SwiftLogistics Project File Structure

Root Directory Structure

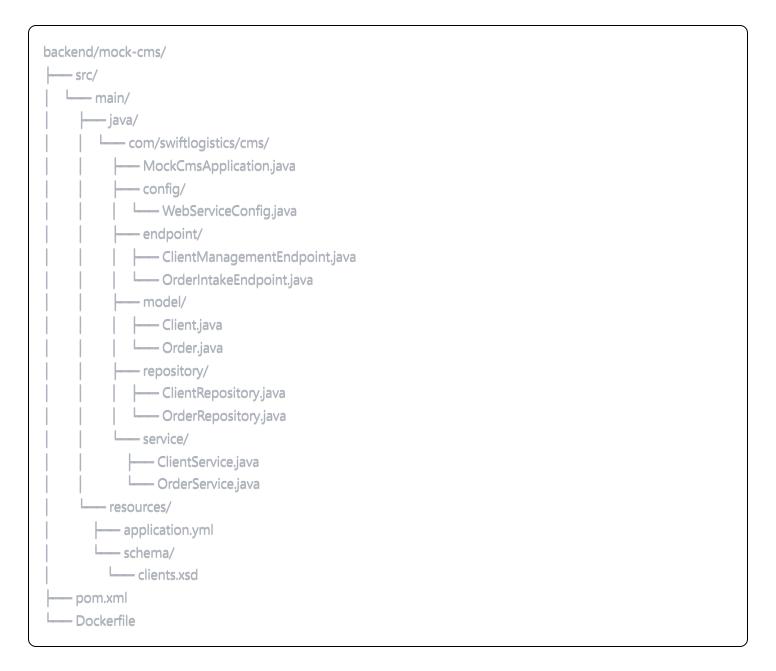
SwiftLogistics/	
README.md	
— docker-compose.yml	
gitignore	
— documentation/	
architecture-diagrams/	
api-specs/	
solution-report.pdf	
backend/	
api-gateway/	
— mock-cms/	
— mock-ros/	
order-service/	
user-service/	
L shared-libs/	
frontend/	
L client-portal/	
— mobile/	
driver-app/	
L—— infrastructure/	
—— message-broker/	
L— database/	
l	

Backend Structure (Spring Boot Microservices)

1. API Gateway



2. Mock CMS (SOAP Service)



3. Mock ROS (REST Service)



4. Mock WMS (TCP/IP Messaging)



5. Order Service



6. Tracking Service



7. User Service



Frontend Structure (React - Client Portal)



Mobile App Structure (Flutter - Driver App)

```
mobile/driver-app/
  --- lib/
      — main.dart
       - screens/
       login_screen.dart
        — delivery_list_screen.dart
        — delivery_details_screen.dart
         -- route_screen.dart
      – services/
       --- api_service.dart
        --- auth_service.dart
      — models/
        --- delivery.dart
      --- route.dart
      — widgets/
       — delivery_card.dart
      ____ status_update_form.dart
    - pubspec.yaml
     android/
```

Infrastructure Structure

What You Must Complete (Assignment Requirements Only)

1. Mock Systems 🔽

- **Mock CMS**: SOAP-based client management and order intake
- Mock ROS: REST-based route optimization
- **Mock WMS**: TCP/IP messaging for warehouse operations

2. Middleware Architecture 🔽

- Protocol Translation: Bridge SOAP

 REST

 TCP/IP
- **Service Orchestration**: Coordinate between mock systems
- Message Broker: Handle asynchronous communication

3. Client Applications 🔽

- Web Portal: Client login, order submission, status tracking
- Mobile App: Driver manifest, route view, delivery updates

4. Core Functionality

- Order Processing Workflow: End-to-end order handling
- Real-time Updates: Live tracking using WebSockets
- High-volume Processing: Handle multiple orders asynchronously

5. Required Documentation

- Architecture Diagrams: Conceptual and implementation
- Alternative Architectures: 2 alternatives with rationales
- Technology Justification: Why specific tools chosen
- Security Considerations: Information security analysis

6. Demonstration 🔽

- Basic Order Flow: Client submission through all systems
- Real-time Client UI: Show live updates
- 10-minute Presentation: All team members participate

This structure focuses ONLY on what the assignment explicitly requires - no extras.