TAXPAYER'S PORTAL APPLICATION DOCUMENTATION

1. PROJECT OVERVIEW

Brief Description

This is a taxpayer portal web application that enables taxpayers to access online services from the revenue authority. It provides a comprehensive platform for taxpayer registration, TIN (Taxpayer Identification Number) application, and various tax-related services.

Main Features

- Backend: Developed using Spring framework with Java
- Frontend: Implemented using React
- Database: MySQL Community Server
- TIN application and management
- Taxpayer profile creation and management
- Integration with various tax-related services (e.g., VAT calculation, tax returns)
- Secure login and authentication system

Technologies Used

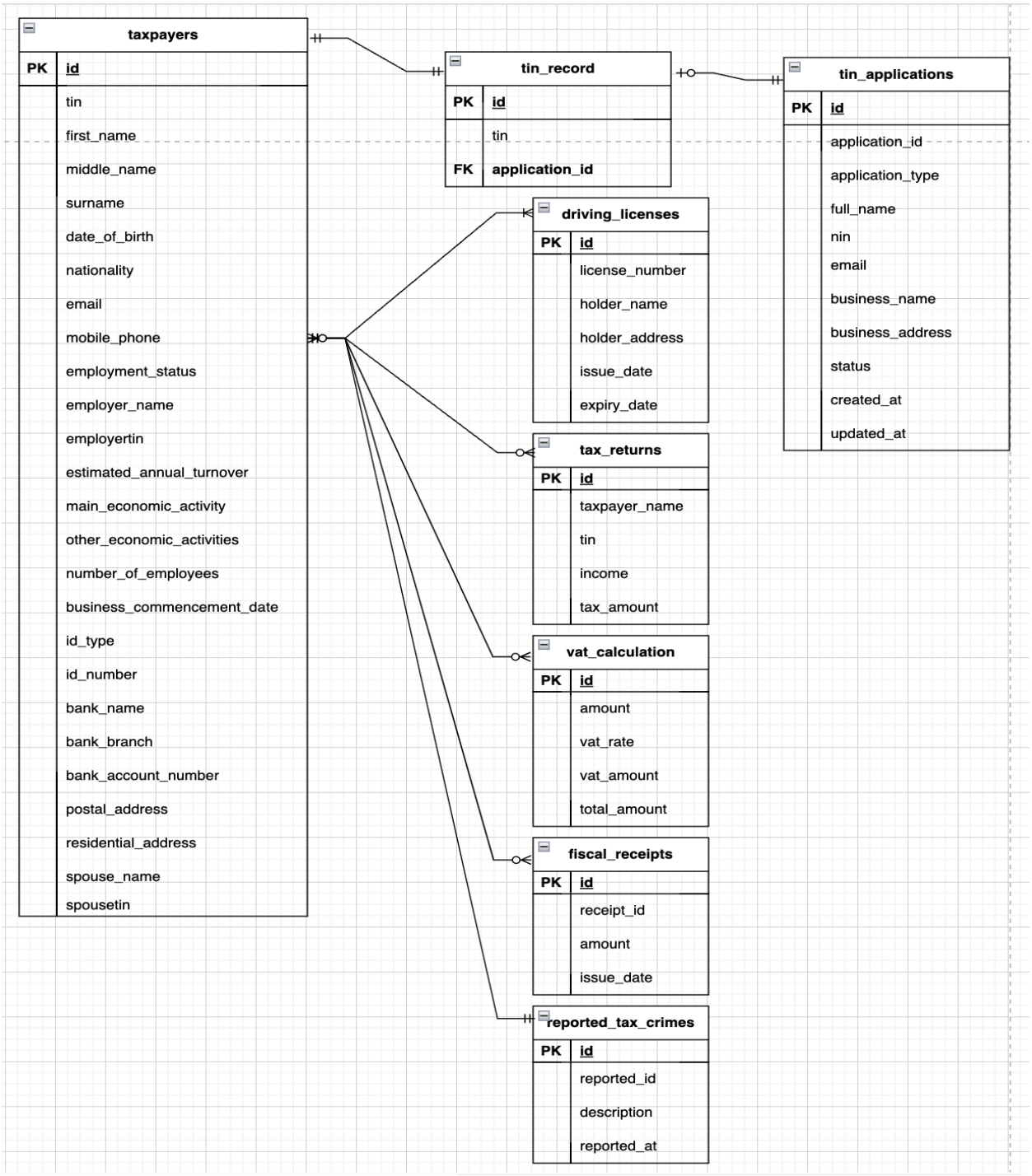
- Backend: Spring Boot (Java)
- Frontend: React
- Database: MySQL Community Server 8.0.26
- API: RESTful APIs for communication between frontend and backend

