Entity-relationship diagram :

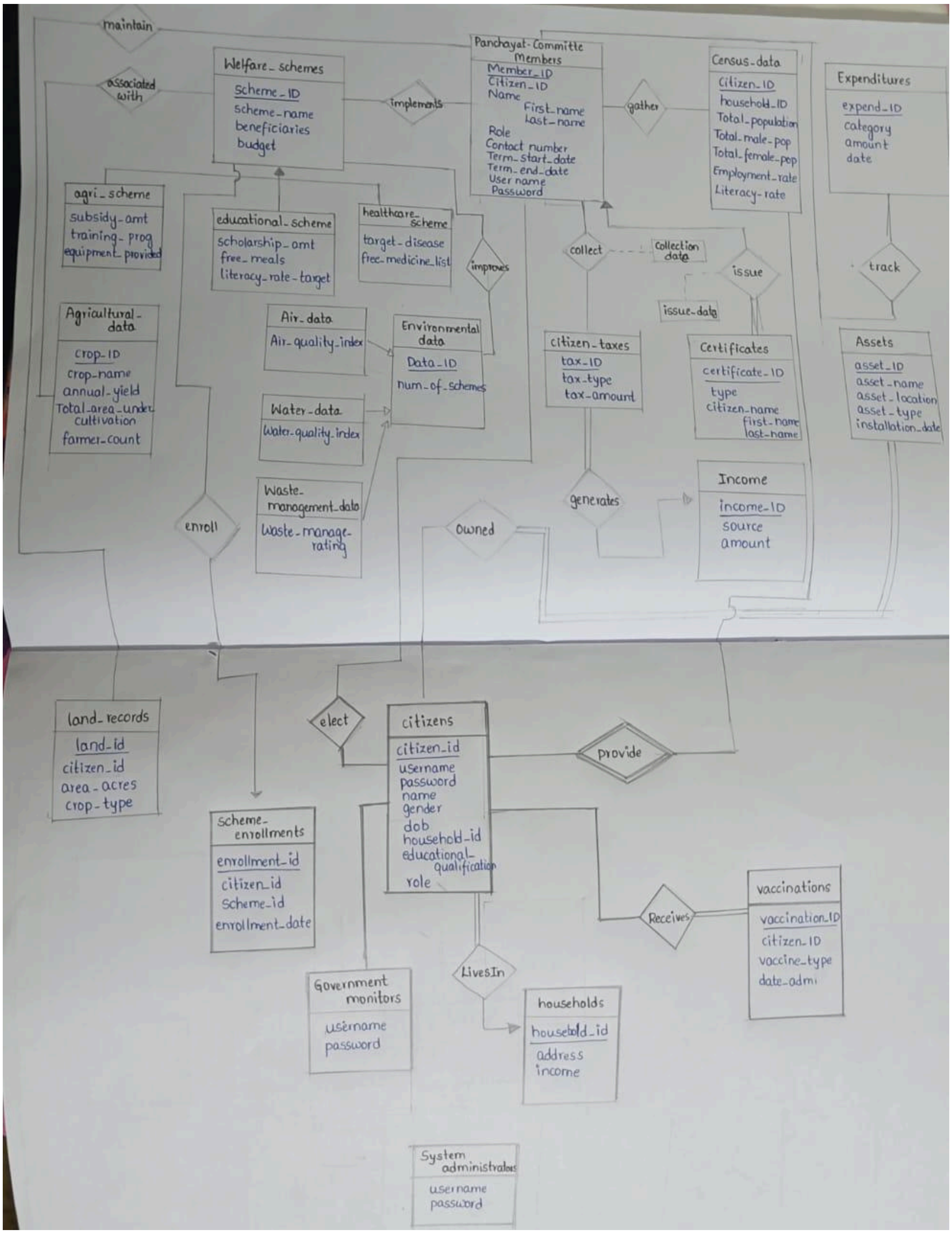


Table schema :

```sql
CREATE TABLE households (
   household_id INT PRIMARY KEY,
   address TEXT,
   income DECIMAL(10,2)
);

CREATE TABLE citizens (
   citizen_id INT PRIMARY KEY,
   username VARCHAR(255) UNIQUE NOT NULL,
   password VARCHAR(255) NOT NULL,
   name VARCHAR(255) NOT NULL,
   gender VARCHAR(10),
   dob DATE,
   household_id INT,
   educational_qualification VARCHAR(255),
   role VARCHAR(100),
   FOREIGN KEY (household_id) REFERENCES households(household_id)
);


CREATE TABLE land_records (
   land_id INT PRIMARY KEY,
   citizen_id INT,
   area_acres DECIMAL(10,2),
   crop_type VARCHAR(255),
   FOREIGN KEY (citizen_id) REFERENCES citizens(citizen_id)
);
CREATE TABLE welfare_schemes (
   scheme_id INT PRIMARY KEY,
   scheme_name VARCHAR(255),
   beneficiaries TEXT,
   budget DECIMAL(15,2)
);

CREATE TABLE scheme_enrollments (
   enrollment_id INT PRIMARY KEY,
   citizen_id INT,
   scheme_id INT,
   enrollment_date DATE,
   FOREIGN KEY (citizen_id) REFERENCES citizens(citizen_id),
   FOREIGN KEY (scheme_id) REFERENCES welfare_schemes(scheme_id)
);



CREATE TABLE educational_schemes (
   scheme_id INT PRIMARY KEY,
   scholarship_amount DECIMAL(10,2),
   free_meals BOOLEAN,
   literacy_role_target VARCHAR(255),
   FOREIGN KEY (scheme_id) REFERENCES welfare_schemes(scheme_id)
);

CREATE TABLE agri_schemes (
   scheme_id INT PRIMARY KEY,
   subsidy_amount DECIMAL(10,2),
   training TEXT,
   prog_equipment_provided TEXT,
   FOREIGN KEY (scheme_id) REFERENCES welfare_schemes(scheme_id)
);

CREATE TABLE healthcare_schemes (
   scheme_id INT PRIMARY KEY,
   target_disease VARCHAR(255),
   free_medicine_list TEXT,
   FOREIGN KEY (scheme_id) REFERENCES welfare_schemes(scheme_id)
);

CREATE TABLE panchayat_committee_members (
   member_id INT PRIMARY KEY,
   citizen_id INT,
   first_name VARCHAR(255),
   last_name VARCHAR(255),
   role VARCHAR(255),
   contact_number VARCHAR(15),
   term_start_date DATE,
   term_end_date DATE,
   username VARCHAR(255) UNIQUE NOT NULL,
   password VARCHAR(255) NOT NULL,
   FOREIGN KEY (citizen_id) REFERENCES citizens(citizen_id)
);
```

```sql
CREATE TABLE citizen_taxes (
   tax_id INT PRIMARY KEY,
   citizen_id INT,
   tax_type VARCHAR(255),
   tax_amount DECIMAL(15,2),
   collection_date DATE,
   FOREIGN KEY (citizen_id) REFERENCES citizens(citizen_id)
);

CREATE TABLE census_data (
   census_id INT PRIMARY KEY,
   citizen_id INT,
   household_id INT,
   total_population INT,
