Government Grievance Portal: Project Documentation
Module 3: System Architecture & Technology

Mankaran Singh Tandon (UID: 23BCS10204)

Section: KRG2B
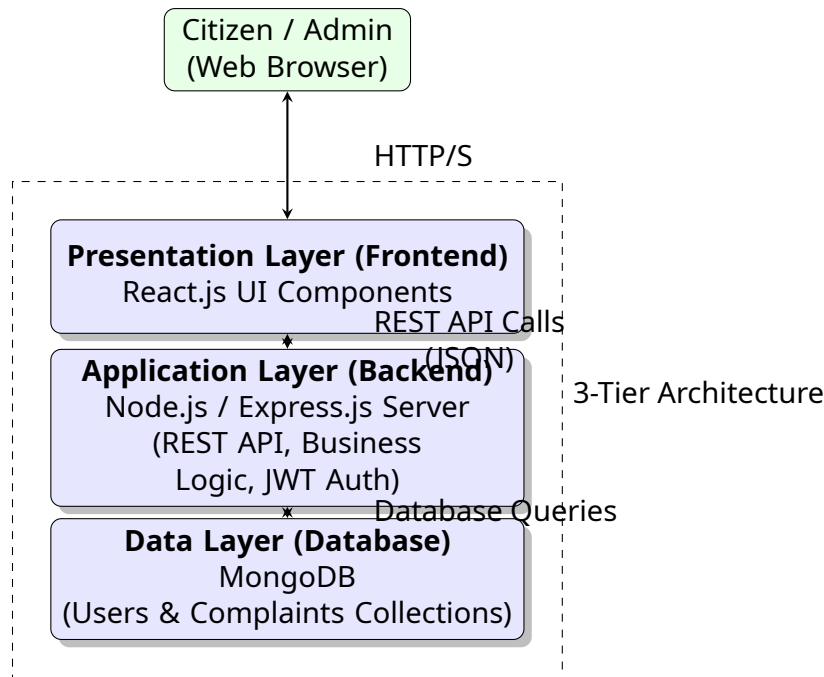Course: Full Stack Development (PBLJ / Web Technologies)

5. Technology Stack

The portal is built using a modern, full-stack JavaScript (MERN) stack, chosen for its performance, scalability, and robust ecosystem.
- **Frontend (Presentation Layer): React.js**
  - Used for building the dynamic, responsive, and interactive user interface (UI).
  - Leverages components, state management, and routing to create a seamless single-page application (SPA) experience for citizens and admins.
  - Complemented by standard **HTML5**, **CSS3**, and **JavaScript (ES6+)**.
- **Backend (Application Layer): Node.js** with **Express.js**
  - **Node.js** provides the runtime environment for executing JavaScript on the server.
  - **Express.js** is a minimal and flexible web application framework that provides a robust set of features for building the RESTful APIs used for communication between the frontend and database.
- **Database (Data Layer): MongoDB**
  - A NoSQL, document-oriented database used for its flexibility in storing schema-less data (like grievances and user profiles).
  - Its scalability and performance make it ideal for handling large volumes of user-generated content.
- **Authentication: JSON Web Tokens (JWT)**
  - Used for securely transmitting information between parties as a JSON object, enabling stateless authentication for managing user sessions and securing API endpoints.
- **Version Control: Git & GitHub**
  - Git is used for local version control, and GitHub is used for remote repository hosting, collaboration, and code management.
- **Development Tools:**
  - **VS Code:** Primary code editor.
  - **Postman:** Used for designing, testing, and documenting the RESTful APIs.
  - **MongoDB Compass:** GUI for querying, visualizing, and managing the MongoDB database.

6. System Architecture

The application follows a 3-Tier Architecture, which logically separates the application into three distinct computing layers. This separation of concerns enhances scalability, maintainability, and flexibility.

```
        ┌─────────────────────┐
        │   Citizen / Admin   │
        │   (Web Browser)     │
        └─────────────────────┘
                  ↕ HTTP/S
  ┌ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ┐
      ┌─────────────────────────┐
  │   │ Presentation Layer      │     │
      │      (Frontend)         │
  │   │ React.js UI Components  │     │
      └─────────────────────────┘
  │         ↕ REST API Calls          │
              (JSON)
  │   ┌─────────────────────────┐     │ 3-Tier Architecture
      │ Application Layer        │
  │   │    (Backend)             │     │
      │ Node.js / Express.js     │
  │   │ Server                   │     │
      │ (REST API, Business      │
  │   │ Logic, JWT Auth)         │     │
      └─────────────────────────┘
  │         ↕ Database Queries        │
      ┌─────────────────────────┐
  │   │ Data Layer (Database)   │     │
      │ MongoDB                 │
  │   │ (Users & Complaints     │     │
      │ Collections)            │
  │   └─────────────────────────┘     │
  └ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ┘
```

- **Presentation Tier (Frontend):** This is the user interface, managed entirely by React.js. It is what the user (citizen, admin) sees and interacts with in their web browser. It is responsible for rendering data and sending user requests to the backend.
- **Application Tier (Backend):** This is the "brain" of the application, powered by Node.js and Express.js. It handles all business logic, processes requests from the frontend, enforces security and authentication (JWT), and communicates with the database to fetch or store data.
- **Data Tier (Database):** This is where all persistent data is stored. MongoDB holds the collections for users, complaints, departments, etc. This layer is only accessible via the Application Tier, never directly by the frontend, ensuring data security.