**CHAPTER 4: SYSTEM DESIGN**

**4.1 Introduction**

This Ethiopian Stock Market Simulation Platform is a comprehensive system designed to emulate real-world stock trading environments. It facilitates user registration, KYC (Know Your Customer) verification, role-based access control, stock listing, order placement, trade execution, portfolio management, regulatory compliance, and real-time notifications. The platform is architected using Django for the backend, Django Rest Framework (DRF) for the API layer, and PostgreSQL for robust data storage. Deploying on a Virtual Private Server (VPS) ensures scalability, security, and high availability. This system is modular, comprising distinct applications such as **Users**, **Stocks**, **Regulations**, **Notifications**, and **Surveillance**, each handling specific functionalities to maintain a clear separation of concerns and facilitate maintainability.

**4.2 Proposed Software Architecture**

**4.2.1 System Decomposition**

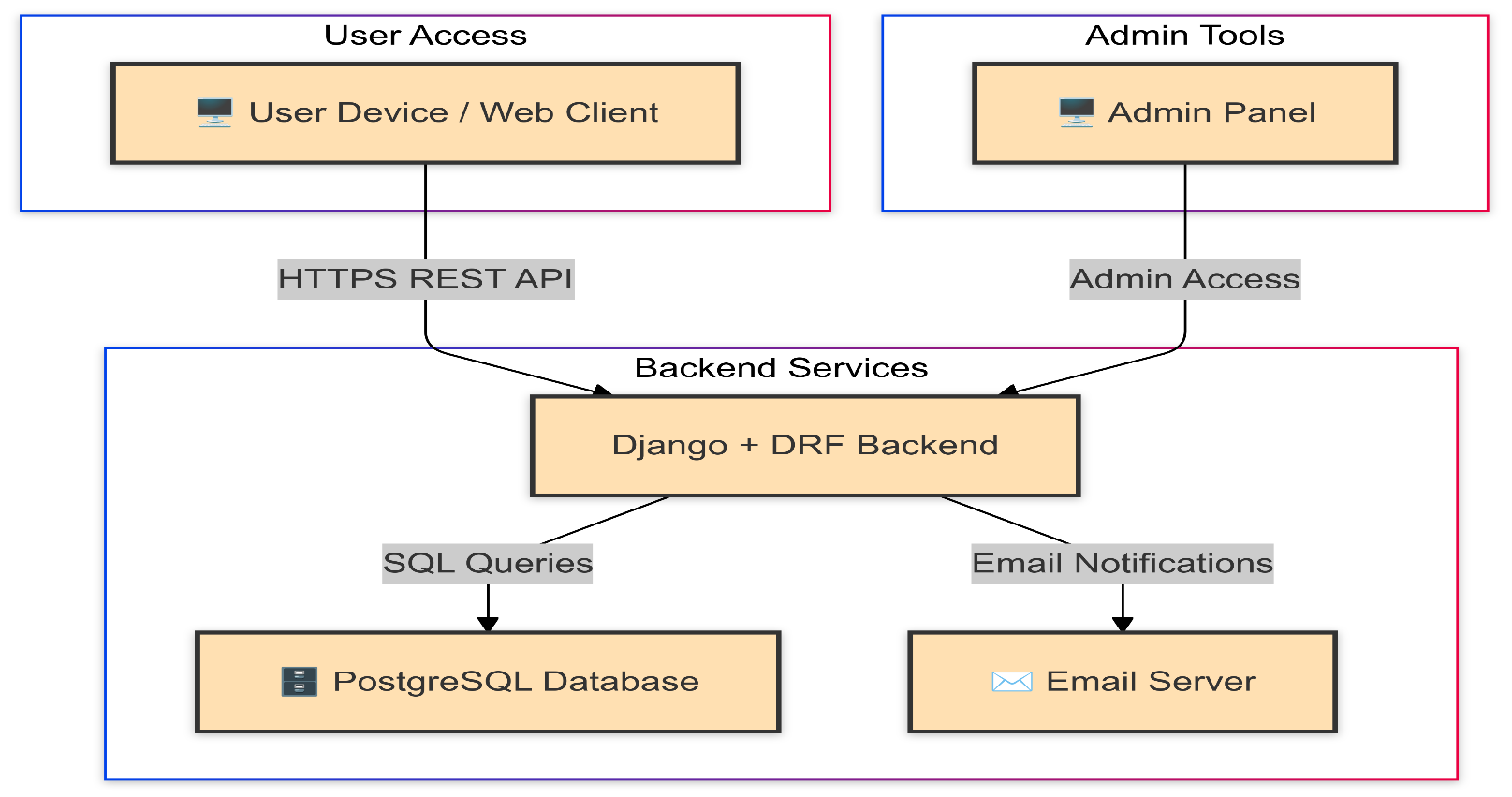
To manage the complexity and enhance scalability, the platform is decomposed into several interconnected Django applications, each responsible for a specific domain:

