

Last indexed: 11 August 2025 (172577)

| |
|---|
| Overview |
| Repository Structure |
| Getting Started |
| Project 1: Product Management (Proy1) |
| Database & Configuration |
| Product Management System |
| Development Environment |
| Project 2: Course Platform (Proy2_Cursos) |
| Data Models & Database |
| Authentication & Middleware |
| Development & Testing |
| Project 3: Food Delivery Platform (Proy3_Pedidos) |
| Architecture & Dependencies |
| Database Models & Schema |
| Environment & Configuration |
| Development Environment |
| VS Code & Debugging |
| Dependency Management |

Project 3: Food Delivery Platform (Proy3_Pedidos)

> Relevant source files

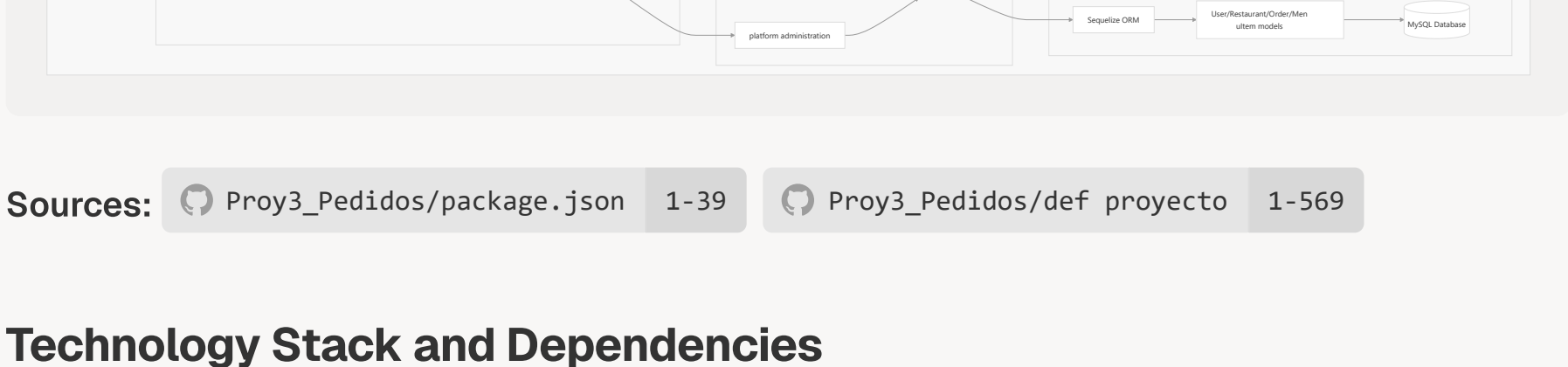
Purpose and Scope



This document covers FoodExpress, a comprehensive food delivery platform implementation that represents the most sophisticated project in the IronHack Course 2 repository. The system implements a multi-role web application connecting customers, restaurants, and platform administrators through a complete order management workflow.

This page focuses on the overall system architecture, technology stack, and core business logic. For database model details, see [Database Models & Schema](#). For environment configuration specifics, see [Environment & Configuration](#). For overall monorepo structure, see [Overview](#).

System Overview

FoodExpress is built as an Express.js application with a complex multi-role authentication system supporting three distinct user types: clients, restaurant administrators, and platform administrators. The system manages the complete food delivery lifecycle from restaurant menu management through order placement and status tracking.



Sources:  Proy3_Pedidos/package.json 1-39  Proy3_Pedidos/def proyecto 1-569

Technology Stack and Dependencies

The FoodExpress platform utilizes an extensive technology stack with 15 production dependencies and 2 development dependencies, representing a mature web application architecture.

