAM**: 16 GB minimum.
    - **Storage**: 500 GB SSD or higher.
    - **Network**: 1 Gbps connection for smooth streaming.
- **Scalability**: Use cloud services like AWS, Azure, or Google Cloud for auto-scaling.

1.2 Software Architecture

- **Frontend**:
  - HTML,CSS, BOOTSRAP AND JAVASCRIPT.
- **Backend and Server**:
  - PHP
- **Database**:
  - **Relational**: MySQL
- **DevOps**:
  - Continuous Deployment: GitHub

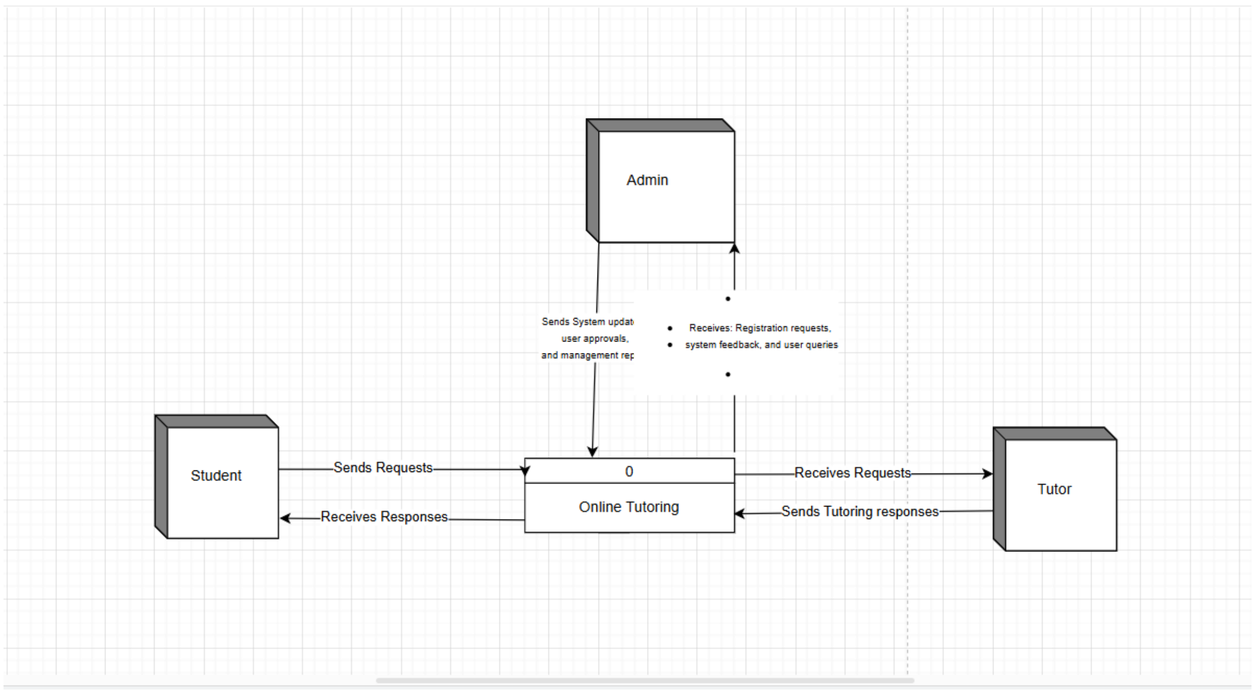
1.3 Network Architecture

- **Communication Protocols**:
  - HTTPS for secure communication.
  - WebSocket for real-time features like chat and live class notifications.

