| | **Software Design Specifications**  Tutorly-Online Tuition Platform  **Version: 1.0**   | Project Code |  | | --- | --- | | Supervisor | Ms. Fizza Aqeel | | Co Supervisor |  | | Project Team | Hashir Khan 22K-4419  Raahim Hussain 22K-4289  Kainat Faisal 22K-4405 | | Submission Date | 6th of May, 2025 | | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |   **Document History**   | Version | Name of Person | Date | Description of change | | --- | --- | --- | --- | | 0.1 | Hashir Khan | 2024-11-15 | Initial draft created. | | 0.2 | Kainat Faisal | 2024-11-20 | Added Non-Functional Requirements. | | 0.3 | Raahim Hussain | 2024-11-25 | Updated Use Case UC-1.2 (Payment Flow). | | 0.4 | Kainat Faisal | 2024-12-01 | Revised Class Diagrams and Data Dictionary. | | 1.0 | Entire Team | 2024-12-10 | Final approved version. |   **Distribution List**   | **Name** | **Role** | | --- | --- | | Ms. Fizza Aqeel | Supervisor | |  | Co Supervisor | |  |  |       **Document Sign-Off**  *[Following table will contain sign-off details of document. Once the document is prepared and revised, this should be signed-off by the sign-off authority.*  *Any subsequent changes in the document after the first sign-off should again get a formal sign-off by the authorities.]*   | **Version** | **Sign-off Authority** | **Project Role** | **Signature** | **Sign-off Date** | | --- | --- | --- | --- | --- | |  |  |  |  |  | |  |  |  |  |  | |  |  |  |  |  | |  |  |  |  |  | |  |  |  |  |  | |  |  |  |  |  | |  |  |  |  |  | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

**Document Information**

| **Category** | **Information** |
| --- | --- |
| Customer | FAST-NU |
| Project | Tutorly- Tuition Learning Platform |
| Document | Software Design Specification |
| Document Version | 1.0 |
| Status | Draft |
| Author(s) | Raahim Hussain, Hashir Khan, Kainat Faisal |
| Approver(s) | Ms. Fizza Aqeel |
| Issue Date | 06-05-2025 |
| Document Location |  |
| Distribution | Advisor  Project Coordinator’s Office (through Advisor) |

**Definition of Terms, Acronyms and Abbreviations**

| **Term** | **Description** |
| --- | --- |
| JWT | JSON Web Token – A secure method for authentication and authorization. |
| Stripe | Third-party payment processing service integrated into the platform. |
| API | Application Programming Interface |
| PK | Primary Key |
| FK | Foreign Key |
| HTTPS | Hypertext Transfer Protocol Secure |
|  |  |
|  |  |

**Table of Contents**

[**1**](#_hyemrjkgdu7f) **Introduction 8**

[*1.1*](#_xof60q4egvwc) *Purpose of Document 8*

[*1.2*](#_vgnh4sy5wz9l) *Intended Audience 8*

[*1.3*](#_e2qhuv9fdxlv) *Document Convention 8*

[*1.4*](#_c8xlyydmwmzf) *Project Overview 8*

[*1.5*](#_etxy5d6kk8v9) *Scope 8*

[**2**](#_qlevjzqi4jir) **Design Considerations 9**

[*2.1*](#_fph93pv31oab) *Assumptions and Dependencies 9*

[*2.2*](#_ec51dkdhhmop) *Risks and Volatile Areas 9*

[**3**](#_k87tjof6ur6k) **System Architecture 10**

[*3.1*](#_zc4tquvxd6k4) *System Level Architecture 10*

[*3.2*](#_4d7g1u5r2gir) *Software Architecture 10*

[**4**](#_sq3d0sd84m66) **Design Strategy 11**

[**5**](#_mi9v9c50kjc4) **Detailed System Design 12**

[*5.1*](#_hbak3c7vqrz) *Database Design 12*

[5.1.1](#_djxwy84272e) ER Diagram 12

[5.1.2](#_5v3c3okc7qtr) Data Dictionary 12

[5.1.2.1](#_n1euivv2p7xc) Data 1 12

[5.1.2.2](#_i0tilhp1jps0) Data 2 12

[5.1.2.3](#_ftvmadguph5n) Data n 12

[*5.2*](#_qkh3m4acfy29) *Application Design 14*

[5.2.1](#_fx20nnakttrt) Sequence Diagram 14

[5.2.1.1](#_120muvpk6meq) <Sequence Diagram 1> 14

[5.2.1.2](#_g0xzipubbh14) <Sequence Diagram 2> 14

[5.2.1.3](#_r3c1y0g8nhp0) <Sequence Diagram n> 14

[5.2.2](#_pm0fnltiz4nq) State Diagram 14

[5.2.2.1](#_4n997xx5iarw) <State Diagram 1> 14

[5.2.2.2](#_qff2028cz63h) <State Diagram 2> 14

[5.2.2.3](#_ctstvj2jjdzb) <State Diagram n> 14

[**6**](#_y9e7mjmturik) **References 15**

[**7**](#_4qpucaciao21) **Appendices 16**

# **Introduction**

## ***Purpose of Document***

This document serves as a **Software Design Specification (SDS)** for **Tutorly: A Tuition Learning Platform**. It outlines the detailed design of the system, including its architecture, database schema, core features, and functionalities. The purpose is to provide a clear blueprint for developers, designers, and stakeholders to ensure the system is built according to requirements and best practices. The design methodology adopted for this project is **Object-Oriented Design (OOD)**, which emphasizes modularity, reusability, and scalability through classes, objects, and their interactions.

## ***Intended Audience***

The intended audience for this document includes:

1. **Developers**: To implement the system based on the specified design.
2. **Designers**: To create user interfaces aligned with the system's functionality.
3. **Project Managers**: To oversee development and ensure alignment with project goals.
4. **Stakeholders**: To review the system's capabilities and provide feedback.
5. **Quality Assurance (QA) Team**: To design test cases based on the system's features.