Core Framework Dependencies

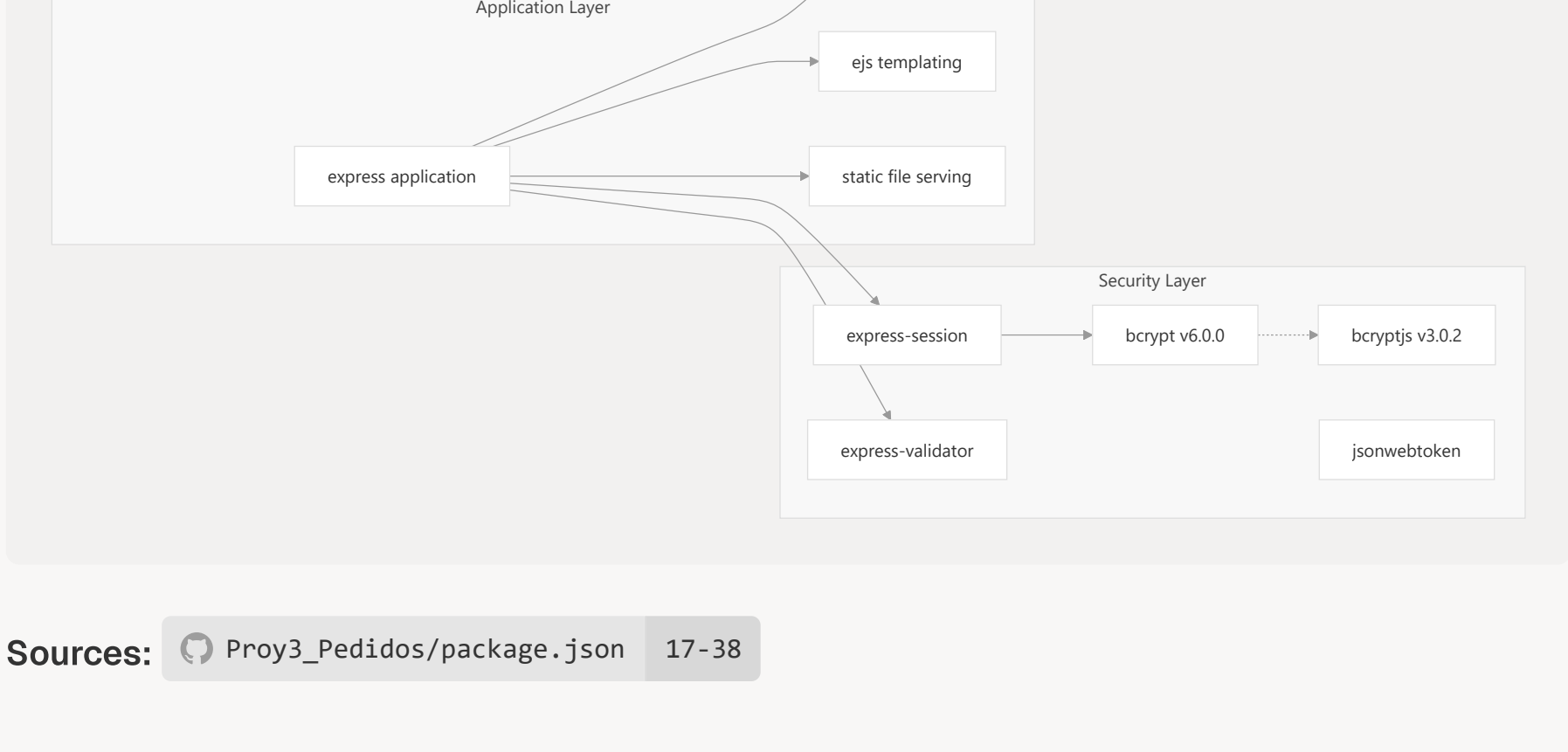
| Dependency | Version | Purpose |
|------------------------|---------|---------------------------|
| <code>express</code> | ~4.18.2 | Core web framework |
| <code>ejs</code> | ~3.1.9 | Templating engine |
| <code>mysql2</code> | ~3.9.1 | MySQL database driver |
| <code>sequelize</code> | ~6.37.7 | Object-relational mapping |

Authentication and Security

| Dependency | Version | Purpose |
|--------------------------------|---------|-----------------------------|
| <code>bcrypt</code> | ~6.0.0 | Password hashing (primary) |
| <code>bcryptjs</code> | ~3.0.2 | Password hashing (fallback) |
| <code>express-session</code> | ~1.18.2 | Session management |
| <code>jsonwebtoken</code> | ~9.0.2 | JWT token handling |
| <code>express-validator</code> | ~7.0.1 | Input validation |