2. SYSTEM REQUIREMENTS

- Java Development Kit (JDK): Version 17
- Node.js: Version 20.12.2
- MySQL Community Server: Version 8.0.26
- Web Browser: Latest versions of Chrome, Firefox, or Safari recommended

3. DATABASE DESIGN

Entity-Relationship Diagram (ERD)



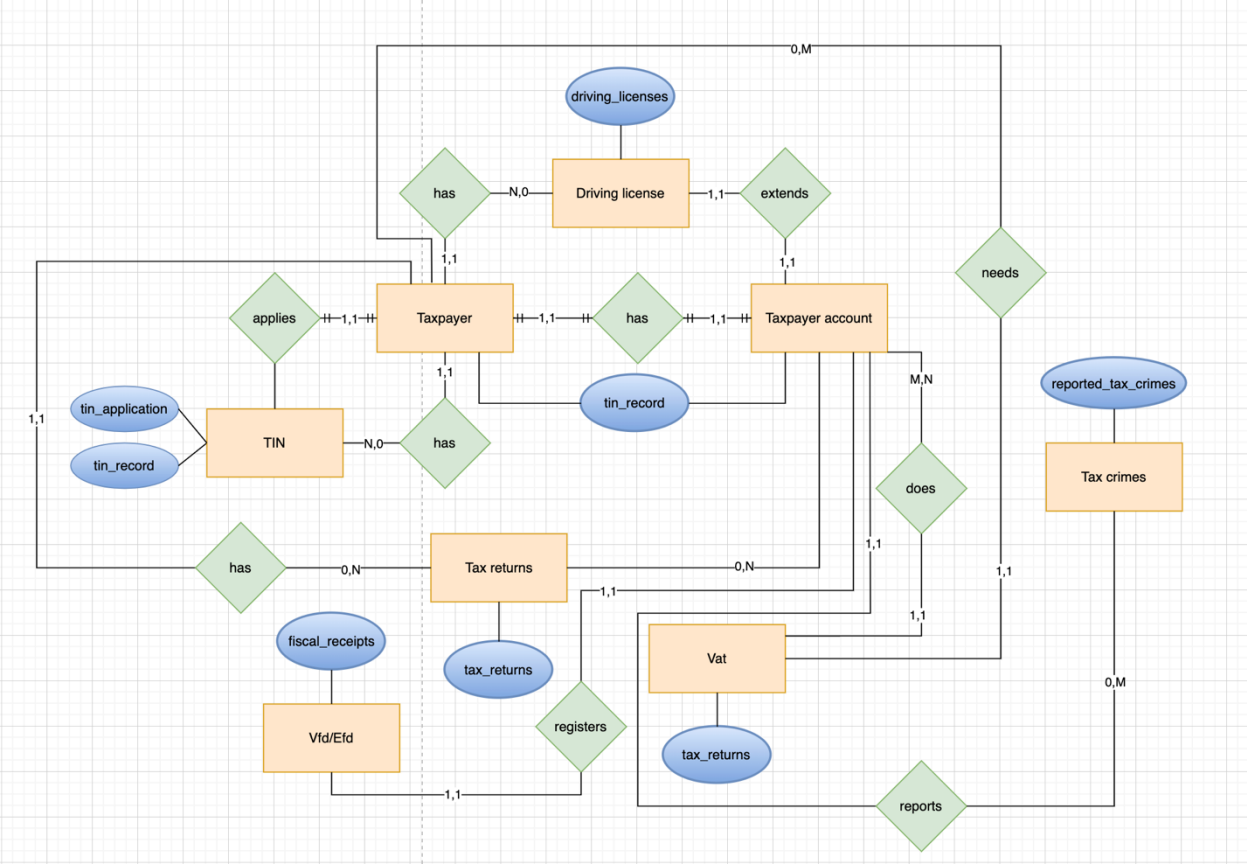
ER model

The system's database design is represented in the provided ERD, key entities include:

1. taxpayers
2. tin_applications
3. tin_record
4. driving_licenses
5. tax_returns
6. vat_calculation
7. fiscal_receipts
8. reported_tax_crimes

Key Relationships

- The `taxpayers` table is the central entity, linked to multiple other entities.
- `tin_record` acts as an intermediary between `tin_applications` and `taxpayers`.
- Various service-related tables (e.g., `driving_licenses`, `tax_returns`) are linked to the `taxpayers` table.



Entity relationship diagram

4. SYSTEM ARCHIRTECTURE

Overview

The system follows microservice architecture with a client-server interaction with a React frontend communicating with a Spring Boot backend via RESTful APIs. The backend interacts with a MySQL database for data persistence.

Components

1. Frontend (React)
    - Login form.

    Provides a secure interface for users to enter their credentials.
    Implements client-side validation for username/email and password.
    Sends authentication requests to the backend API.
    Handles error messages and successful login redirections.

    - Dashboard for accessing various services.

    Acts as the main hub after user authentication.
    Displays a menu or tiles for different tax-related services (e.g., TIN application, tax returns, VAT calculation).
    Implements role-based access control to show relevant services based on user type (individual/entity).
    Provides a user profile section for viewing and updating personal information.

    - Forms for TIN application and taxpayer registration.

    TIN application form captures necessary details for new taxpayers.
    Taxpayer registration form allows existing TIN holders to complete their profile.
    Implements form validation and error handling.
    Sends form data to backend APIs for processing

2. Backend (Spring Boot)
    - API endpoints for user authentication.

    Implements JWT (JSON Web Token) or session-based authentication.
    Provides endpoints for login, logout, and password reset functionalities.
    Integrates with Spring Security for robust authentication and authorization.

    - Independent services for taxpayer registration, TIN application, and other tax-related functionalities.

    TaxpayerService: Handles CRUD operations for taxpayer information.
    TINApplicationService: Manages the TIN application process, including status updates.
    TaxReturnService: Handles submission and retrieval of tax returns.
    VATCalculationService: Provides VAT calculation functionality.

ReportTaxCrimeService: Manages the submission of tax crime reports.

- Data access layer for interacting with the MySQL database.