## ***Document Convention***

1. **Font**: Inter
2. **Font Size**: 11pt for body text, 14pt for headings, and 16pt for main titles.
3. **Emphasis**: Bold for key terms, italics for definitions, and bullet points/ numbered list for lists.

## ***Project Overview***

The **Tutorly Tuition Learning Platform** is a **web-based Management Information System (MIS)**designed to connect students and teachers in Pakistan. Key functionalities include:

* **Teacher/Student Registration**: Profiles with details like subjects, availability, and location.
* **Advanced Search**: Filters for subjects, grade levels, and teaching modes (online/physical).
* **Contract Management**: Creation, negotiation, and tracking of tutoring contracts.
* **Real-time Communication**: Chat system and notifications for updates.
* **AI Chatbot**: Gemini LLM-based assistance for users.
* **Payment Tracking**: Secure handling of contract payments.

The design approach follows **Object-Oriented Methodology**, with modular components (e.g., User, Contract, Payment classes) and relational databases (PostgreSQL) for data management

## ***Scop***e

**In Scope:**

1. User registration and profile management for teachers and students.
2. Search functionality with filters (subjects, location, availability).
3. Contract creation, acceptance, and payment tracking.
4. Real-time chat and notification systems.
5. AI chatbot for user support.
6. Rating, reviews, and session history.

**Out of Scope:**

1. Offline functionality (requires internet connectivity).
2. Video conferencing tools (integration with third-party apps like Zoom is assumed).
3. Advanced analytics (e.g., predictive performance tracking).
4. Multi-language support beyond basic UI localization.

This document ensures all stakeholders share a unified understanding of the system's design and limitations.

# **Design Considerations**

1. **Modularity**: The system should be designed with modular components to facilitate independent development, testing, and maintenance. This includes separating the frontend, backend, and database layers.
2. **Scalability**: The architecture must support horizontal scaling to handle increasing loads, especially during peak usage times (e.g., exam seasons). This involves leveraging serverless functions and auto-scaling capabilities provided by Vercel.
3. **Security**: Design must incorporate robust security measures, such as encryption for data in transit (HTTPS) and at rest, secure authentication (JWT), and role-based access control (RBAC).
4. **Performance**: The system should be optimized for low latency, especially for search functionality and payment processing. Caching mechanisms (e.g., Redis) may be considered for frequently accessed data.
5. **Interoperability**: The system must seamlessly integrate with third-party services like Stripe for payments and PostgreSQL for database operations. APIs should be well-documented and versioned.
6. **User Experience**: The UI should be intuitive and responsive, ensuring a smooth experience across different devices and browsers.
7. **Error Handling**: Global error handling strategies should be implemented to manage failures gracefully, with logging and monitoring for debugging and performance tuning.

## ***Assumptions and Dependencies***

1. **Third-Party Services**: The design assumes uninterrupted availability of Stripe for payment processing and Railway for PostgreSQL database hosting. Contingency plans (e.g., fallback payment gateways) may be needed.
2. **Internet Connectivity**: The system assumes stable internet connectivity for both clients and servers. Offline functionality (e.g., caching UI elements) is not prioritized.
3. **Browser Compatibility**: The frontend is designed for modern browsers (Chrome, Firefox) and may not fully support older versions without polyfills.
4. **Database Schema Stability**: The initial design assumes a fixed PostgreSQL schema. Schema migration tools (e.g., Flyway) will be required for future changes.
5. **Serverless Cold Starts**: The design accounts for potential latency due to serverless function cold starts on Vercel, with optimizations like keeping functions warm.

## ***Risks and Volatile Areas***

1. **New Requirements**: Future features like video conferencing or additional payment methods may require significant architectural changes. The system should be designed with extensibility in mind (e.g., plugin architecture).
2. **Technology Changes**: Updates to Next.js, PostgreSQL, or Stripe APIs could impact the system. The design should isolate third-party dependencies behind interfaces.
3. **Scalability Limits**: Auto-scaling on Vercel may have limits under extreme loads. Monitoring and manual scaling adjustments may be necessary.
4. **Security Vulnerabilities**: Emerging threats (e.g., new attack vectors) may require updates to authentication or encryption mechanisms. Regular security audits are recommended.
5. **Data Privacy Regulations**: Changes in laws (e.g., GDPR) may necessitate design modifications for data storage or user consent management.

## **System Architecture**

The **Tutorly Online Tuition Platform** follows a **modular, three-tier architecture**, ensuring clear separation of concerns, scalability, and maintainability. The system is decomposed into subsystems that interact via well-defined interfaces to deliver the desired functionality.

## ***System Level Architecture***

##### Subsystems:

1. **Frontend (Next.js on Vercel)**
   * **Responsibilities**:
     + User interfaces for students, teachers, and admins.
     + Rendering dynamic content (teacher search, contracts, reviews).
     + Handling user input and interactions (e.g., authentication, payment initiation).
   * **Interfaces**:
     + REST APIs to communicate with the backend.
     + Stripe SDK for payment processing.
2. **Backend (Serverless Functions on Vercel)**
   * **Responsibilities**:
     + Authentication/authorization (JWT validation, role-based access).
     + Business logic (contract management, review handling, availability updates).
     + Integration with PostgreSQL and Stripe.
   * **Interfaces**:
     + REST APIs for frontend requests.
     + PostgreSQL queries for data persistence.
     + Stripe API for payment processing.
3. **Database (PostgreSQL on Railway)**
   * **Responsibilities**:
     + Storing user profiles, contracts, reviews, and availability schedules.
     + Ensuring data integrity and security.
   * **Interfaces**:
     + SQL queries from the backend.