1. **Users App (Authentication and Authorization)**
   * **Responsibilities**:
     + User registration and authentication.
     + Role assignment (Trader, Company Admin).
     + KYC document handling and verification.
     + OTP generation and validation for secure access.
2. **Stocks App (Trading and Portfolio Management)**
   * **Responsibilities**:
     + Managing listed companies and their stocks.
     + Facilitating order placement (buy/sell) and trade execution.
     + Maintaining user portfolios and handling dividends.
     + Logging daily closing prices and financial disclosures.
     + Detecting and managing suspicious trading activities.
3. **Regulations App (Compliance and Oversight)**
   * **Responsibilities**:
     + Defining and managing regulatory rules (e.g., daily trade limits).
     + Suspending traders from trading activities.
     + Setting and enforcing trading working hours.
4. **Notifications App (Communication)**
   * **Responsibilities**:
     + Sending real-time email notifications for trade executions, KYC approvals, suspensions, and other critical events.
     + Ensuring notifications are dispatched promptly without being stored in the database.
5. **Surveillance App (Monitoring and Security)**
   * **Responsibilities**:
     + Monitoring trades for anomalies and potential fraudulent activities.
     + Flagging suspicious trades for regulator review.
     + Integrating with the Regulations App to enforce suspensions based on surveillance findings.

This modular approach ensures each component can be developed, tested, and maintained independently, promoting scalability and ease of updates.

**4.2.2 Hardware/Software Mapping**

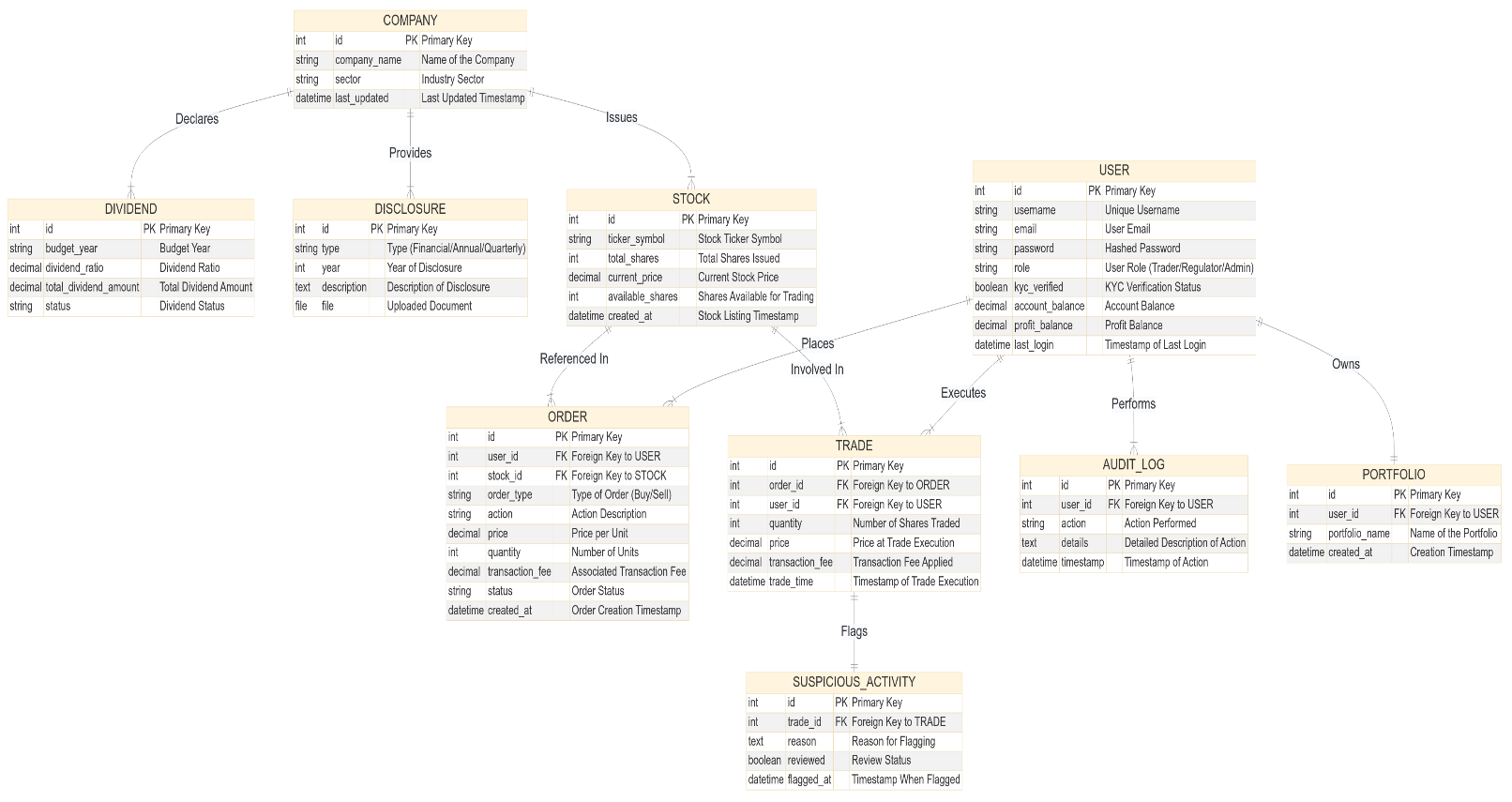
The system leverages a combination of hardware and software resources to ensure seamless operation and performance. The hardware includes servers hosting the Django backend and PostgreSQL database, while the software stack encompasses Django Rest Framework (DRF), PostgreSQL, and an email server for communication. User devices (laptops, desktops, or mobile devices) serve as client interfaces, interacting with the backend through secure HTTPS APIs. Admin access is facilitated via a web-based admin panel hosted on the same server infrastructure.