   total_male_population INT,
   total_female_population INT,
   employment_rate DECIMAL(5,2),
   literacy_rate DECIMAL(5,2),
   FOREIGN KEY (citizen_id) REFERENCES citizens(citizen_id),
   FOREIGN KEY (household_id) REFERENCES households(household_id)
);

CREATE TABLE environmental_data (
   data_id INT PRIMARY KEY,
   num_of_schemes INT
);

CREATE TABLE air_data (
   data_id INT PRIMARY KEY,
   air_quality_index INT,
   FOREIGN KEY (data_id) REFERENCES environmental_data(data_id)
);

CREATE TABLE water_data (
   data_id INT PRIMARY KEY,
   water_quality_index INT,
   FOREIGN KEY (data_id) REFERENCES environmental_data(data_id)
);

CREATE TABLE waste_management_data (
   data_id INT PRIMARY KEY,
   waste_management_rating INT,
    FOREIGN KEY (data_id) REFERENCES environmental_data(data_id)
);

CREATE TABLE income (
    income_id INT PRIMARY KEY,
    source VARCHAR(255),
    amount DECIMAL(15,2)
);

CREATE TABLE expenditures (
    expend_id INT PRIMARY KEY,
    category VARCHAR(255),
    amount DECIMAL(15,2),
    date DATE
);

CREATE TABLE assets (
    asset_id INT PRIMARY KEY,
    asset_name VARCHAR(255),
    asset_type VARCHAR(255),
    installation_date DATE
);

CREATE TABLE certificates (
    certificate_id INT PRIMARY KEY,
    citizen_id INT,
    type VARCHAR(255),
    issue_date DATE,
    FOREIGN KEY (citizen_id) REFERENCES citizens(citizen_id)
);

CREATE TABLE vaccinations (
    vaccination_id INT PRIMARY KEY,
    citizen_id INT,
    vaccine_type VARCHAR(255),
    date_administered DATE,
    FOREIGN KEY (citizen_id) REFERENCES citizens(citizen_id)
);

CREATE TABLE government_monitors (
    username VARCHAR(255) PRIMARY KEY,
    password VARCHAR(255) NOT NULL
);
```

```sql
CREATE TABLE system_administrators (
   username VARCHAR(255) PRIMARY KEY,
   password VARCHAR(255) NOT NULL
);

CREATE TABLE citizen_services (
   service_id INT PRIMARY KEY,
   citizen_id INT,
   service_type VARCHAR(255),
   FOREIGN KEY (citizen_id) REFERENCES citizens(citizen_id)
);

CREATE TABLE tax_collections (
   collection_id INT PRIMARY KEY,
   monitor_username VARCHAR(255),
   tax_id INT,
   collection_date DATE,
   FOREIGN KEY (monitor_username) REFERENCES
government_monitors(username),
   FOREIGN KEY (tax_id) REFERENCES citizen_taxes(tax_id)
);

CREATE TABLE scheme_maintenance (
   maintenance_id INT PRIMARY KEY,
   scheme_id INT,
   maintained_by INT,
   FOREIGN KEY (scheme_id) REFERENCES welfare_schemes(scheme_id),
   FOREIGN KEY (maintained_by) REFERENCES
panchayat_committee_members(member_id)
);
```