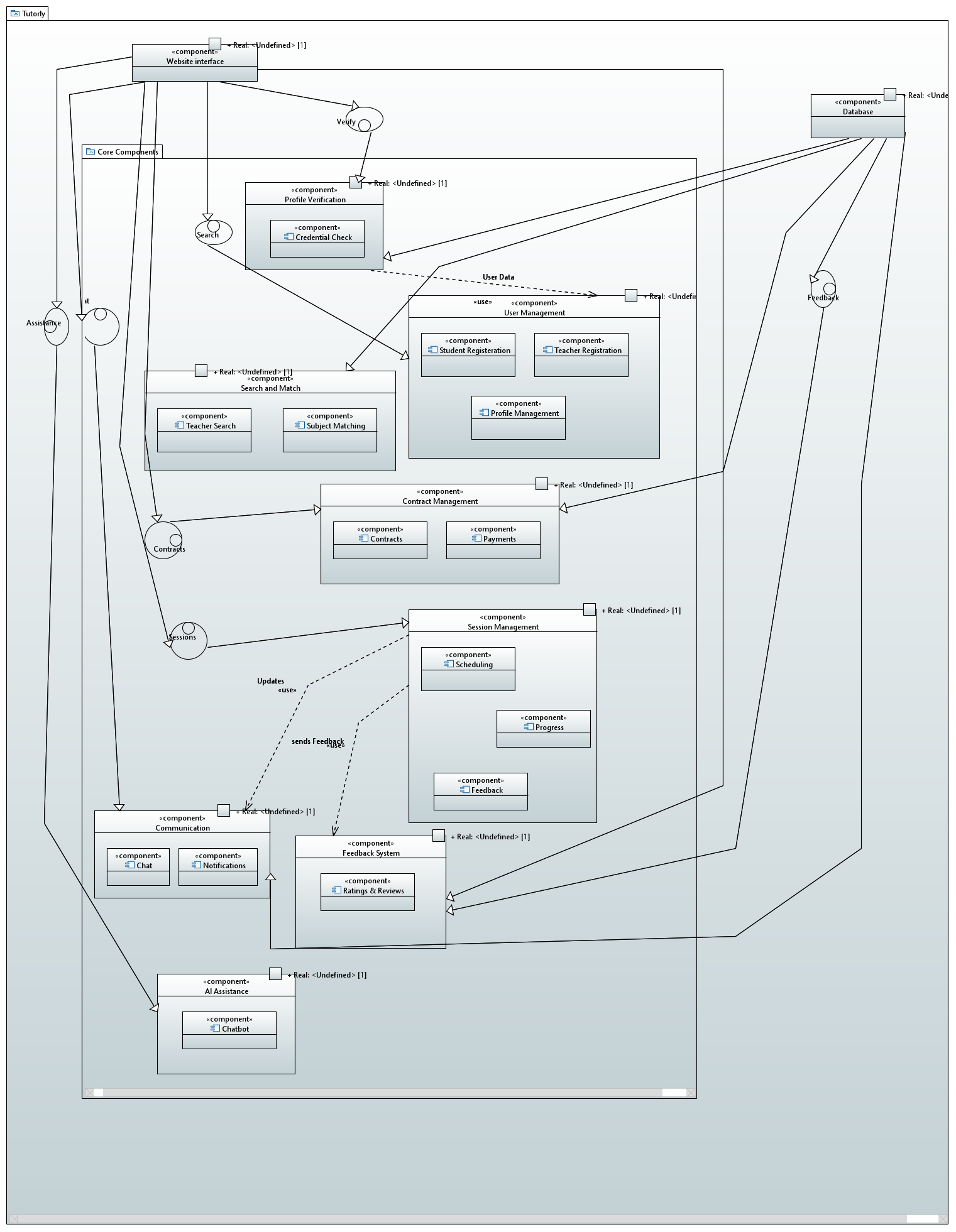
##### External Systems:

* **Stripe**: Handles payment processing via webhooks and API calls.
* **Vercel**: Hosts frontend and backend with auto-scaling capabilities.

##### Global Design Strategies:

* **Error Handling**: Centralized logging and error responses (HTTP status codes).
* **Security**: HTTPS for all communications, JWT for session management.
* **Scalability**: Serverless architecture (Vercel) and managed PostgreSQL (Railway).

***Component Diagram***

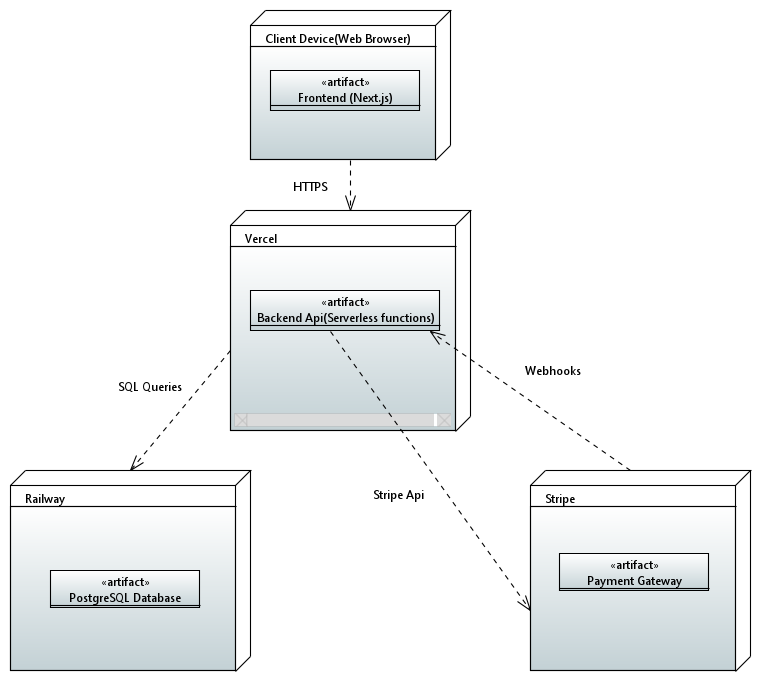


## ***Software Architecture***

The software follows a **3-tier architecture** to separate concerns and ensure modularity.