**Diagram Explanation**

1. **User Access:**
   * **User Device / Web Client:** Represents end-users accessing the platform via web browsers or mobile devices.
2. **Backend Services:**
   * **Django + DRF Backend:** Core application responsible for handling business logic, API endpoints, and serving client requests.
   * **PostgreSQL Database:** Manages all persistent data storage, including user information, orders, trades, portfolios, and regulatory data.
   * **Email Server:** Handles the dispatching of email notifications for events such as trade executions, KYC verifications, and account updates.
3. **Admin Tools:**
   * **Admin Panel:** Interface for administrators and regulators to manage users, oversee trading activities, enforce regulations, and perform other administrative tasks.

**4.2.3 Persistent Data Modeling**

****

**4.2.4 Access Control and Security**

The platform enforces strict access control and security measures to ensure data integrity, user privacy, and regulatory compliance. Access is primarily managed through role-based permissions, ensuring that each user type can only perform actions pertinent to their role.

**User Roles:**

1. **Trader**
   * **Permissions**:
     + Register and verify account via OTP.
     + Upload KYC documents.
     + Place buy/sell orders within set limits.
     + Execute direct purchases from listed companies.
     + View and manage personal portfolio.
     + Receive email notifications on trade executions and KYC status.
   * **Restrictions**:
     + Cannot approve KYC or manage regulations.
     + Cannot upload financial disclosures, publish stocks.
2. **Regulator**
   * **Permissions**:
     + Approve or reject KYC submissions.
     + Define and update regulations (e.g., daily trade limits).
     + Suspend traders from trading activities.
     + Review and mark suspicious activities.
     + View audit logs for compliance monitoring.
     + Define and update working hours for trading.
   * **Restrictions**:
     + Cannot place orders or manage company disclosures.
3. **Company Admin**
   * **Permissions**:
     + Upload and manage financial disclosures for listed companies.
     + Execute direct purchases from the company’s stock pool.
     + View company-specific portfolios.
   * **Restrictions**:
     + Cannot approve KYC or define regulations.
     + Cannot suspend traders or review suspicious activities.