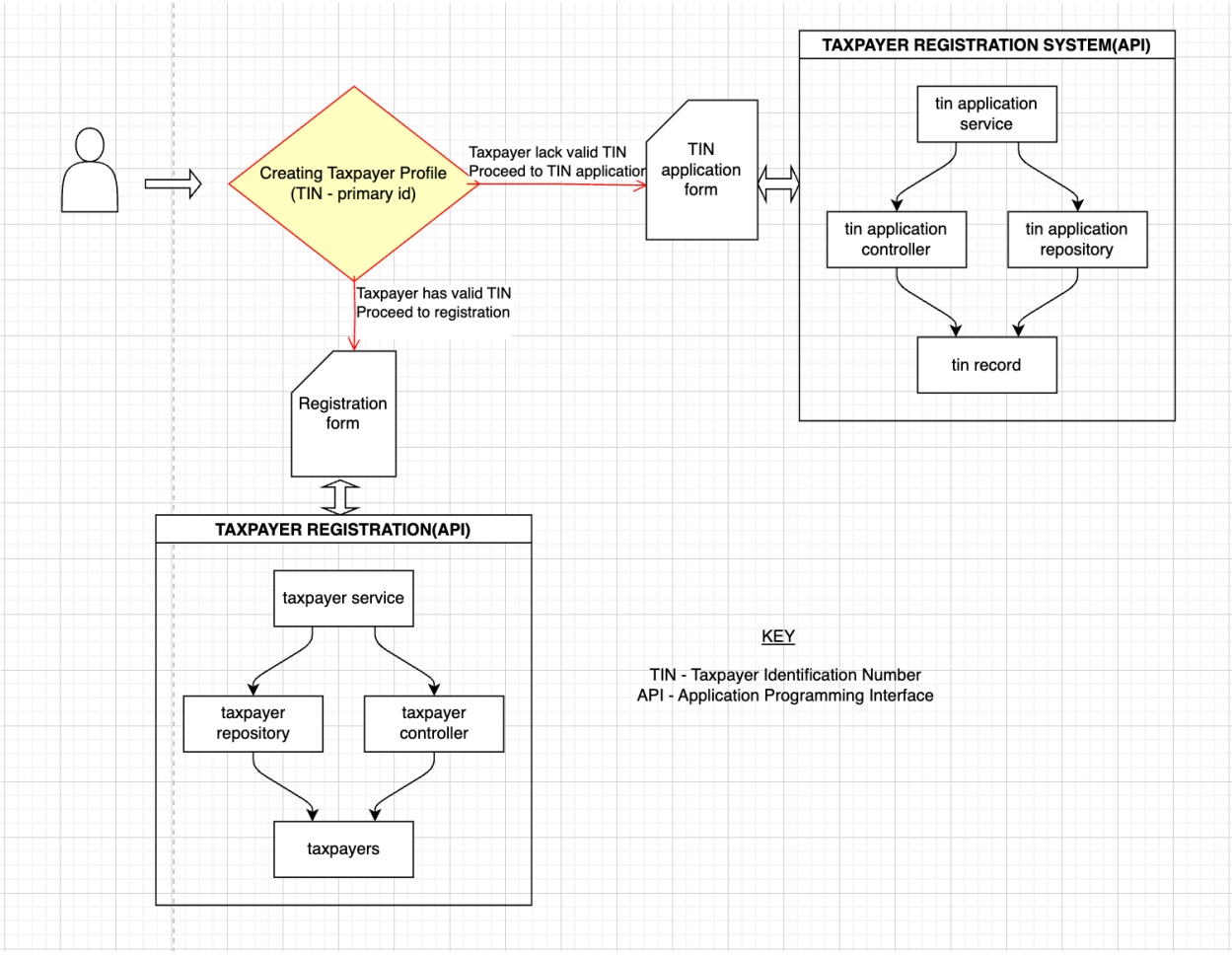
Utilizes Spring Data JPA for ORM (Object-Relational Mapping).
Implements repository interfaces for each entity (e.g., TaxpayerRepository, TINApplicationRepository).
Handles database transactions and ensures data integrity.