##### Layers:

1. **User Interface Layer (Frontend)**
   * **Technologies**: Next.js, React.
   * **Responsibilities**:
     + Render views (login, dashboard, search).
     + Capture user input and validate client-side.
2. **Middle Tier (Backend)**
   * **Technologies**: Node.js (Serverless Functions).
   * **Responsibilities**:
     + Authentication (JWT generation/validation).
     + Business logic (contracts, reviews, teacher search).
     + API orchestration (Stripe, database).
3. **Data Access Layer (Database)**
   * **Technologies**: PostgreSQL.
   * **Responsibilities**:
     + CRUD operations for user data, contracts, and reviews.
     + Enforce data constraints and relationships.

***Deployment Diagram*****

# **Design Strategy**

**1. Architectural Approach**

* Layered architecture (UI, Business Logic, Data)
* Microservices for key functions (Auth, Payments)
* REST APIs for inter-component communication

**2. Key Decisions & Trade-offs**

* Frontend: Next.js for SSR (better SEO vs higher complexity)
* Backend: Serverless (scalability vs cold starts)
* DB: PostgreSQL (ACID compliance vs NoSQL flexibility)
* Payments: Stripe integration (speed vs vendor lock-in)

**3. Future Considerations**

* Plugin architecture for new features
* Webhooks for external integrations
* Shared auth library across services

**4. Data & Concurrency**

* Centralized DB with role-based access
* Optimistic UI updates for responsiveness
* Queue-based processing for payments

**5. UI Strategy**

* Role-based dashboards
* Progressive Web App capabilities
* Responsive design framework

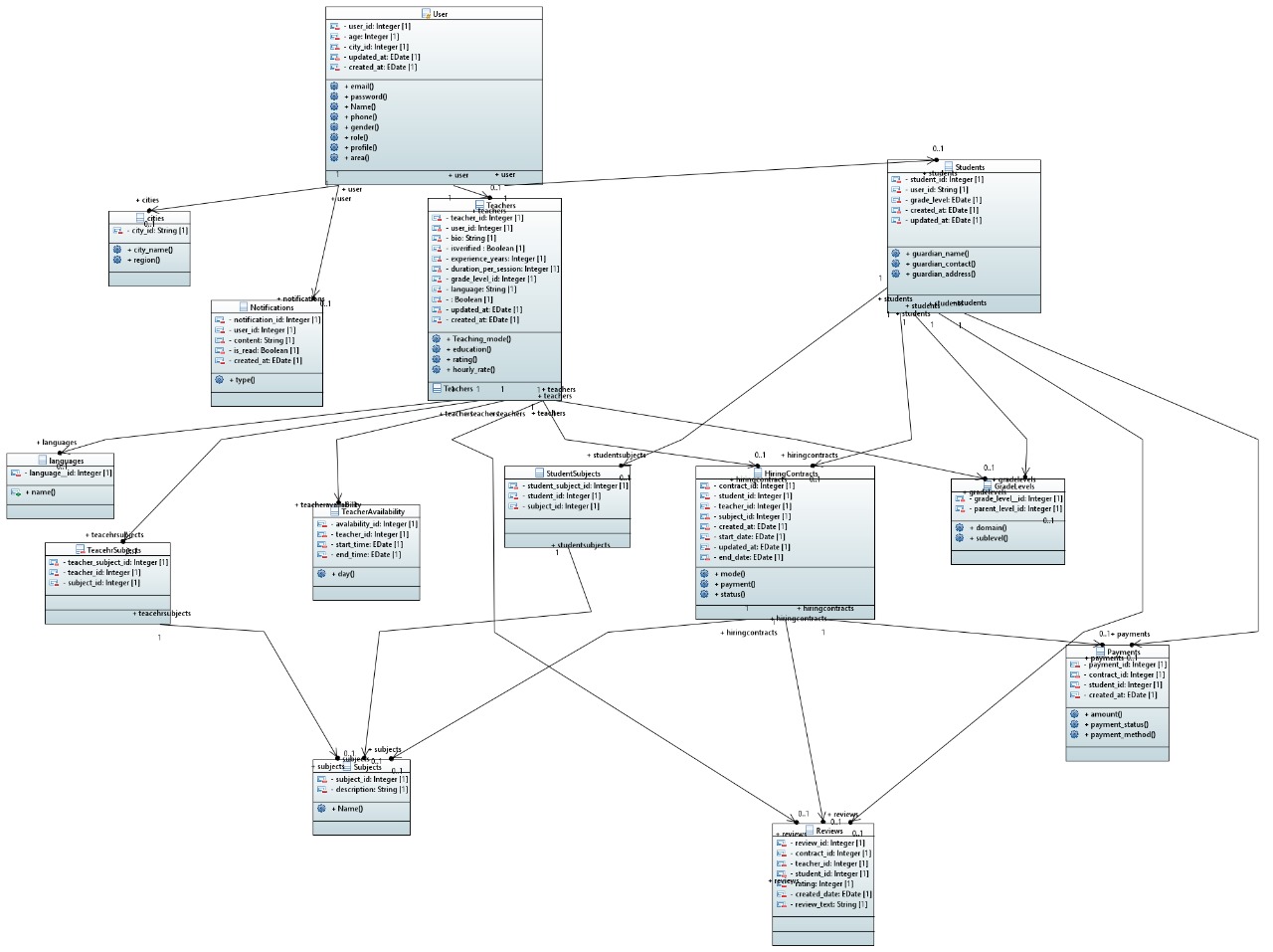
**Trade-off Summary**

1. Faster development vs vendor dependencies
2. Auto-scaling vs cold start latency
3. Strict data consistency vs performance