**Access Control and Security:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Feature / Activity | Requires Login | Trader | Regulator | Company Admin |
| Register Account (Sign Up) | No | ✓ | ✗ | ✓ |
| OTP Verification | Yes | ✓ | ✓ | ✓ |
| Upload KYC Documents | Yes | ✓ | ✗ | ✗ |
| Approve/Reject KYC | Yes | ✗ | ✓ | ✗ |
| Place Buy/Sell Orders | Yes | ✓ | ✗ | ✗ |
| Direct Company Stock Purchase | Yes | ✓ | ✗ | ✓ |
| View/Manage Personal Portfolio | Yes | ✓ | ✓\* | ✓ |
| Upload Disclosures | Yes | ✗ | ✗ | ✓ |
| Define/Update Regulations | Yes | ✗ | ✓ | ✗ |
| Suspend Trader | Yes | ✗ | ✓ | ✗ |
| Review/Mark Suspicious Activities | Yes | ✗ | ✓ | ✗ |
| View Audit Logs | Yes | ✗ | ✓ | ✗ |
| Define/Update Working Hours | Yes | ✗ | ✓ | ✗ |
| Receive Email Notifications | Yes | ✓ | ✓ | ✓ |

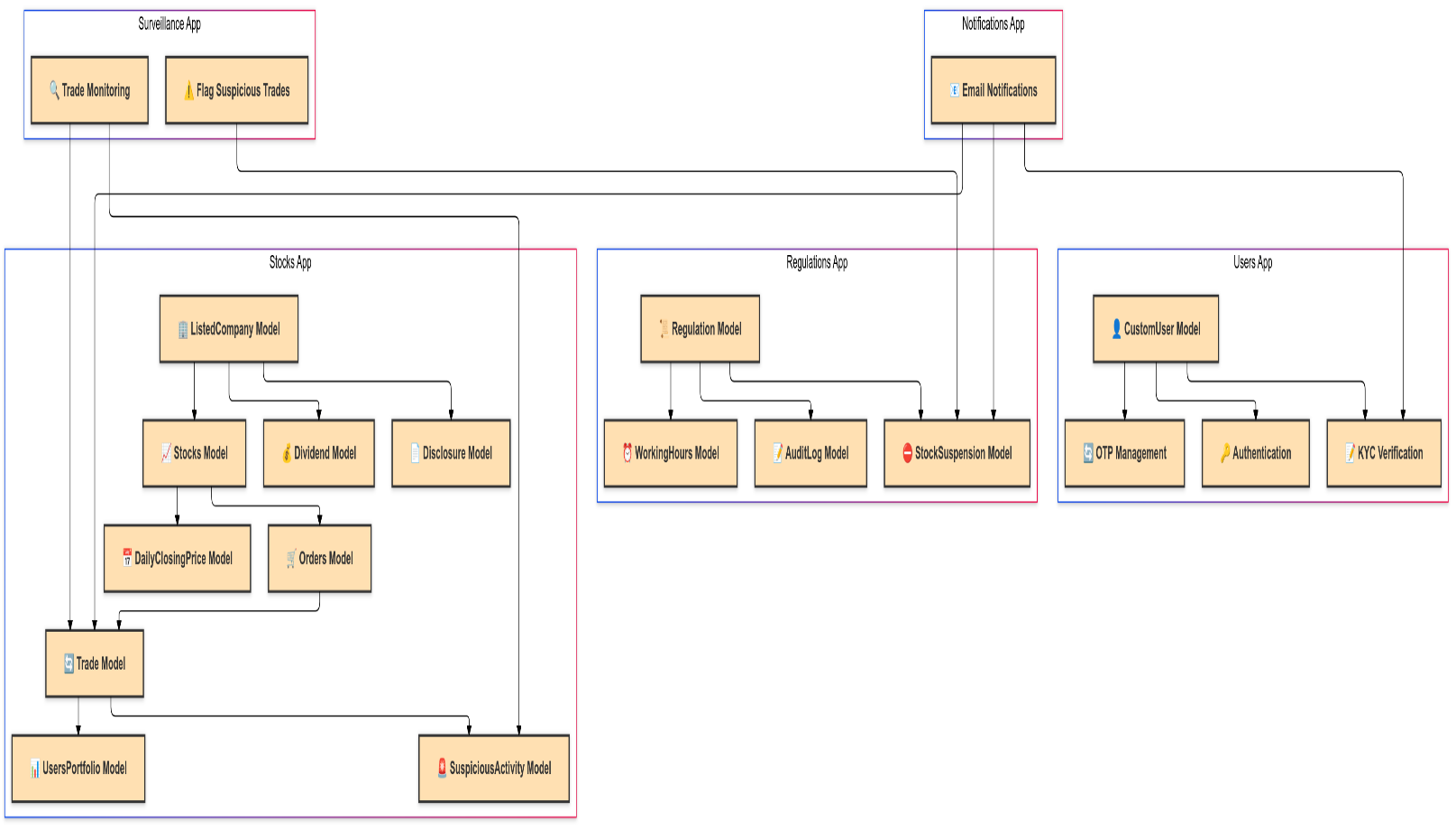
**Note:** Regulators may have view permissions over all traders' portfolios for compliance monitoring.