Functionalities implemented:

1. Introduction

This report provides an in-depth overview of the functionalities implemented in the system. The system is designed to cater to multiple user roles, including citizens, Panchayat officials, government monitors, and administrators. It provides essential services such as user authentication, taxation, land records management, scheme enrollment, financial management, health services, and government monitoring.

---

2. User Authentication System

To ensure secure access and role-based authentication, the following login and registration functionalities have been implemented:

2.1 Login System

- **General Login (`Login.js`)** – Provides a unified login interface for different user roles.
- **Citizen Login (`CitizenLogin.js`)** – Allows citizens to log in and access their personalized services.
- **Panchayat Official Login (`PanchayatLogin.js`)** – Enables Panchayat officials to log in and manage their respective administrative tasks.
- **Government Monitor Login (`GovernmentMonitorLogin.js`)** – Grants access to government officials responsible for monitoring Panchayat activities.
- **Admin Login (`AdminLogin.js`)** – Provides system administrators with full control over the platform.

2.2 Registration System

- **Citizen Registration (`CitizenRegister.js`)** – Enables citizens to sign up and access government services.
- **Panchayat Official Registration (`PanchayatRegister.js`)** – Allows Panchayat officials to register and manage Panchayat-related data.

- **Government Monitor Registration (`GovernmentMonitorRegister.js`)** – Facilitates government monitors in signing up for their roles.
- **Admin Registration (`AdminRegister.js`)** – Grants access to system administrators.
- **General Registration Page (`RegisterPage.js`)** – Serves as a universal registration interface for all users.

---

3. Citizen Services

Citizens have access to multiple services through the platform to facilitate interactions with the government.

3.1 Profile Management

- **Citizen Profile (`CitizenProfile.js`)** – Allows users to view, update, and manage their personal details, including service history and applied schemes.

3.2 Financial and Land Services

- **Tax Management (`CitizenTaxes.js`)** – Provides an interface for citizens to view their taxes online securely.
- **Land Records (`CitizenLandRecords.js`)** – Enables citizens to check ownership details, property boundaries, and other land-related information.

3.3 Government Schemes

- **Scheme Information (`CitizenSchemePage.js`)** – Allows citizens to explore various government schemes, view eligibility criteria, and apply.

3.4 Health Services

- **Vaccination Records (`CitizenVaccination.js`)** – Enables citizens to check their vaccination status, upcoming immunization schedules, and download certificates.

---

4. Panchayat Services

Panchayat officials have access to various administrative functionalities to manage governance at the local level.

4.1 Profile Management

- **Panchayat Profile (`PanchayatProfile.js`)** – Allows officials to update Panchayat details, jurisdiction, and office information.

4.2 Financial Management

- **Expenditure Tracking (`PanchayatExpenditure.js`)** – Records and monitors Panchayat spending on development and welfare programs.
- **Income Management (`PanchayatIncome.js`)** – Tracks revenue sources, including tax collections, grants, and government funds.
- **Tax Collection (`PanchayatTaxes.js`)** – Facilitates tax collection processes and provides detailed reports.