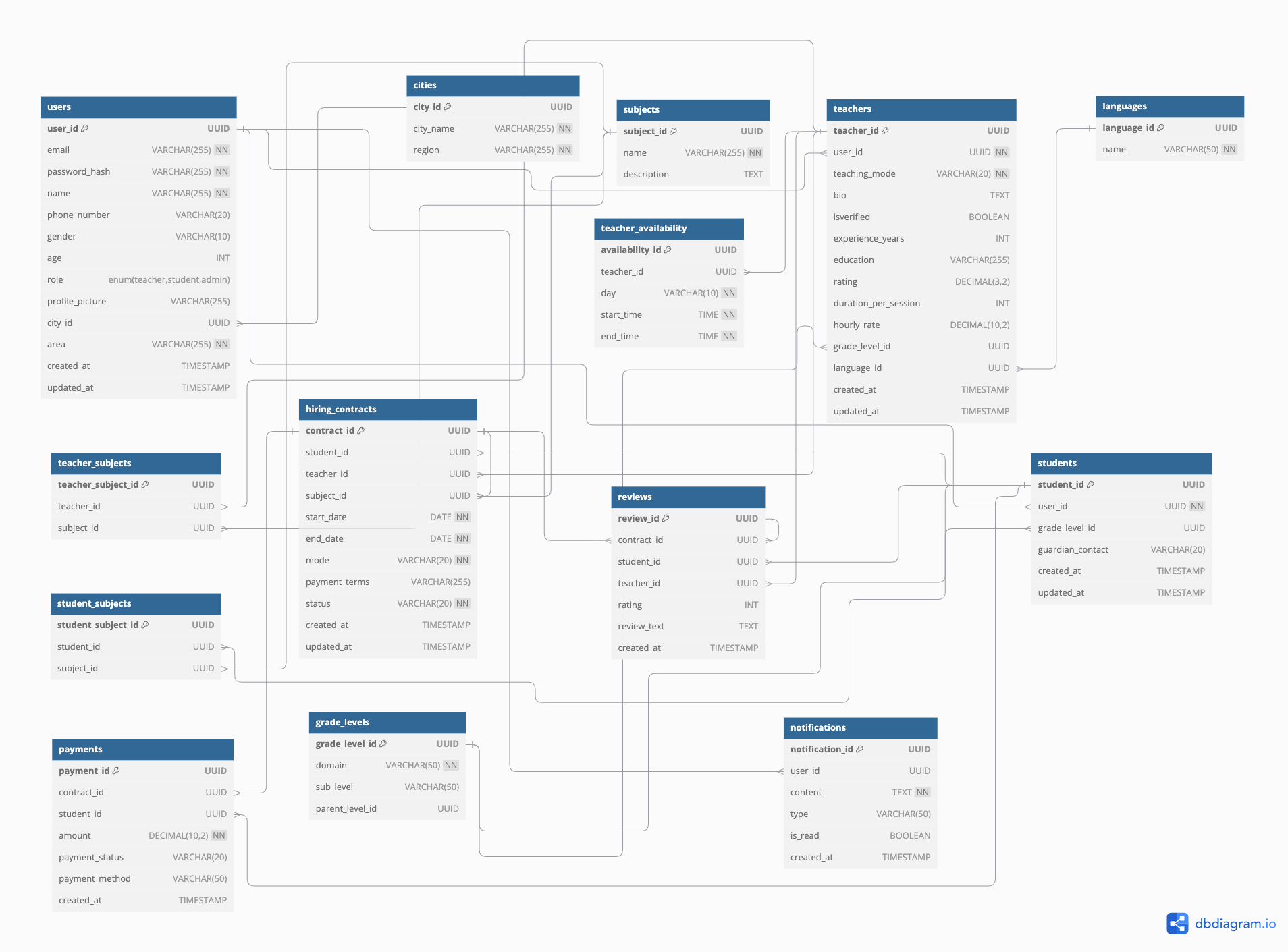
# **Detailed System Design**

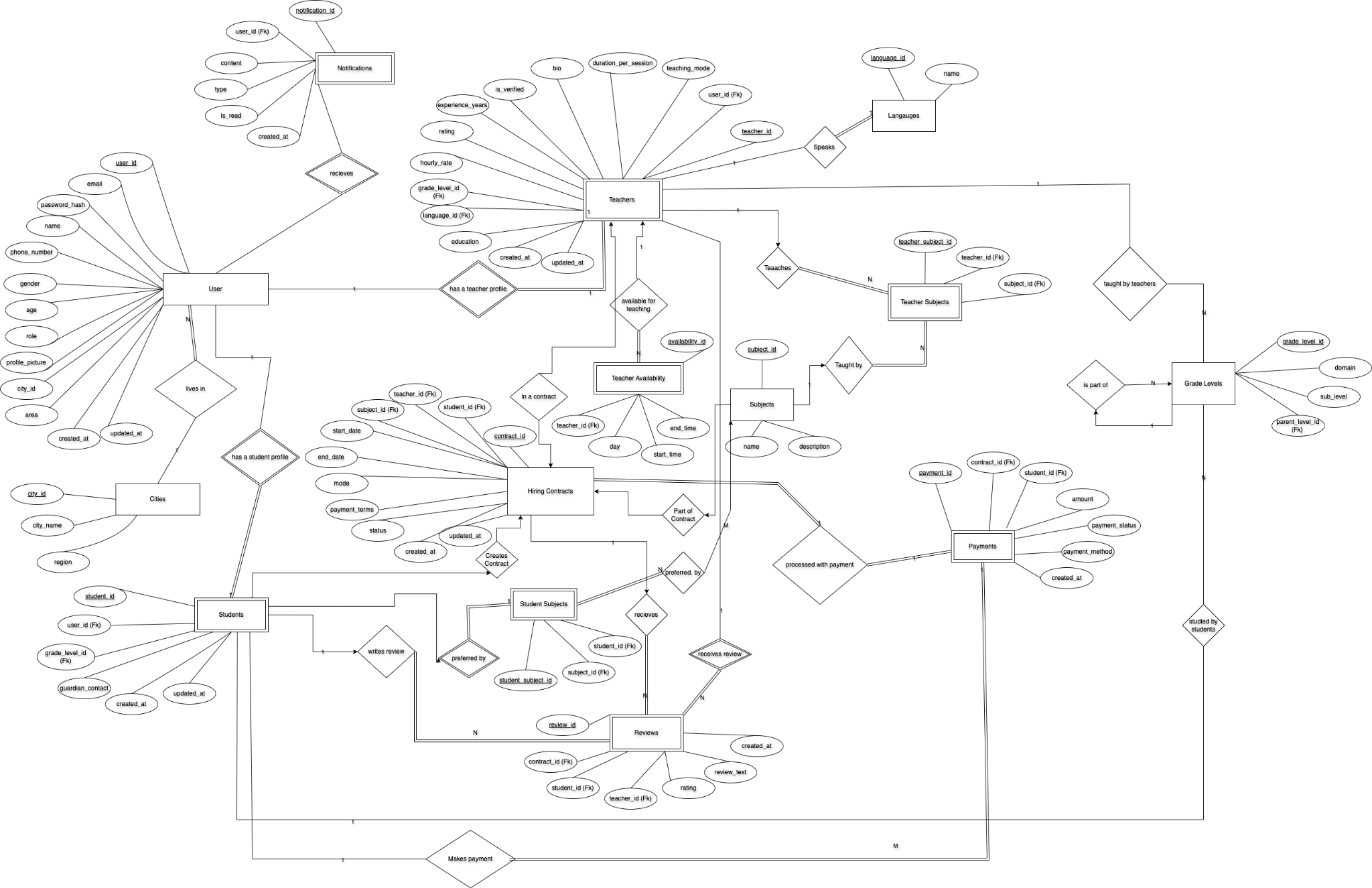
## ***Database Design***

### Class Diagram



### **ER Diagrams**



**

### **Data Dictionary**

**Data 1**

**Name:** users  
**Alias:** User Information Table

| Column Name | Description | Type | Length | Nullable | Default Value | Key Type |
| --- | --- | --- | --- | --- | --- | --- |
| user\_id | Unique identifier for user | UUID | — | No | — | PK |
| email | Email address of user | VARCHAR | 255 | No | — |  |
| password\_hash | Hashed password | VARCHAR | 255 | No | — |  |
| name | Full name | VARCHAR | 255 | No | — |  |
| phone\_number | Contact number | VARCHAR | 20 | No | — |  |
| gender | Gender of the user | VARCHAR | 10 | No | — |  |
| age | Age of the user | INT | — | No | — |  |
| role | Role of the user | ENUM | — | No | — |  |
| profile\_picture | URL to profile image | VARCHAR | 255 | Yes | — |  |
| city\_id | Reference to user's city | UUID | — | No | — | FK |
| area | Area within city | VARCHAR | 255 | No | — |  |
| created\_at | Record creation timestamp | TIMESTAMP | — | No | CURRENT\_TIME |  |
| updated\_at | Record update timestamp | TIMESTAMP | — | Yes | — |  |

**Data 2**

**Name:** students  
**Alias:** Student Profile Table

| Column Name | Description | Type | Length | Nullable | Default Value | Key Type |
| --- | --- | --- | --- | --- | --- | --- |
| student\_id | Unique identifier for student | UUID | — | No | — | PK |
| user\_id | Link to user profile | UUID | — | No | — | FK |
| grade\_level\_id | Academic grade level | UUID | — | Yes | — | FK |
| guardian\_contact | Emergency/guardian contact | VARCHAR | 20 | Yes | — |  |
| created\_at | Record creation timestamp | TIMESTAMP | — | No | CURRENT\_TIME |  |
| updated\_at | Record update timestamp | TIMESTAMP | — | Yes | — |  |

**Data 3**

**Name:** teachers  
**Alias:** Teacher Profile Table

| Column Name | Description | Type | Length | Nullable | Default Value | Key Type |
| --- | --- | --- | --- | --- | --- | --- |
| teacher\_id | Unique identifier for teacher | UUID | — | No | — | PK |
| user\_id | Link to user profile | UUID | — | No | — | FK |
| teaching\_mode | Online or Offline | VARCHAR | 20 | No | — |  |