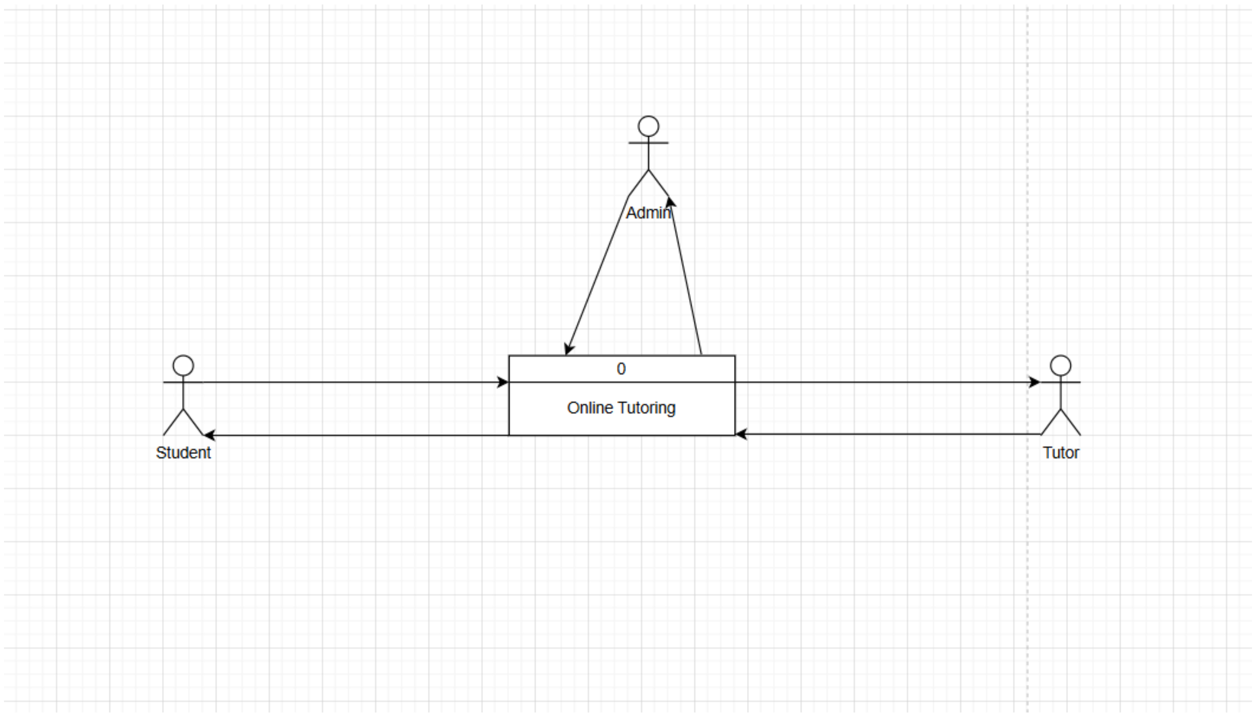
study.eddx

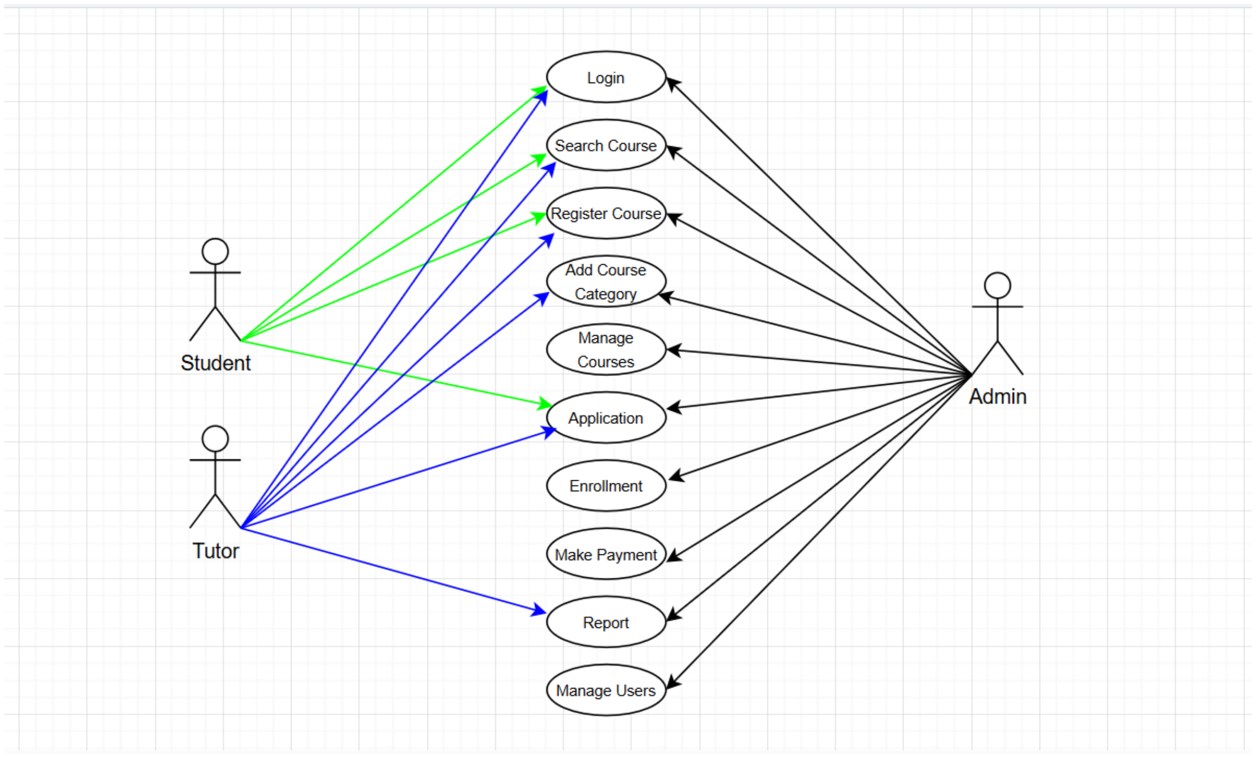
3.3 System Diagram

3.3.1 level 0 Diagram

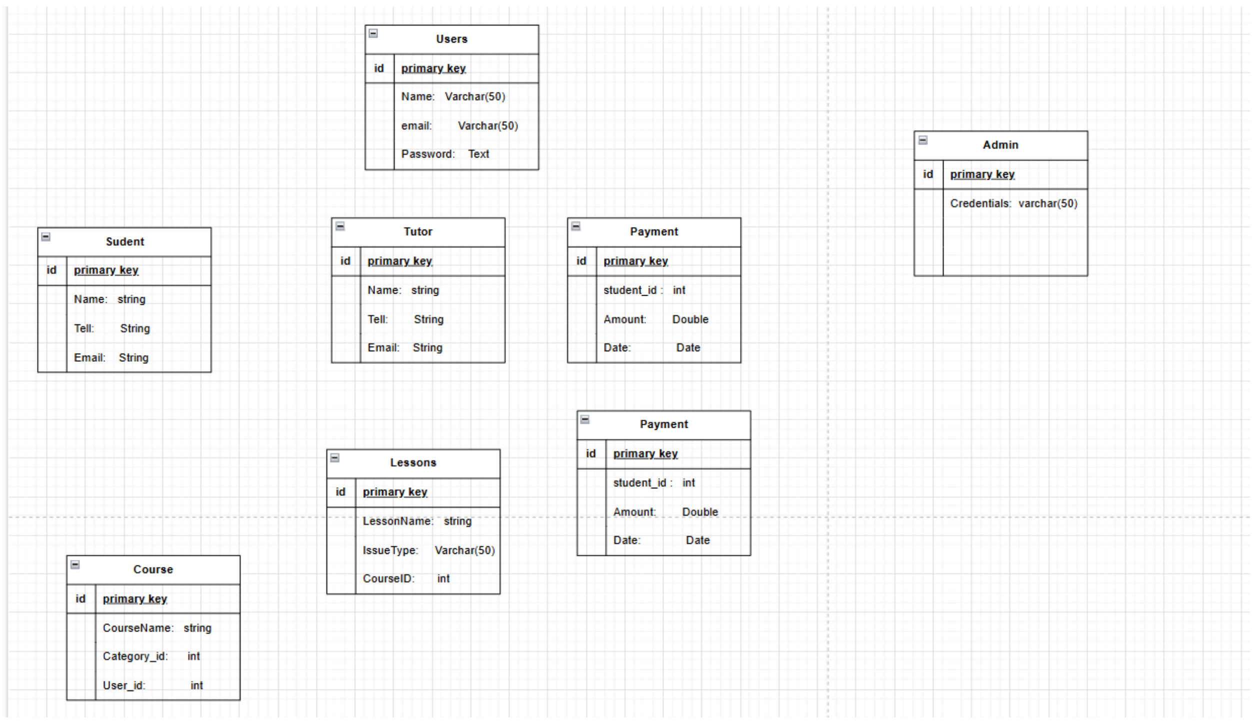


3.3.2 Use Case Diagram

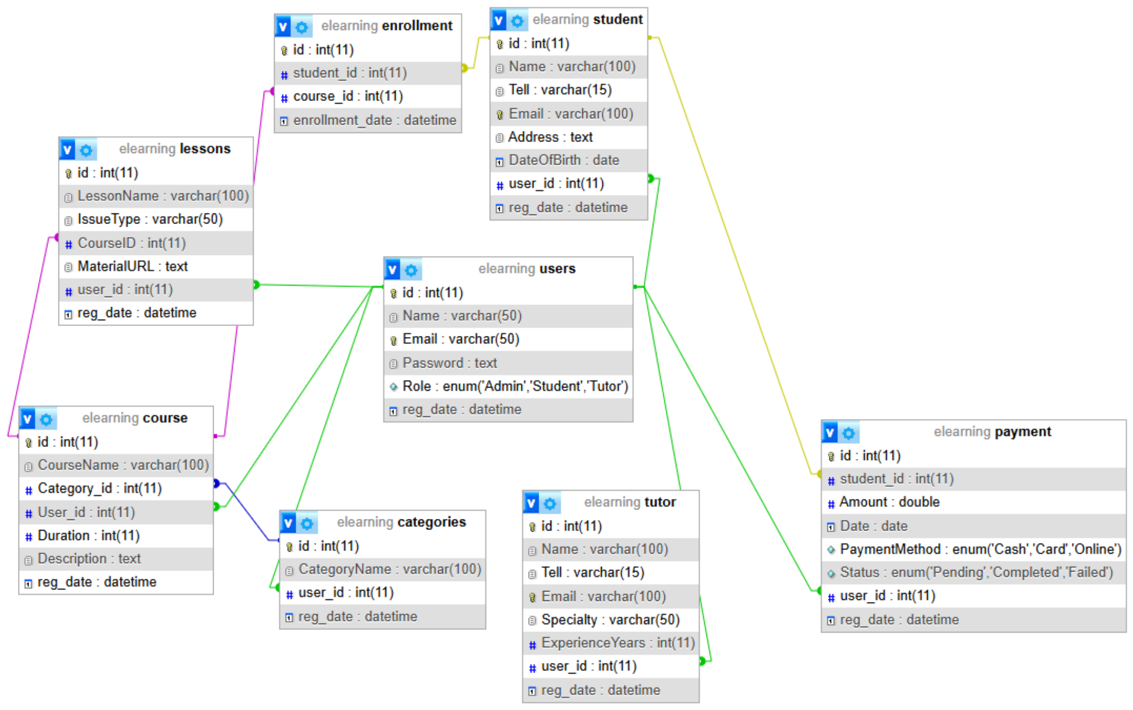

Admin
0
Online Tutoring
Student
Tutor


Login
Search Course
Register Course
Add Course Category
Manage Courses
Application
Enrollment
Make Payment
Report
Manage Users
Student
Tutor
Admin

Class diagram



ERD diagram



onsole

Step 2: Create Database Schema

Users		
Column name	**Data Type**	**Description**
Id(pk)	Int not null primary key auto_increment	
username	Varchar(50)	
email	Varchar(50)	
password	Varchar(50)	
image	Text	
status	Enum('active','inactive')	
Reg_date	timestamp	

Student		
Column name	**Data Type**	**Description**
Id(pk)	Int not null primary key auto_increment	
name	Varchar(50)	
tell	Varchar(50)	
dob	date	
sex	Enum('male,female)	

address	Varchar(50)	