3. Database (MySQL)
    - Stores all taxpayer information, TIN records, and related data

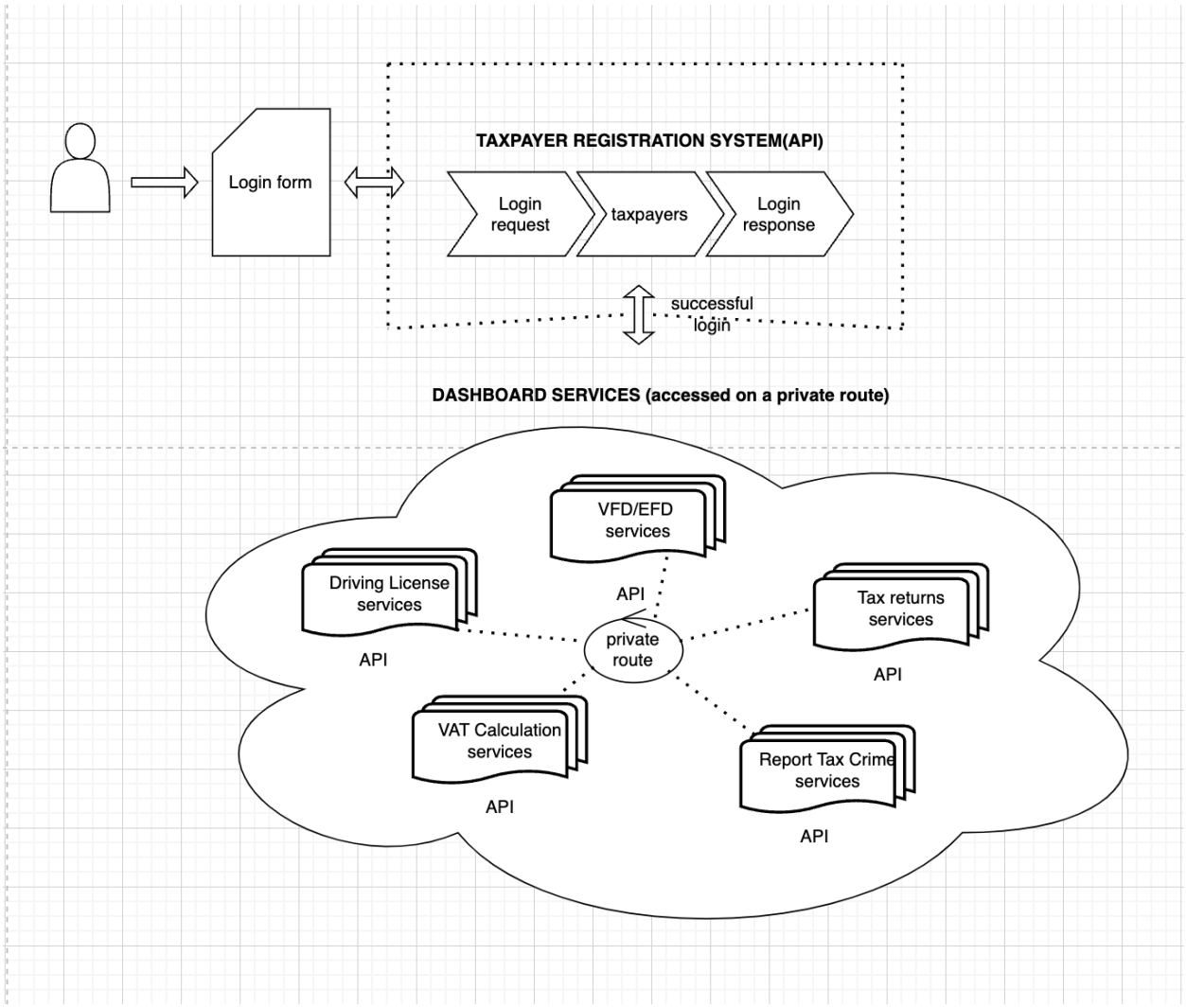
Stores all taxpayer information, TIN records, and related data.

- taxpayers table: Stores comprehensive taxpayer details including personal and financial information
- tin_applications table: Tracks TIN application status and details
- tin_record table: Links TINs to their corresponding applications and taxpayers
- driving_licenses table: Stores driving license information related to taxpayers
- tax_returns table: Keeps record of submitted tax returns
- vat_calculation table: Stores VAT calculation details
- fiscal_receipts table: Manages fiscal receipt information
- reported_tax_crimes table: Stores information about reported tax crimes

Implements proper indexing for optimized query performance.
Utilizes foreign key constraints to maintain referential integrity between tables.
Implements appropriate data types and constraints to ensure data validity.