| bio | Short biography | TEXT | — | Yes | — |  |
| isverified | Whether verified by admin | BOOLEAN | — | Yes | false |  |
| experience\_years | Years of teaching experience | INT | — | Yes | — |  |
| education | Qualification | VARCHAR | 255 | Yes | — |  |
| rating | Average rating | DECIMAL(4,2) | — | Yes | — |  |
| duration\_per\_session | Duration per class in minutes | INT | — | Yes | — |  |
| hourly\_rate | Hourly teaching rate | DECIMAL(10,2) | — | Yes | — |  |
| grade\_level\_id | Teaching level | UUID | — | Yes | — | FK |
| language\_id | Language taught | UUID | — | Yes | — | FK |
| created\_at | Record creation timestamp | TIMESTAMP | — | No | CURRENT\_TIME |  |
| updated\_at | Record update timestamp | TIMESTAMP | — | Yes | — |  |

**Data 4**

**Name:** hiring\_contracts  
**Alias:** Tuition Contracts

| Column Name | Description | Type | Length | Nullable | Default Value | Key Type |
| --- | --- | --- | --- | --- | --- | --- |
| contract\_id | Unique ID for the contract | UUID | - | No | - | PK |
| student\_id | Linked student | UUID | - | No | - | FK |
| teacher\_id | Linked teacher | UUID | - | No | - | FK |
| subject\_id | Subject taught | UUID | - | No | - | FK |
| start\_date | Contract start | DATE | - | No | - |  |
| end\_date | Contract end | DATE | - | No | - |  |
| mode | Online/Offline | VARCHAR | 20 | No | - |  |
| payment\_terms | Payment terms description | VARCHAR | 255 | No | - |  |
| status | Contract status | VARCHAR | 20 | No | - |  |
| created\_at | Created timestamp | TIMESTAMP | - | No | CURRENT\_TIME |  |
| updated\_at | Updated timestamp | TIMESTAMP | - | Yes | - |  |

**Data 5**

**Name:** payments  
**Alias:** Tuition Payments

| Column Name | Description | Type | Length | Nullable | Default Value | Key Type |
| --- | --- | --- | --- | --- | --- | --- |
| payment\_id | Unique payment ID | UUID | - | No | - | PK |
| contract\_id | Linked contract | UUID | - | No | - | FK |
| student\_id | Paying student | UUID | - | No | - | FK |
| amount | Paid amount | DECIMAL(10,2) | - | No | - |  |
| payment\_status | e.g., Paid, Pending | VARCHAR | 20 | No | - |  |
| payment\_method | Method used (Card, UPI, etc.) | VARCHAR | 50 | No | - |  |
| created\_at | Timestamp of payment | TIMESTAMP | - | No | CURRENT\_TIME |  |