4.3 Resource Management

- **Asset Management (`PanchayatAsset.js`)** – Maintains a database of Panchayat-owned properties, vehicles, and resources.
- **Census Data (`PanchayatCensusData.js`)** – Stores demographic data essential for governance and policy-making.
- **Agricultural Data (`PanchayatAgriData.js`)** – Maintains records on crops, irrigation facilities, and productivity trends.
- **Environmental Data (`PanchayatEnviData.js`)** – Monitors pollution levels, water quality, and ecological concerns.

4.4 Government Schemes

- **Scheme Management (`PanchayatSchemePage.js`)** – Allows Panchayat officials to oversee scheme details, update policies, and track beneficiaries.
- **Scheme Enrollment (`PanchayatSchemeEnrollment.js`)** – Enables officials to process citizen applications for various welfare schemes and subsidies.

4.5 Health Administration

- **Vaccination Program (`PanchayatVaccinations.js`)** – Manages vaccination drives, tracks immunization records, and ensures community health compliance.

---

5. Government Monitoring

Government monitors oversee Panchayat activities to ensure compliance with national and state regulations.

5.1 Access and Authentication

- **Government Monitor Login (`GovernmentMonitorLogin.js`)** – Provides secure login for officials monitoring Panchayat affairs.
- **Government Monitor Registration (`GovernmentMonitorRegister.js`)** – Enables government monitors to create accounts and manage their responsibilities.

5.2 Oversight and Reporting

- **Audit and Compliance** – Ensures that Panchayat activities align with government policies and regulations.
- **Data Verification** – Cross-checks census data, expenditure reports, and resource allocations.

---

6. General Functionalities

6.1 Home Page

- **Landing Page (`Home.js`)** – Serves as the primary interface, directing users to their respective dashboards and functionalities.