**Security Measures:**

* **JWT Authentication**: Ensures secure and stateless user sessions.
* **OTP Verification**: Adds an additional layer of security during user registration.
* **Role-Based Access Control (RBAC)**: Restricts actions based on user roles.
* **Input Validation**: Prevents SQL injection, cross-site scripting (XSS), and other common attacks by validating and sanitizing all user inputs.
* **Secure File Handling**: KYC documents and disclosures are securely uploaded and stored with proper access restrictions.
* **Audit Logging**: All critical actions are logged for accountability and forensic analysis.
* **Rate Limiting and Throttling**: Implements rate limiting on API endpoints to prevent abuse and mitigate DDoS attacks

These measures collectively ensure that the platform remains secure, reliable, and compliant with industry standards.

**Package Diagram**

****

**1. Users App**

* **CustomUser Model:** Extends the default user model to include roles (Trader, Regulator, Company Admin), KYC documents, and OTP codes.
* **Authentication:** Manages user login, logout, and session handling.
* **KYC Verification:** Handles the submission and approval of Know Your Customer (KYC) documents to ensure user compliance.
* **OTP Management:** Generates and validates One-Time Passwords for secure user verification processes.

**2. Stocks App**

* **ListedCompany Model:** Represents companies listed on the platform, storing details like company name and sector.
* **Stocks Model:** Manages stock information, including ticker symbols, current prices, total shares, and available shares.
* **Orders Model:** Facilitates the creation and management of buy and sell orders placed by traders.
* **Trade Model:** Executes trades by matching buy and sell orders, updating user portfolios accordingly.
* **UsersPortfolio Model:** Tracks individual user holdings, including the quantity of stocks owned and investment metrics.
* **Disclosure Model:** Allows company admins to upload financial disclosures and related documents.
* **Dividend Model:** Manages the distribution of dividends to shareholders based on company performance.
* **DailyClosingPrice Model:** Logs daily closing prices of stocks for historical reference and analysis.
* **SuspiciousActivity Model:** Flags and manages any trading activities that appear irregular or potentially fraudulent.

**3. Regulations App**

* **Regulation Model:** Stores and manages regulatory rules governing trading activities, such as daily trade limits and operational hours.
* **AuditLog Model:** Records all significant administrative and regulatory actions for accountability and compliance monitoring.
* **StockSuspension Model:** Manages the suspension of traders from trading activities based on regulatory decisions or detected suspicious activities.
* **WorkingHours Model:** Defines permissible trading hours to regulate when users can place orders, ensuring adherence to market operating times.

**4. Notifications App**

* **Email Notifications:** Handles the dispatching of email alerts for various platform events, including trade executions, KYC approvals/rejections, suspensions, and disclosures.

**5. Surveillance App**

* **Trade Monitoring:** Continuously monitors trading activities to identify unusual patterns or volumes that may indicate fraudulent behavior.
* **Flag Suspicious Trades:** Automatically flags trades that meet predefined suspicious criteria for further review by regulators.

## **4.1.3.3 Design Pattern (Architecture Layers)**

The **ETHIOPIAN Stock Market Simulation Platform** utilizes a **layered architectural pattern**, a standard design approach in the industry. This pattern organizes the system into distinct layers, each with specific responsibilities, promoting modularity, scalability, and ease of maintenance.