**Data 6**

**Name:** subjects  
**Alias:** Subject Catalog

| Column Name | Description | Type | Length | Nullable | Default Value | Key Type |
| --- | --- | --- | --- | --- | --- | --- |
| subject\_id | Unique subject ID | UUID | - | No | - | PK |
| name | Subject name | VARCHAR | 255 | No | - |  |
| description | Detailed info about subject | TEXT | - | Yes | - |  |

**Data 7**

**Name:** teacher\_subjects  
**Alias:** Subjects taught by Teachers

| Column Name | Description | Type | Length | Nullable | Default Value | Key Type |
| --- | --- | --- | --- | --- | --- | --- |
| teacher\_subject\_id | Unique record ID | UUID | - | No | - | PK |
| teacher\_id | Linked teacher | UUID | - | No | - | FK |
| subject\_id | Linked subject | UUID | - | No | - | FK |

**Data 8**

**Name:** student\_subjects  
**Alias:** Subjects selected by Students

| Column Name | Description | Type | Length | Nullable | Default Value | Key Type |
| --- | --- | --- | --- | --- | --- | --- |
| student\_subject\_id | Unique record ID | UUID | - | No | - | PK |
| student\_id | Linked student | UUID | - | No | - | FK |
| subject\_id | Linked subject | UUID | - | No | - | FK |

**Data 9**

**Name:** teacher\_availability  
**Alias:** Teacher Weekly Availability

| Column Name | Description | Type | Length | Nullable | Default Value | Key Type |
| --- | --- | --- | --- | --- | --- | --- |
| availability\_id | Unique record ID | UUID | - | No | - | PK |
| teacher\_id | Linked teacher | UUID | - | No | - | FK |
| day | Day of the week | VARCHAR | 10 | No | - |  |
| start\_time | Start time (HH:MM) | TIME | - | No | - |  |
| end\_time | End time (HH:MM) | TIME | - | No | - |  |

**Data 10**

**Name:** reviews  
**Alias:** Teacher Reviews

| Column Name | Description | Type | Length | Nullable | Default Value | Key Type |
| --- | --- | --- | --- | --- | --- | --- |
| review\_id | Unique review ID | UUID | - | No | - | PK |
| contract\_id | Linked contract | UUID | - | No | - | FK |
| student\_id | Reviewer's ID | UUID | - | No | - | FK |
| teacher\_id | Reviewed teacher | UUID | - | No | - | FK |
| rating | Rating value | INT | - | No | - |  |
| review\_text | Text feedback | TEXT | - | Yes | - |  |
| created\_at | Timestamp of review | TIMESTAMP | - | No | CURRENT\_TIME |  |

**Data 11**

**Name:** grade\_levels  
**Alias:** Grade Levels

| Column Name | Description | Type | Length | Nullable | Default Value | Key Type |
| --- | --- | --- | --- | --- | --- | --- |
| grade\_level\_id | Unique grade level ID | UUID | - | No | - | PK |
| domain | Domain (e.g., Math, Science) | VARCHAR | 50 | No | - |  |
| sub\_level | Grade or year level | VARCHAR | 50 | No | - |  |
| parent\_level\_id | Hierarchical link | UUID | - | Yes | - | FK |