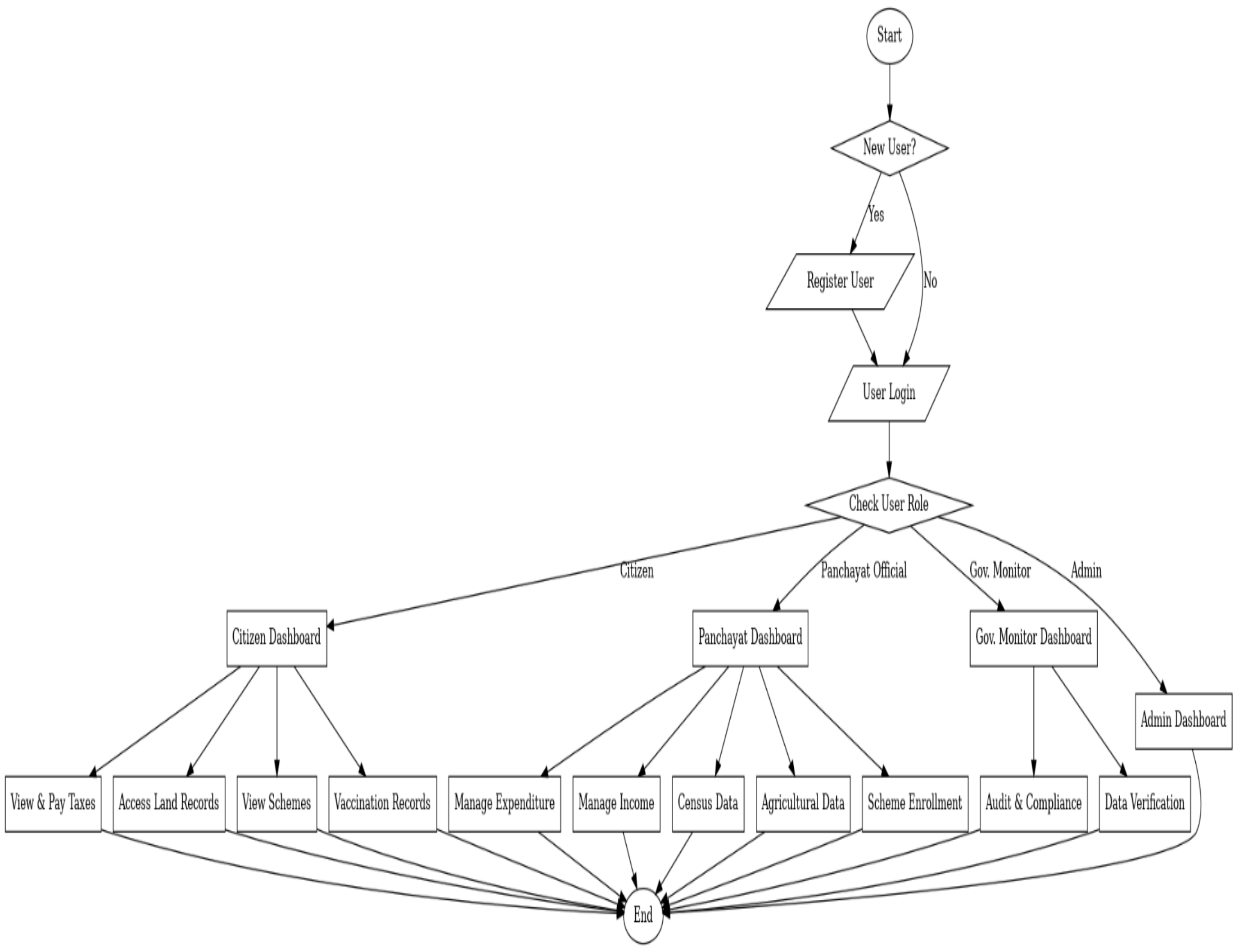
6.2 User Interface Features

- **Navigation System** – Provides an intuitive UI for easy access to various features.
- **Role-Based Dashboards** – Custom views for citizens, Panchayat officials, and government monitors based on their privileges.

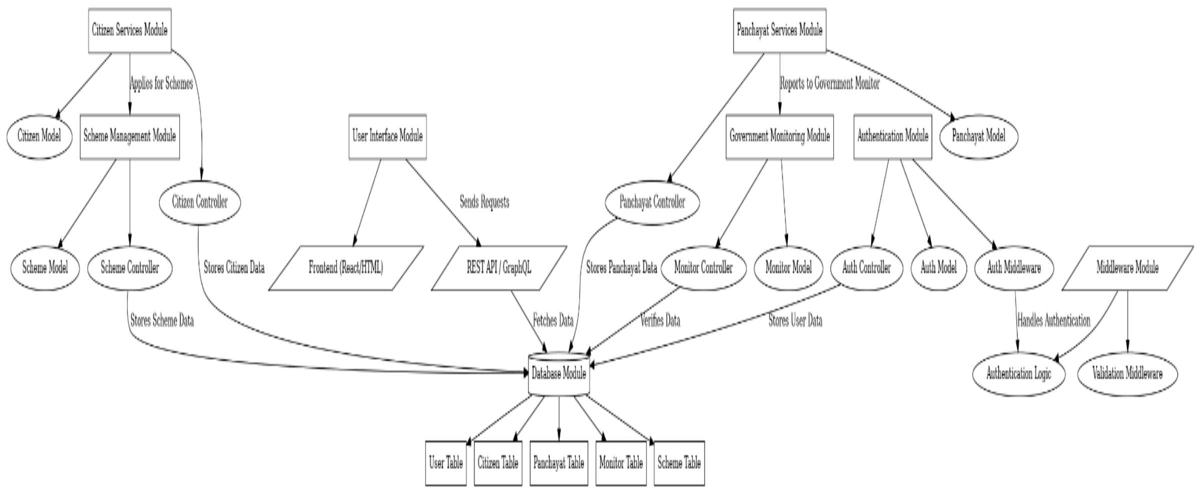
7. Conclusion

The system integrates multiple functionalities to create a seamless user experience for different stakeholders. The authentication system ensures security, while the various modules cater to the financial, administrative, and welfare needs of citizens and officials. The subdivision of functionalities ensures detailed access to services, enabling efficient governance, compliance, and public service delivery. This report provides a comprehensive breakdown of the system's components, ensuring clarity in its scope and capabilities.

WORKFLOW DIAGRAM:



MODULE DIAGRAM:



TOOLS USED:

Tools & Technologies Used in the System

Based on the extracted project files and system structure, the following tools and technologies have been used:

1. Backend Development

- **Node.js & Express.js** – Used for building the server-side logic and API endpoints.
- **JavaScript (ES6+)** – Core language for backend development.
- **SQL (PostgreSQL)** – Database management using SQL schema files.
- **JWT (JSON Web Tokens)** – Used for authentication and session management.

2. Frontend Development

- **React.js** – Used for building the user interface.
- **CSS3** – Standard web technologies for structuring and styling.
- **Fetch** – For making API requests to the backend.

3. Authentication & Security

- **bcrypt.js** – Used for password hashing and authentication security.
- **jsonwebtoken (JWT)** – For secure user authentication.
- **Express Middleware** – Used for request validation and authorization.

4. Database Management

- **PostgreSQL** – Stores user, citizen, Panchayat, and government data.
- **SQL Scripts** – Schema creation and data migration handled via SQL files.

5. API Development

- **RESTful APIs** – Backend exposes REST APIs for handling requests.
- **Postman (Testing)** – Used for API testing and debugging.

=============================END=================================