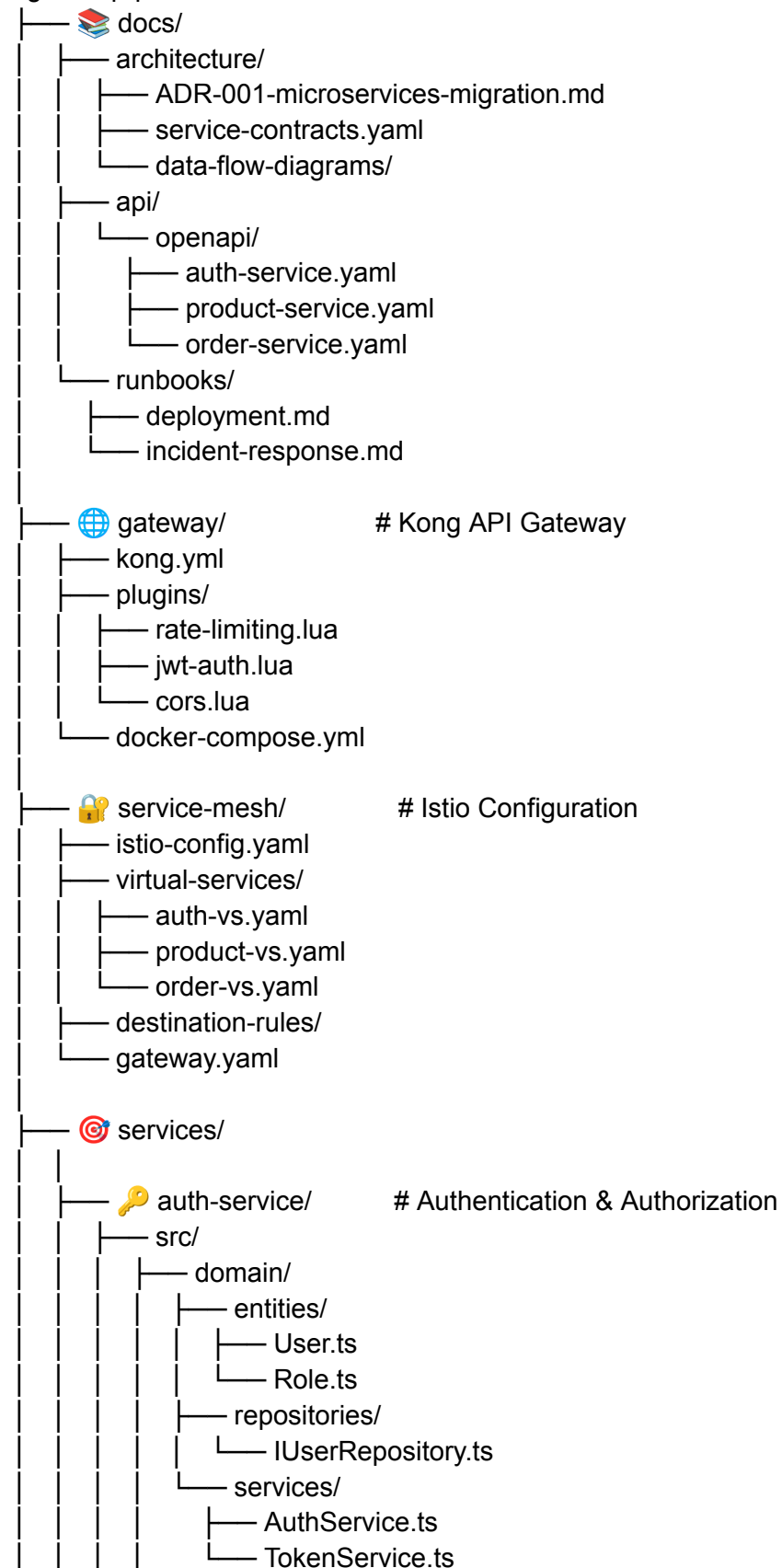
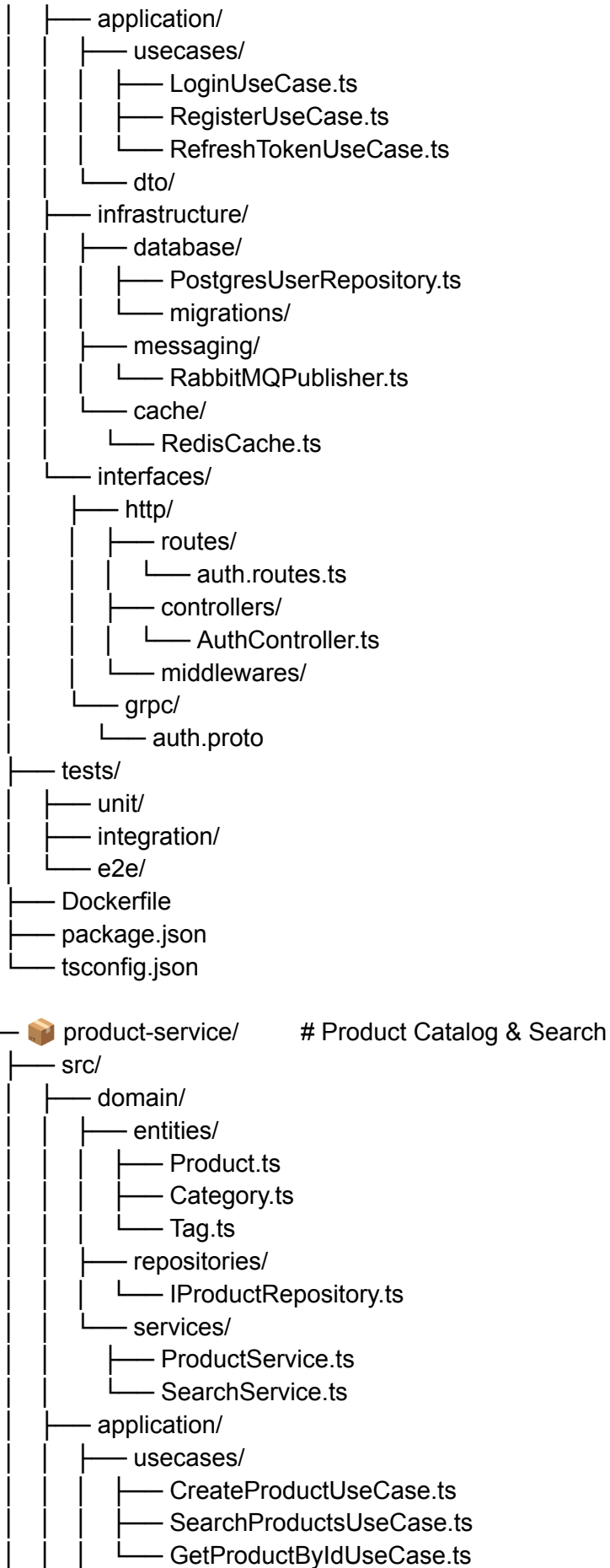


