Notifications

This project involves creating a Customer Notification Address Facade System, a microservice to centralize and manage customer contact information and preferences for notifications. This system will act as a single source of truth for all recipient addresses and delivery statuses, helping other systems in the ecosystem to fetch and update customer delivery data efficiently.

Key Features:

User Management:

- Create and manage admin accounts.
- Allow admins to log in securely.

Customer Management:

- Add new customers with details like name, contact information (email, phone number, etc.), and notification preferences.
- Edit and update customer information.
- Remove customers from the system.
- Display a list of all customers with their notification preferences.

Address Management:

- Store and manage different types of customer addresses (e.g., email, SMS, postal).
- Allow updates to customer addresses.
- Remove outdated or incorrect addresses.
- Display a list of all addresses associated with each customer.

Preference Management:

- Record and manage customer preferences for receiving notifications (e.g., opt-in/opt-out for SMS, email, promotional messages).
- Update notification preferences based on customer requests.
- Display current notification preferences for each customer.

Integration and API:

- Provide a RESTful API for other systems to access customer addresses and preferences.
- Ensure secure and authenticated access to the API.
- Allow batch updates for customer data.

Notification Tracking:

- Track the status of notifications sent to customers (e.g., delivered, failed, pending).
- Provide endpoints for querying notification statuses.
- Generate reports on notification delivery success rates.

Search and Filtering:

- Implement search functionality for customers based on criteria like name, contact information, and notification preferences.
- Allow sorting and filtering of customer lists.

Reporting:

- Generate reports on customer data, such as the number of customers opted-in for different notification types.
- Provide statistics on notification delivery and failures.

Technical Details:

User Interface:

- Design an easy-to-use web-based interface for admins.
- Ensure it is responsive and works well on different devices.

Security:

- Implement user authentication and access control for admins.
- Secure sensitive information, such as customer contact details and preferences.

Database:

- Use a relational database (e.g., MySQL, PostgreSQL) to store customer data.
- Design an appropriate database schema to handle customer details and notification preferences.

Performance:

- Optimize data retrieval and updates to ensure quick response times.
- Ensure the system can handle concurrent access by multiple users.

Scalability:

- Design the system to accommodate future growth in customer data and user load.

Technology Stack:

Backend: Java (with Spring Boot) **Frontend:** HTML, CSS, JavaScript **Database:** Relational (e.g., MySQL, PostgreSQL) **Security:** Spring Security **Database Access:** Hibernate **User Interface Templating:** Thymeleaf or similar

Project Deliverables:

- The project's source code.
- Documentation (README) explaining how to set up and run the application.
- Brief documentation describing the project's structure and design choices.
- Source code should be provided as a git repository.

Evaluation Criteria:

The project will be assessed based on:

- **Functionality:** Does the system meet all functional requirements?
- **User Interface:** Is the interface user-friendly and adaptable to different devices?
- **Code Quality:** Is the code well-organized, readable, and maintainable?
- **Security:** Is user authentication and data protection implemented correctly?
- **Database Design:** Does the database structure fit the project's needs?
- **Documentation:** Is there clear documentation for setting up and using the application?
- **Error Handling:** Does the application handle errors gracefully?
- **Performance:** How efficiently does the application perform its tasks?