Developer Guide — Nidula

Restaurant Platform Engineering

Project Responsibility Summary

You are responsible for the system backbone and analytics. This includes core backend logic and data intelligence services.

- Order Service with lifecycle engine
- Session and table management
- RBAC roles and permissions
- Analytics events pipeline
- ETL from MongoDB raw events to ClickHouse
- Analytics API development
- Admin Analytics Dashboard in React
- System architectural consistency and guidance

Tech Stack

- Spring Boot (Java)
- MySQL (orders, sessions, RBAC)
- Kafka (event streaming)
- MongoDB (raw event store)
- ClickHouse (analytics warehouse)
- Redis (session state cache optional)
- React with charts

Database Responsibilities

MySQL Tables

```
orders(
 id, session_id, restaurant_id, status,
 subtotal, tax, tip, total,
 created_at, updated_at
)

order_items(
 id, order_id, item_id, qty, unit_price, modifiers json
)

sessions(
 id, table_id, user_id, status, started_at, ended_at
)

tables(
 id, restaurant_id, number
)

roles(id, name)
permissions(id, code)
role_permissions(role_id, permission_id)
staff(id, restaurant_id, email, role_id)
```

MongoDB Raw Events

```
events_raw {
 eventId,
 type,
 restaurantId,
 sessionId,
 timestamp,
 payload{}
}
```

ClickHouse Warehouse

```
events(
 eventId UUID,
 type String,
 restaurantId String,
 sessionId String,
 timestamp DateTime,
 payload JSON
)
```

Endpoints

Orders

```
POST   /api/v1/orders
GET    /api/v1/orders/{orderId}
PATCH  /api/v1/orders/{orderId}/status
```

Sessions

```
POST   /api/v1/sessions/qr-checkin
GET    /api/v1/sessions/{sessionId}
PATCH  /api/v1/sessions/{sessionId}/end
```

RBAC

```
GET    /api/v1/roles
POST   /api/v1/roles
POST   /api/v1/staff
PATCH  /api/v1/staff/{id}
```

Analytics API

```
GET /api/v1/analytics/dashboard
GET /api/v1/analytics/peak-hours
GET /api/v1/analytics/menu
GET /api/v1/analytics/feedback
```

Backend Development Guidelines

Spring Boot Rules

- Use layered structure:

  ```
  controller/
  service/
  repository/
  entity/
  dto/
  config/
  ```

- Publish events for every state change
- Use Flyway migrations
- Cache hot data using Redis if required

Analytics Architecture Rules

- Write events to MongoDB first
- Batch load to ClickHouse or use streaming insert
- Use materialized views for fast reads
- Expose dashboard friendly JSON endpoints

Testing and Validation

- Unit test event producer and state engine
- Query benchmarking for dashboard endpoints

Frontend Development Guidelines

React Dashboard Rules

- Pages:

  ```
  sales, peak hours, menu performance, feedback insights
  ```

- Use charts:
  - Line: sales over time
  - Heat map: peak hours
  - Bar: top dishes
  - Pie: order breakdown
- Export to CSV
- Filters: date range, category, meal time

Definition of Done

- Orders and sessions fully functional
- RBAC integrated and enforced
- Analytics dashboard loaded under one second
- Mongo to ClickHouse ETL tested with real data