Product Requirements & Specification Document

Project Name

AgroTrack - Farm Produce Logistics API

Description

AgroTrack is a Spring Boot REST API for managing and tracking farm produce logistics. The system enables real-time tracking of shipment status, inventory management, and delivery scheduling. It leverages JPA for ORM, MySQL for data persistence, Docker for containerization, Maven for build management, and JWT for API security.

---

1. Objectives

- Enable efficient tracking of farm produce shipments.
- Provide real-time inventory and delivery scheduling.
- Ensure secure, scalable, and maintainable API architecture.

---

2. Stakeholders

Role	Responsibility
Product Owner	Requirements, prioritization
Developers	Implementation, testing
QA Engineers	Quality assurance
End Users	Logistics managers, farmers

---

3. Functional Requirements

ID	Requirement
FR1	CRUD operations for Produce, Shipment, Inventory, and Delivery entities
FR2	Track shipment status (e.g., pending, in transit, delivered)
FR3	Manage inventory levels per produce type
FR4	Schedule and update deliveries
FR5	Secure all endpoints with JWT authentication
FR6	Provide RESTful API endpoints with JSON payloads
FR7	Support filtering and searching for shipments and inventory

---

4. Non-Functional Requirements

ID	Requirement
NFR1	API response time < 500ms
NFR2	99.5% uptime
NFR3	Dockerized deployment
NFR4	MySQL as persistent storage
NFR5	Maven for build and dependency management
NFR6	Comprehensive API documentation (Swagger)

---

5. System Architecture

```
[Client] <--REST/JSON--> [Spring Boot API] <--JPA--> [MySQL DB]
                                  |
                              [JWT Auth]
                                  |
                              [Dockerized]
```

---

6. Entity Model

Entity	Attributes
Produce	id, name, type, description
Inventory	id, produce_id (FK), quantity, location
Shipment	id, produce_id (FK), status, origin, destination, departure, arrival
Delivery	id, shipment_id (FK), scheduled_time, actual_time, status
User	id, username, password, role

Relationships:

- Produce 1:N Inventory
- Produce 1:N Shipment
- Shipment 1:1 Delivery

---

7. API Endpoints (Sample)

Method	Endpoint	Description	Auth Required
POST	/api/auth/login	User authentication (JWT)	No
GET	/api/produce	List all produce	Yes
POST	/api/produce	Create new produce	Yes

GET	/api/inventory	List inventory	Yes
PUT	/api/inventory/{id}	Update inventory	Yes
GET	/api/shipments	List/filter shipments	Yes
POST	/api/shipments	Create shipment	Yes
PUT	/api/shipments/{id}/status	Update shipment status	Yes
GET	/api/deliveries	List deliveries	Yes
POST	/api/deliveries	Schedule delivery	Yes

---

8. Security

- All endpoints (except /auth/login) require JWT authentication.
- Role-based access control for sensitive operations.
- Passwords stored hashed (BCrypt).

---

9. Deployment

Component	Specification
Container	Docker
Database	MySQL 8.x
Build Tool	Maven
API Docs	Swagger/OpenAPI

Sample Docker Compose:

```yaml
version: '3.8'
services:
  api:
    build: .
    ports:
      - "8080:8080"
    environment:
      - SPRING_DATASOURCE_URL=jdbc:mysql://db:3306/agrotrack
      - SPRING_DATASOURCE_USERNAME=root
      - SPRING_DATASOURCE_PASSWORD=secret
    depends_on:
      - db
  db:
    image: mysql:8
    environment:
      MYSQL_ROOT_PASSWORD: secret
      MYSQL_DATABASE: agrotrack
    ports:
      - "3306:3306"
```

10. Acceptance Criteria

- All core entities and relationships implemented.
- JWT-secured endpoints functional.
- CRUD and tracking features operational.
- API documented and tested.
- Application runs via Docker Compose.

11. Out of Scope

- Mobile or web frontend
- Third-party logistics integrations
- Advanced analytics or reporting

12. Timeline & Milestones

Milestone	Target Date
Requirements Finalized	Week 1
Core API Implementation	Week 2-3
Security & Testing	Week 4
Dockerization & Docs	Week 5
UAT & Handover	Week 6

13. Glossary

Term	Definition
JWT	JSON Web Token, for API authentication
JPA	Java Persistence API, ORM for Java
REST	Representational State Transfer
CRUD	Create, Read, Update, Delete

End of Document