📁 Structure AgroDeep Platform - Microservices Architecture


...

agrodeep-platform/





- └─ dto/
- └─ infrastructure/
 - └─ database/
 - └─ PostgresProductRepository.ts
 - └─ ElasticsearchProductRepository.ts
 - └─ storage/
 - └─ MinIOStorage.ts
- └─ interfaces/
 - └─ http/
 - └─ routes/
 - └─ products.routes.ts
 - └─ categories.routes.ts
 - └─ events/
 - └─ ProductEventHandler.ts
- └─ tests/
- └─ Dockerfile
- └─ package.json

- └─  order-service/ # Order Management
 - └─ src/
 - └─ domain/
 - └─ entities/
 - └─ Order.ts
 - └─ OrderItem.ts
 - └─ OrderStatus.ts
 - └─ aggregates/
 - └─ OrderAggregate.ts
 - └─ services/
 - └─ OrderService.ts
 - └─ application/
 - └─ usecases/
 - └─ CreateOrderUseCase.ts
 - └─ UpdateOrderStatusUseCase.ts
 - └─ GetOrderHistoryUseCase.ts
 - └─ sagas/
 - └─ OrderSaga.ts
 - └─ dto/
 - └─ infrastructure/
 - └─ database/
 - └─ PostgresOrderRepository.ts
 - └─ messaging/
 - └─ OrderEventPublisher.ts
 - └─ PaymentEventConsumer.ts
 - └─ cache/
 - └─ interfaces/
 - └─ http/
 - └─ events/
 - └─ tests/