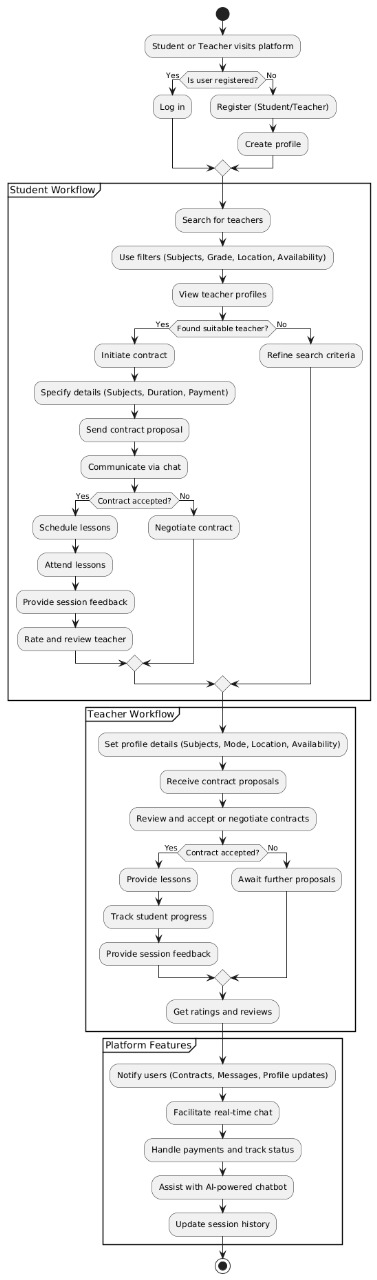
**Data 12**

**Name:** notifications  
**Alias:** Notification Center

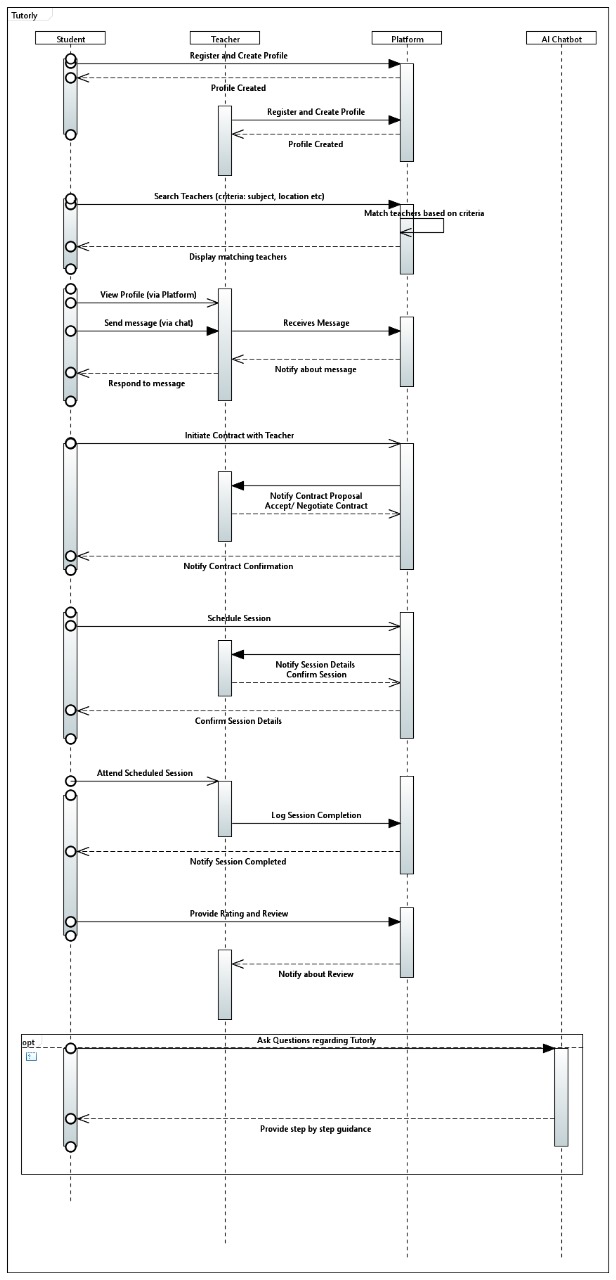
| Column Name | Description | Type | Length | Nullable | Default Value | Key Type |
| --- | --- | --- | --- | --- | --- | --- |
| notification\_id | Unique notification ID | UUID | - | No | - | PK |
| user\_id | Receiver's user ID | UUID | - | No | - | FK |
| content | Message content | TEXT | - | No | - |  |
| type | Notification type (e.g., alert) | VARCHAR | 50 | No | - |  |
| is\_read | Read status | BOOLEAN | - | No | false |  |
| created\_at | Timestamp | TIMESTAMP | - | No | CURRENT\_TIME |  |

***Application Design***

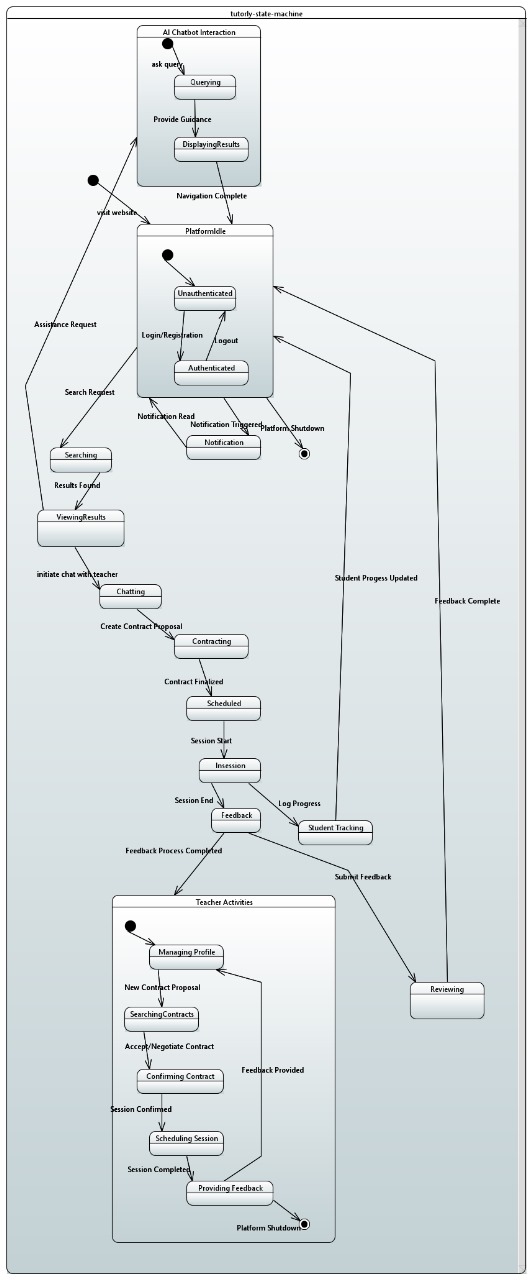
### Activity Diagram



### **Sequence Diagram**



### **State Diagram**



#### **References**

1. **Stripe API Documentation**

Publisher: Stripe, Inc.

Date: Accessed 2024

Availability: <https://stripe.com/docs>

Description: Official documentation for Stripe payment processing API integration.

1. **Next.js Documentation**

Publisher: Vercel

Date: Accessed 2024

Availability: <https://nextjs.org/docs>

Description: Comprehensive guide for Next.js framework used in frontend development.

1. **PostgreSQL Documentation**

Publisher: PostgreSQL Global Development Group

Date: Accessed 2024

Availability: <https://www.postgresql.org/docs>

Description: Official documentation for PostgreSQL database management system.

1. **Railway Documentation**

Publisher: Railway

Date: Accessed 2024

Availability: <https://railway.app/docs>

Description: Deployment and hosting guide for applications on Railway platform.

1. **JSON Web Tokens (JWT) RFC 7519**

Publisher: Internet Engineering Task Force (IETF)

Date: May 2015

Availability: <https://tools.ietf.org/html/rfc7519>

Description: Technical specification for JWT authentication standard.

# **Appendices**

All supporting diagrams, including **ER Diagrams, Sequence Diagrams, Class Diagrams, and Deployment Diagrams**, are available in the project’s Google Drive folder. The link will be shared with authorized stakeholders upon request.

For access, please contact:

Project Lead: [k224419@nu.edu.pk](mailto:k224419@nu.edu.pk)