**PGatEase Technical Architecture and Implementation Documentation**

**1. Introduction**

PGatEase is a digital platform designed to streamline property management for PG (Paying Guest) owners and enhance the tenant experience. It enables PG owners to manage complaints, payments, announcements, customizable food preferences, and laundry services. Tenants benefit from a transparent and efficient system where they can track their complaints, payments, and access customizable services.

**2. Tech Stack**

The chosen tech stack is designed for flexibility, scalability, and efficient performance, ensuring that PGatEase can scale as the business grows. The platform is built with modern frameworks and tools that allow for smooth integration of services and the ability to handle large-scale data and traffic.

**Frontend:**

* **Next.js (React.js)**: For building the user interface, using **Next.js** provides benefits such as server-side rendering (SSR), static site generation (SSG), and a flexible routing system (using App Router).
* **Tailwind CSS**: A utility-first CSS framework that facilitates the quick and responsive design of the platform's user interface.

**Backend:**

* **Node.js with Express.js**: The backend is built with **Node.js** and **Express.js**. Node.js ensures that the platform is fast and scalable, while Express.js simplifies building RESTful APIs.
* **JWT (JSON Web Tokens)**: For secure user authentication, JWTs are used to ensure session security and prevent unauthorized access.
* **REST API**: The platform uses a **RESTful API** architecture, enabling seamless communication between the frontend and backend.

**Database:**

* **MySQL / PostgreSQL**: Both **MySQL** and **PostgreSQL** are supported for database management, providing flexibility based on user preferences and specific requirements.
  + **MySQL**: An open-source relational database management system widely known for its simplicity, reliability, and performance.
  + **PostgreSQL**: An advanced open-source relational database that offers strong consistency, support for complex queries, and robust performance for enterprise-level applications.

**Database Management**:

* + **Sequelize ORM**: Sequelize is used to interact with both MySQL and PostgreSQL databases in an easy and unified way. The ORM helps in managing database queries and migrations without writing raw SQL.
  + **Django ORM (optional)**: If Django is chosen, the built-in ORM will be used to interact with the PostgreSQL database.

**Hosting and Deployment:**

* **Heroku**: PGatEase will be hosted on **Heroku**, providing automatic scaling and integrated management of both MySQL and PostgreSQL databases through add-ons.
* **Docker**: The platform will be containerized using Docker for consistent deployment across environments, ensuring that the application behaves the same in development, staging, and production.

**Third-Party Integrations:**

* **Twilio / SendGrid**: These services will be used for SMS and email notifications, including rent reminders, complaint updates, and laundry service notifications.
* **Stripe / Razorpay (Future Integration)**: A payment gateway for future integration to process online rent payments.

**3. Features and Functionality**

PGatEase is designed to provide essential features for both PG owners and tenants, leveraging the tech stack for smooth interactions and scalability.

**3.1 PG Owner Features:**

* **Owner Dashboard**: Built with Next.js, this dashboard allows PG owners to view and manage multiple properties, complaints, payments, food preferences, and laundry requests.
* **Customizable Food Service**: PG owners can upload weekly or monthly meal plans, and tenants can customize their meal preferences. This is stored in the database, which can be either MySQL or PostgreSQL.
* **Laundry Service Management**: PG owners can manage tenant laundry requests, scheduling pickups, deliveries, and processing.

**3.2 Tenant Features:**

* **Tenant Portal**: A responsive portal where tenants can view their complaints, track payments, and request services such as food and laundry.
* **Food Preferences**: Tenants can choose meal plans or customize their food choices based on availability. Data is stored in the database and updated accordingly.
* **Laundry Service Requests**: Tenants can submit requests for laundry services, track their orders, and receive notifications when their laundry is ready for pickup.

**4. Database Design**

PGatEase supports both **MySQL** and **PostgreSQL**, allowing for flexibility in database selection. Below are the key tables and data flow considerations for both databases:

**Tables:**

* **Users**: Stores tenant and PG owner data (name, contact info, role, etc.).
* **Complaints**: Stores complaints raised by tenants, including status, type, and timestamps.
* **Payments**: Tracks rent payments made by tenants, including payment status and due dates.
* **FoodPreferences**: Stores customizable food preferences for tenants.
* **LaundryRequests**: Stores laundry requests made by tenants, including pickup and delivery statuses.

**Data Flow:**

1. **Complaint Management**:
   * Tenants submit complaints through the portal.
   * Data is stored in the **Complaints** table (relational design).
   * PG owners can view, resolve, and track complaints.
2. **Payment Tracking**:
   * Rent payments are logged in the **Payments** table and accessible by both PG owners and tenants.
   * Future integration with payment gateways (e.g., Stripe, Razorpay) will allow tenants to pay rent online.
3. **Food and Laundry Customization**:
   * Tenants’ food preferences are stored in the **FoodPreferences** table, allowing PG owners to customize meal plans.
   * Laundry requests are logged in the **LaundryRequests** table for scheduling and status tracking.

**5. Security and Authentication**

PGatEase ensures that user data is protected and only accessible to authorized users through the following security measures:

* **JWT Authentication**: Users are authenticated via **JWT tokens**, which provide secure sessions.
* **Role-Based Access Control**: Users have different access levels based on their roles (PG owners or tenants), ensuring that only authorized users can access sensitive data (e.g., complaints, payments).
* **Data Encryption**: All sensitive data (e.g., payment details, tenant information) is encrypted using industry-standard encryption techniques, ensuring privacy and security.

**6. Scalability and Performance**

The platform is designed for scalability and optimal performance:

* **Database Optimization**: Both MySQL and PostgreSQL are optimized for handling large datasets. PostgreSQL supports advanced indexing and complex queries, making it a good choice for large-scale data. MySQL is efficient for simpler queries and fast read/write operations.
* **Docker**: The use of Docker allows easy scaling by creating multiple containers as demand increases, ensuring consistency across environments.
* **Heroku Hosting**: With Heroku’s scaling capabilities, PGatEase can scale resources based on user demand without manual intervention.

**7. Future Enhancements**

* **Payment Gateway Integration**: Integration of online payment systems (e.g., Stripe, Razorpay) for processing rent payments will be added in future versions.
* **Mobile Application**: A mobile app will be developed to provide tenants and PG owners with access to services on-the-go, ensuring convenience and flexibility.
* **Advanced Analytics**: PG owners will gain insights into tenant preferences, complaint resolution times, and payment history to optimize property management.