- └─ Dockerfile
- └─ package.json

- └─ 🏠 payment-service/ # Payment Integration

- └─ src/
 - └─ domain/
 - └─ entities/
 - └─ Payment.ts
 - └─ PaymentMethod.ts
 - └─ services/
 - └─ StripePaymentService.ts
 - └─ PayPalPaymentService.ts
 - └─ application/
 - └─ usecases/
 - └─ ProcessPaymentUseCase.ts
 - └─ RefundPaymentUseCase.ts
 - └─ dto/
 - └─ infrastructure/
 - └─ database/
 - └─ external/
 - └─ StripeClient.ts
 - └─ PayPalClient.ts
 - └─ messaging/
 - └─ interfaces/
- └─ tests/
- └─ Dockerfile
- └─ package.json

- └─ 🚚 delivery-service/ # GPS Tracking & Route Optimization

- └─ src/
 - └─ domain/
 - └─ entities/
 - └─ Delivery.ts
 - └─ Vehicle.ts
 - └─ services/
 - └─ TrackingService.ts
 - └─ RouteOptimizationService.ts
 - └─ application/
 - └─ usecases/
 - └─ CreateDeliveryUseCase.ts
 - └─ UpdateLocationUseCase.ts
 - └─ dto/
 - └─ infrastructure/
 - └─ database/
 - └─ PostgresDeliveryRepository.ts
 - └─ QdrantVectorDB.ts
 - └─ messaging/
 - └─ websocket/

- └─ LocationBroadcaster.ts
- └─ interfaces/
- └─ tests/
- └─ Dockerfile
- └─ package.json



ai-service/ # ML Recommendations & NLP

- └─ src/
 - └─ domain/
 - └─ entities/
 - └─ Recommendation.ts
 - └─ services/
 - └─ RecommendationEngine.ts
 - └─ SentimentAnalysisService.ts
 - └─ application/
 - └─ usecases/
 - └─ GetRecommendationsUseCase.ts
 - └─ AnalyzeReviewsUseCase.ts
 - └─ dto/
 - └─ infrastructure/
 - └─ database/
 - └─ QdrantVectorRepository.ts
 - └─ ml/
 - └─ models/
 - └─ collaborative_filtering.py
 - └─ bert_sentiment.py
 - └─ training/
 - └─ cache/
 - └─ interfaces/
- └─ models/ # ML Models
- └─ checkpoints/
- └─ tests/
- └─ Dockerfile
- └─ requirements.txt



analytics-service/ # ClickHouse Analytics

- └─ src/
 - └─ domain/
 - └─ entities/
 - └─ Metric.ts
 - └─ services/
 - └─ AnalyticsService.ts
 - └─ application/
 - └─ usecases/
 - └─ GenerateReportUseCase.ts
 - └─ GetKPIsUseCase.ts
 - └─ dto/
 - └─ infrastructure/