**Layered Architecture Overview**

1. **Presentation Layer (UI Subsystem)**
   * **Purpose:** Manages all user interactions and displays information to end-users.
   * **Components:** Web Client, Responsive Design, User Dashboards.
   * **Responsibilities:** Render user-friendly interfaces, capture user inputs, and display data from the Business Logic Layer.
2. **Business Logic Layer (User Management & Trading Management Subsystems)**
   * **Purpose:** Encapsulates the platform’s core functionalities and business rules.
   * **Components:** User Management Subsystem, Trading Management Subsystem.
   * **Responsibilities:** Authenticate users, enforce trading rules, and coordinate data flow between Presentation and Data Access Layers.
3. **Data Access Layer (Regulatory and Notification Subsystem)**
   * **Purpose:** Handles data storage, retrieval, and manipulation, ensuring data integrity.
   * **Components:** Regulatory Subsystem, Notification Subsystem, PostgreSQL Database.
   * **Responsibilities:** Perform CRUD operations, maintain audit logs, enforce regulations, and send user notifications.
4. **Infrastructure Layer**
   * **Purpose:** Provides essential services and infrastructure to support other layers.
   * **Components:** Email Server, Admin Panel.
   * **Responsibilities:** Manage communication services, support administrative tasks, and ensure system reliability.

**Benefits of Layered Architecture**

* **Separation of Concerns:** Each layer has a defined role, reducing interdependencies and simplifying maintenance.
* **Scalability:** Layers can be scaled independently to handle increasing demands efficiently.
* **Maintainability:** Isolated layers facilitate easier debugging, testing, and updates without impacting other components.
* **Reusability:** Common functionalities within layers can be reused across different parts of the application, enhancing consistency.
* **Flexibility:** New features or technologies can be integrated by modifying or adding layers without disrupting existing functionalities.

**Implementation in Ethiopian Stock Market Simulation Platform**

* **Presentation Layer:** Implemented through the **UI Subsystem**, offering intuitive interfaces for Traders, Regulators, and Company Admins.
* **Business Logic Layer:** Comprises the **User Management** and **Trading Management Subsystems**, handling authentication, KYC verification, order processing, and trade execution.
* **Data Access Layer:** Managed by the **Regulatory and Notification Subsystem**, interacting with the **PostgreSQL Database** to enforce regulations, maintain audit trails, and manage notifications.
* **Infrastructure Layer:** Supported by the **Email Server** and **Admin Panel**, ensuring efficient communication and administrative oversight.

4.2 Object Design Document (ODD)

The **Object Design Document (ODD)** provides a detailed blueprint of the Ethiopian Stock Market Simulation Platform's system architecture. It outlines the structure, behavior, and interactions of the system's objects, ensuring a clear understanding of how various components collaborate to achieve the platform's functionalities.

**4.2.2 Class Interface**

The **Class Interface** section delineates the primary classes within the Ethiopian Stock Market Simulation Platform, highlighting their attributes, methods, and relationships. This structured overview facilitates a comprehensive understanding of the system's object-oriented design.

