Spaza Shop Registration and Monitoring System (SSRMS)

Software Overview

The SSRMS is a web-based system designed to regulate and support spaza shop operations in South Africa. It provides a digital platform for registering shops, monitoring compliance, reviewing services, and ensuring the sale of healthy food by approved suppliers. The platform includes roles for Government Officials, Shop Owners, Suppliers, and Customers - each with specific responsibilities and permissions.

System Objectives

- Regulate spaza shops by ensuring they are properly registered and approved.
- Improve food quality by linking shop owners to verified suppliers.
- Empower customers with the ability to report and review shop services and products.
- Support government initiatives to formalize spaza shops and promote local ownership.
- Detect and filter fake complaints using AI systems based on historical data and user reputation.

Key Features by User Role

1. Shop Owner Panel

- Register a shop (pending approval).
- Browse verified Suppliers and place orders.
- View and manage shop profile.
- View ratings and reviews submitted by customers.
- Access complaint notifications and resolve flagged issues.

2. Government Panel

- Approve or decline shop registrations based on compliance.
- Approve or reject supplier registrations.
- Monitor complaints, including images submitted by customers.
- Verify shops and suppliers.
- Access dashboards showing highly rated shops by region.
- View system analytics and generate reports.

3. Supplier Panel (Newly Added)

- Register as a Supplier (pending approval).
- Provide food and product catalog with prices.
- Accept and process orders from shop owners.
- Update product information and stock availability.
- Maintain a reputation score visible to shop owners.

4. Customer Panel

- Submit reviews, ratings, and complaints about shops.
- Upload pictures of spoiled or low-quality products.
- View a list of verified shops by area.
- View a ranking board of top-rated shops.
- Receive updates on complaints via email or SMS.

System Behavior and Relationships

- Customers are associated with Shops through reviews, ratings, and complaints.
- Shop Owners can only order from Verified Suppliers.
- Government Officials act as gatekeepers for approving both Shops and Suppliers.
- Complaints and Reviews are processed with AI filters to detect potential abuse.
- A Verification Status is displayed on Shop and Supplier profiles.

Backend Architecture (Layered Approach)

1. Presentation Layer

- Frontend (outside your scope) communicates via RESTful APIs.

2. Application Layer

- Handles business logic for shop registration, complaint filtering, supplier verification, order processing, etc.

3. Service Layer

- Implements services such as:
  - ShopService
  - ReviewService
  - ComplaintService
  - SupplierService
  - AuthService
  - NotificationService

4. API Layer

- Exposes endpoints such as:
  - POST /register-shop
  - POST /register-supplier
  - GET /verified-suppliers
  - GET /reviews/:shopId
  - POST /submit-complaint
  - GET /top-rated-shops

5. Data Access Layer

- Uses repositories or DAOs to interact with the database.

6. Database Layer

- Stores:
  - User information (ShopOwner, Customer, Supplier, Official)
  - Shops and Suppliers
  - Reviews, Ratings, Complaints
  - Verification status
  - Orders and Products

7. Security Layer

- JWT-based authentication for all users.
- Role-based authorization (Admin, Owner, Customer, Supplier).
- Input validation and XSS protection.
- Secure image upload handling (for complaints).
- Rate limiting and monitoring for abuse detection.

Technologies Suggested

- Backend Framework: Node.js / Spring Boot / Django
- Database: PostgreSQL or MongoDB
- Authentication: JWT with role-based access control
- Image Storage: Cloudinary / AWS S3
- AI Complaint Filtering: Python-based ML microservice
- API Documentation: Swagger or Postman

Deployment Considerations

- Host backend on AWS / Azure / Heroku

- Use CI/CD for deployment

- Enforce HTTPS and use environment variables for secrets

- Monitor uptime and logs using tools like Datadog or Sentry