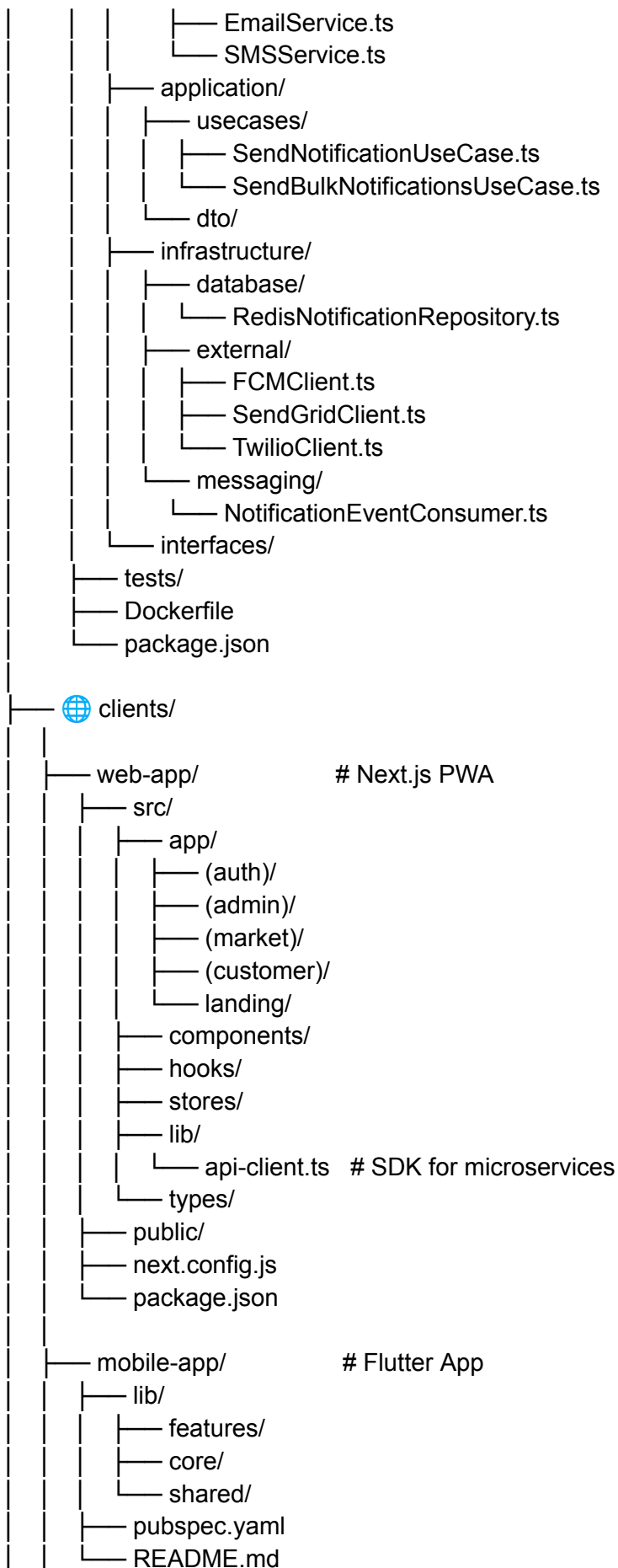
- database/
 - ClickHouseRepository.ts
- streaming/
 - KafkaConsumer.ts
- reporting/
 - PDFGenerator.ts
- interfaces/
- tests/
- Dockerfile
- package.json

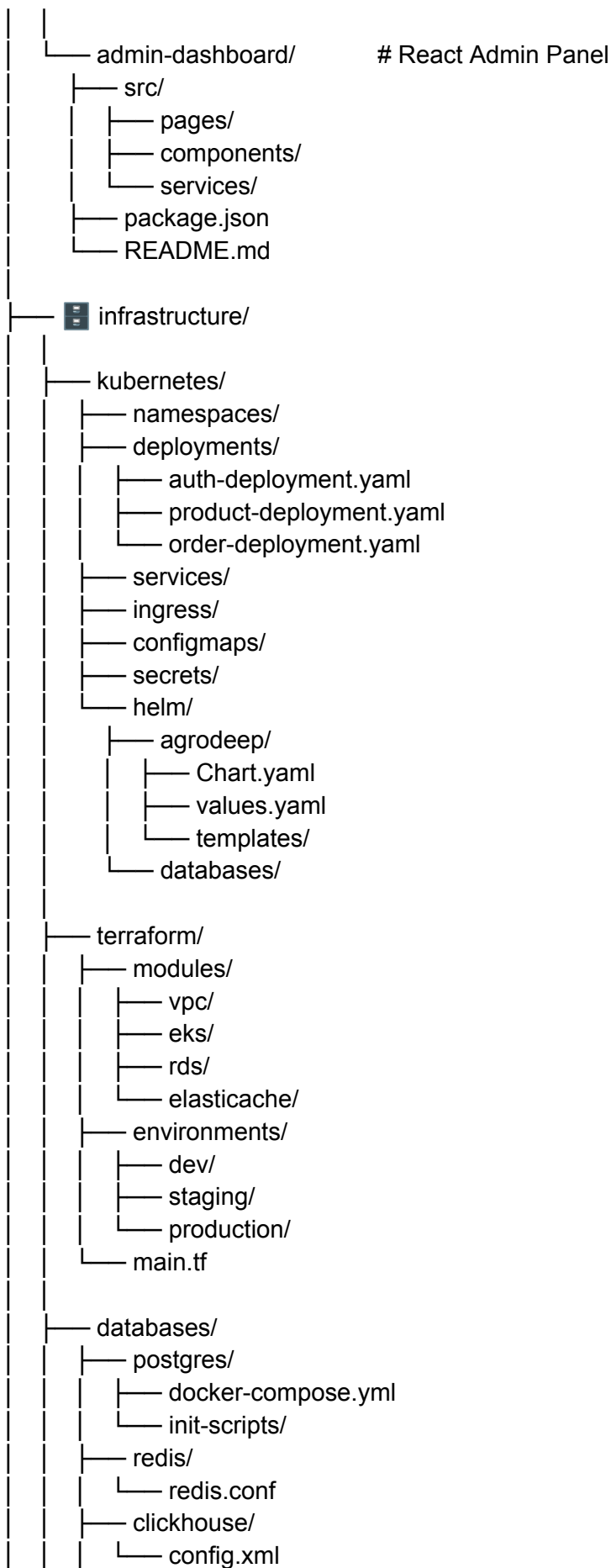
blockchain-service/ # Hyperledger Fabric Traceability