Email	Varchar(50)	
User_id	Int(fk)	References users
Reg_date	timestamp	

Tutor		
Column name	**Data Type**	**Description**
Id(pk)	Int not null primary key auto_increment	
name	Varchar(50)	
tell	Varchar(50)	
Email	Varchar(50)	
speciality	text	
experienceYears	int	
User_id	Int(fk)	References users
Reg_date	timestamp	

Category		
Column name	**Data Type**	**Description**
Id(pk)	Int not null primary key auto_increment	
name	Varchar(50)	
User_id	Int(fk)	References users
Reg_date	timestamp	

Courses		
Column name	**Data Type**	**Description**
Id(pk)	Int not null primary key auto_increment	
Coursename	Varchar(50)	
Duration	Varchar(50)	
description	Varchar(50)	
Category_id	Int(fk)	References categories
User_id	Int(fk)	References users
Reg_date	timestamp	

Schedule		
Column name	**Data Type**	**Description**

Id(pk)	Int not null primary key auto_increment	
Sched_date	date	
Time_in	time	
Time_out	time	
student_id	Int(fk)	References student
Tutor_id	Int(fk)	References courses
User_id	Int(fk)	References users
Reg_date	timestamp	

Charge		
Column name	**Data Type**	**Description**
Id(pk)	Int not null primary key auto_increment	
amount	double	
status	Enum('paid,unpaid)	
student_id	Int(fk)	References student
User_id	Int(fk)	References users
Reg_date	timestamp	

Receipt		
Column name	**Data Type**	**Description**
Id(pk)	Int not null primary key auto_increment	
Charge_id	Int(fk)	References charge
amount	double	
Account_id	Int(fk)	References accounts
description	text	
Receipt_date	date	
User_id	Int(fk)	References users
Reg_date	timestamp	

Accounts		
Column name	**Data Type**	**Description**
Id(pk)	Int not null primary key auto_increment	
Acc_number	Int	
instituition	Varchar(50)	
balances	double	
User_id	Int(fk)	References users
Reg_date	timestamp	

Data Flow:

Students log in → Select available tutor and course → Record usage → Report issues if any.