**Key Classes Overview**

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name | Attributes | Methods | Relationships |
| CustomUser | - id: Integer - username: String - email: String - password: String - role: String - kyc\_document: File - otp\_code: String - is\_approved: Boolean - kyc\_verified: Boolean - account\_balance: Decimal - profit\_balance: Decimal | - register() - authenticate() - verify\_otp() - upload\_kyc() | One-to-One with UsersPortfolio One-to-Many with Orders, Trades |
| ListedCompany | - id: Integer - company\_name: String - sector: String - last\_updated: DateTime | - add\_stock() - upload\_disclosure() - issue\_dividend() | One-to-One with Stocks One-to-Many with Disclosures |
| Stocks | - id: Integer - ticker\_symbol: String - total\_shares: Integer - current\_price: Decimal - available\_shares: Integer - max\_trader\_buy\_limit: Integer | - publish\_stock() - allocate\_shares() - log\_daily\_closing\_price() | Many-to-One with ListedCompany One-to-Many with Orders, Trades |
| Orders | - id: Integer - order\_type: String - action: String - price: Decimal - quantity: Integer - status: String - transaction\_fee: Decimal | - place\_order() - cancel\_order() - execute\_order() | Many-to-One with CustomUser Many-to-One with Stocks One-to-One with Trade |
| Trade | - id: Integer - trade\_time: DateTime - quantity: Integer - order\_id: Integer - price: Decimal - transaction\_fee: Decimal | - execute\_trade() - update\_portfolio() - flag\_suspicious\_activity() | Many-to-One with CustomUser Many-to-One with Stocks One-to-One with Orders One-to-Many with SuspiciousActivity |
| UsersPortfolio | - id: Integer - user\_id: Integer - quantity: Integer - average\_purchase\_price: Decimal - total\_investment: Decimal | - update\_portfolio() | One-to-One with CustomUser |
| Disclosure | - id: Integer - company\_id: Integer - type: String - year: Integer - file: File - description: String - uploaded\_at: DateTime | - upload\_disclosure() - update\_disclosure() | Many-to-One with ListedCompany |
| Dividend | - id: Integer - company\_id: Integer - budget\_year: Integer - dividend\_ratio: Decimal - total\_dividend\_amount: Decimal - status: String | - issue\_dividend() - disburse\_dividend() | Many-to-One with ListedCompany |
| DailyClosingPrice | - id: Integer - stock\_id: Integer - date: Date - closing\_price: Decimal | - log\_closing\_price() - retrieve\_historical\_prices() | Many-to-One with Stocks |
| Regulation | - id: Integer - name: String - value: String - description: String - created\_by: Integer - created\_at: DateTime - last\_updated: DateTime | - create\_regulation() - update\_regulation() - delete\_regulation() | One-to-Many with AuditLog One-to-Many with StockSuspension |
| StockSuspension | - id: Integer - trader\_id: Integer - stock\_id: Integer - suspension\_type: String - initiator: String - reason: String - is\_active: Boolean - created\_at: DateTime - released\_at: DateTime | - suspend\_trader() - release\_trader() | Many-to-One with Regulation Many-to-One with CustomUser Optional Many-to-One with Stocks |
| WorkingHours | - id: Integer - day\_of\_week: String - start\_time: Time - end\_time: Time | - define\_working\_hours() - update\_working\_hours() | One-to-Many with Orders |
| SuspiciousActivity | - id: Integer - reason: String - flagged\_at: DateTime - reviewed: Boolean - trade\_id: Integer | - flag\_activity() - review\_activity() | Many-to-One with Trade |

**User Interface Design**

The **User Interface Design** section details the visual and interactive elements of the ETHIOPIAN Stock Market Simulation Platform, ensuring a seamless and intuitive user experience for all user roles.

**Overview**

The platform's UI is crafted to be user-friendly, responsive, and accessible, catering to the diverse needs of Traders, Regulators, and Company Admins. The design emphasizes clarity, ease of navigation, and efficient access to essential functionalities.

**Key UI Components**

1. **Login and Registration Pages**
   * **Features:**
     + User authentication via username and password.
     + OTP verification for secure access.
     + KYC document upload during registration.
   * **Design Considerations:**
     + Clean and straightforward layout.
     + Clear instructions and validation messages.
2. **User Dashboards**
   * **Traders:**
     + **Portfolio Overview:** Displays current holdings, total investment, and profit/loss metrics.
     + **Trade History:** Lists past trades with details like stock symbol, quantity, price, and date.
     + **Order Placement:** Interface to place buy/sell orders with options to set order type, quantity, and price.
   * **Regulators:**
     + **Regulation Management:** Tools to define and update trading regulations.
     + **Audit Logs:** Access to comprehensive logs of administrative and regulatory actions.
     + **Suspension Controls:** Interface to suspend or reinstate Traders based on activity reviews.
   * **Company Admins:**
     + **Disclosure Management:** Upload and manage financial disclosures for listed companies.
     + **Dividend Distribution:** Tools to issue and track dividend distributions to shareholders.
3. **Trading Interface**
   * **Features:**
     + Real-time stock price updates and charts.
     + Order book display showing current buy and sell orders.
     + Trade execution confirmation and notifications.
   * **Design Considerations:**
     + Interactive charts for better data visualization.
     + Responsive elements for trading on various devices.
4. **Notification Center**
   * **Features:**
     + Centralized hub for all email notifications and alerts.
     + Status indicators for pending actions like KYC verification or order execution.
   * **Design Considerations:**
     + Organized layout to easily track and manage notifications.
     + Clear categorization based on event types.
5. **Admin Panel**
   * **Features:**
     + Comprehensive management tools for overseeing platform operations.
     + User management capabilities to approve/reject KYC, suspend Traders, and manage roles.
     + Regulatory tools to define and enforce trading rules.
   * **Design Considerations:**
     + Secure access with role-based permissions.
     + Intuitive navigation for efficient administrative tasks.