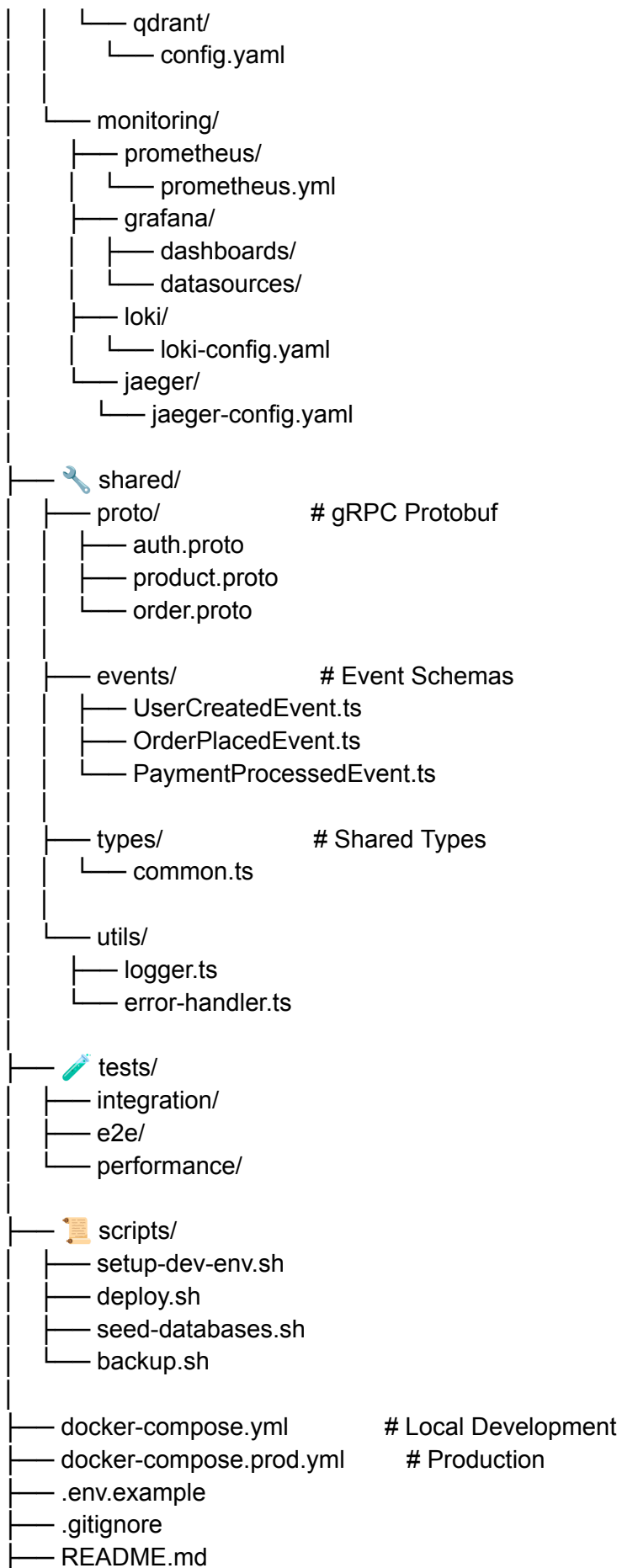
- chaincode/
 - product-trace/
 - lib/
 - ProductTrace.ts
 - index.ts
 - package.json
 - transaction-history/
- src/
 - domain/
 - services/
 - BlockchainService.ts
 - application/
 - usecases/
 - RecordTransactionUseCase.ts
 - VerifyTraceabilityUseCase.ts
 - infrastructure/
 - fabric/
 - FabricClient.ts
 - network-config.yaml
 - database/
 - interfaces/
- network/ # Fabric Network Config
- docker-compose.yaml
- crypto-config/
- channel-artifacts/
- tests/
- Dockerfile
- package.json