Taxpayer registration



Taxpayer login and service access

5. SETUP AND INSTALLATION

1. Backend Setup (Spring Boot)
    - Ensure Java 17 is installed on your system (as shown in Image 5, the application is using Java 17.0.11)
    - Clone the repository containing the Spring Boot project
    - Navigate to the backend directory
    - Run `./gradlew build` to download dependencies and build the project
    - Configure the `application.properties` file with your database credentials
2. Frontend Setup (React)
    - Ensure Node.js is installed (version 20.12.2 as mentioned earlier)
    - Navigate to the frontend directory
    - Run `npm install` to install dependencies
    - Configure the API endpoint in the React app to point to your backend server
3. Database Setup (MySQL)
    - Install MySQL Community Server 8.0.26
    - Create a new database named `trs_db` (as shown in your MySQL output)
    - Run the SQL scripts to create the necessary tables (taxpayers, tin_applications, tin_record, etc.)
    - Ensure the database user has appropriate permissions

Running the Application

1. Backend Startup
    - Navigate to the backend directory
    - Run `./gradlew bootRun` to start the Spring Boot application
    - You should see output similar to Image 5, indicating the application has started successfully
2. Frontend Startup
    - Navigate to the frontend directory
    - Run `npm start` to start the React development server
    - You should see output similar to the right side of Image 5, showing compilation success and the local URL
3. Accessing the Application
    - Open a web browser and navigate to `http://localhost:3000`
    - You should see the Tanzania Revenue Authority Taxpayer Portal homepage (Image 3)
    - Users can log in, create a taxpayer account, or apply for a TIN
    - After successful login, users will see the dashboard with various services (Image 4)

Testing

1. Backend Unit Tests
    - Navigate to the backend directory
    - Run `./gradlew test` to execute the unit tests

    - Ensure all tests pass successfully
2. Frontend Unit Tests (if implemented)
    - Navigate to the frontend directory
    - Run `npm test` to execute React component tests
    - Ensure all tests pass successfully
3. Integration Testing
    - Test the full flow from TIN application to taxpayer registration to services.
    - Verify data persistence in the MySQL database.
    - Test various scenarios (e.g., valid TIN, invalid TIN)

Additional Notes:

- The system follows the flow depicted in Images 1 and 2, starting with TIN application/verification and proceeding to registration
- The dashboard (Image 4) provides access to various services like Tax Returns, Driving License, VFD/EFD, VAT calculation, and Tax Crime Reporting
- Ensure proper error handling and user feedback throughout the application

6. FUTURE IMPROVEMENTS

Known Issues:

1. Duplicacy of TIN and user information:
    - This issue has been largely addressed through database normalization, as evident in the current table structure.
    - The `tin_record` table serves as an intermediary between `taxpayers` and `tin_applications`, ensuring unique TIN assignments.
    - Unique constraints on the `tin` field in relevant tables further prevent duplicates.
2. User notification system:
    - Currently, the system fails to notify users about their TIN application status, generated TIN, or profile registration completion.
    - Root cause: Lack of access to required APIs for email and SMS services.

Planned Features:

1. Implement robust user notification system:
    - Priority: High
    - Description: Integrate with reliable email and SMS APIs to enable automated notifications for:
        - TIN application status updates
        - New TIN generation
        - Successful profile registration
        - Important tax deadlines and reminders
    - Expected impact: Significantly improve user experience and engagement with the system.
2. Enhance data validation and integrity:
    - Priority: Medium
    - Description: Implement additional server-side validations to further prevent data inconsistencies, especially for complex fields like `estimated_annual_turnover` and `main_economic_activity`.
3. Implement advanced analytics dashboard:
    - Priority: Low
    - Description: Develop a comprehensive analytics module for administrators to gain insights into taxpayer demographics, application trends, and system usage patterns.
4. Mobile application development:
    - Priority: Medium
    - Description: Create a mobile app version of the taxpayer portal for improved accessibility and user convenience.

Notes:

- This implementation is currently a demo version, serving as a proof of concept for the Taxpayer Registration System.

- Future development will focus on scalability, security enhancements, and integration with broader tax management ecosystems.
- Continuous feedback from users and stakeholders will be crucial in prioritizing and refining these planned improvements.