notification-service/ # Push, Email, SMS

- src/
 - domain/
 - entities/
 - Notification.ts
 - services/
 - PushService.ts







```
└─ package.json
...
```

```
---
```

```
## 📊 **Database Distribution**
```

```
### **Service → Database Mapping**
```

```
| Service | Primary DB | Secondary | Cache |
|-----|-----|-----|-----|
| Auth Service | PostgreSQL | - | Redis |
| Product Service | PostgreSQL | Elasticsearch | Redis |
| Order Service | PostgreSQL | - | Redis |
| Payment Service | PostgreSQL | - | Redis |
| Delivery Service | PostgreSQL | QdrantDB | Redis |
| AI Service | QdrantDB | ClickHouse | Redis |
| Analytics Service | ClickHouse | - | Redis |
| Blockchain Service | Hyperledger | PostgreSQL | - |
| Notification Service | Redis | PostgreSQL | - |
```

```
---
```

```
## 🔄 **Event-Driven Communication**
```

```
### **Event Types**
```

```
``typescript
// User Events
UserCreatedEvent
UserUpdatedEvent
UserDeletedEvent

// Product Events
ProductCreatedEvent
ProductUpdatedEvent
ProductDeletedEvent
ProductViewedEvent

// Order Events
OrderPlacedEvent
OrderConfirmedEvent
OrderShippedEvent
OrderDeliveredEvent
OrderCancelledEvent

// Payment Events
PaymentInitiatedEvent
```

PaymentProcessedEvent
PaymentFailedEvent
PaymentRefundedEvent

// Delivery Events
DeliveryAssignedEvent
DeliveryStartedEvent
LocationUpdatedEvent
DeliveryCompletedEvent
...

🛡️ ****Security Architecture****

****Authentication Flow****

...

Client → Kong Gateway → JWT Validation → Service Mesh (mTLS) → Microservice

...

****Service-to-Service Communication****

...

Service A → Istio Sidecar → mTLS → Istio Sidecar → Service B

...

📈 ****Scalability Patterns****

****Horizontal Scaling Strategy****

```yaml

# Kubernetes HPA Configuration

apiVersion: autoscaling/v2

kind: HorizontalPodAutoscaler

metadata:

name: product-service-hpa

spec:

scaleTargetRef:

apiVersion: apps/v1

kind: Deployment

name: product-service

minReplicas: 2

maxReplicas: 10

metrics:

- type: Resource

```
resource:
 name: cpu
 target:
 type: Utilization
 averageUtilization: 70
...

```

## ## 🎯 \*\*Migration Strategy\*\*

### ### \*\*Phase 1: Infrastructure Setup\*\* (Week 1-2)

- Setup Kubernetes cluster
- Deploy Kong Gateway
- Configure Istio Service Mesh
- Setup databases (PostgreSQL, Redis, ClickHouse, QdrantDB)

### ### \*\*Phase 2: Core Services\*\* (Week 3-4)

- Migrate Auth Service
- Migrate Product Service
- Migrate Order Service

### ### \*\*Phase 3: Advanced Services\*\* (Week 5-6)

- Implement Payment Service
- Implement Delivery Service
- Implement AI Service

### ### \*\*Phase 4: Analytics & Blockchain\*\* (Week 7-8)

- Setup Analytics Service
- Implement Blockchain Service
- Implement Notification Service

### ### \*\*Phase 5: Testing & Optimization\*\* (Week 9-10)

- Integration testing
- Performance testing
- Security audit
- Production deployment

---

## ## 📊 \*\*Monitoring Stack\*\*

...

Logs: Loki → Grafana

Metrics: Prometheus → Grafana

Traces: Jaeger

APM: OpenTelemetry

Alerts: Alertmanager → Slack/PagerDuty