Enhanced Features

| Dependency | Version | Purpose |
|------------------------------|---------|---------------------------|
| <code>dotenv</code> | ~16.4.5 | Environment configuration |
| <code>connect-flash</code> | ~0.1.1 | Flash messaging |
| <code>method-override</code> | ~3.0.0 | HTTP method override |
| <code>morgan</code> | ~1.10.1 | HTTP request logging |
| <code>multer</code> | ~2.0.2 | File upload handling |

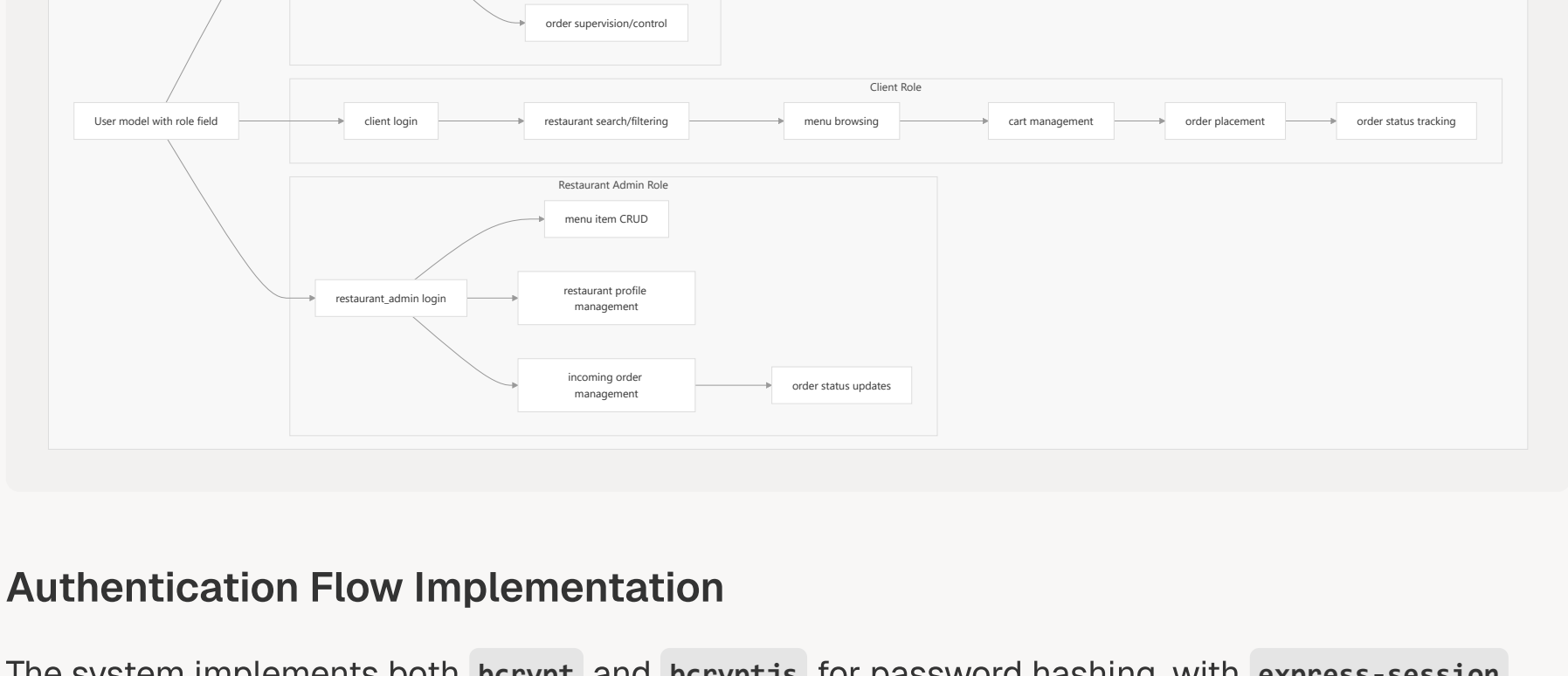


Sources:  Proy3_Pedidos/package.json 17-38

Multi-Role Authentication Architecture

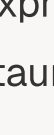

FoodExpress implements a sophisticated role-based authentication system supporting three distinct user roles with different capabilities and access patterns.

Role Definitions and Capabilities



Authentication Flow Implementation

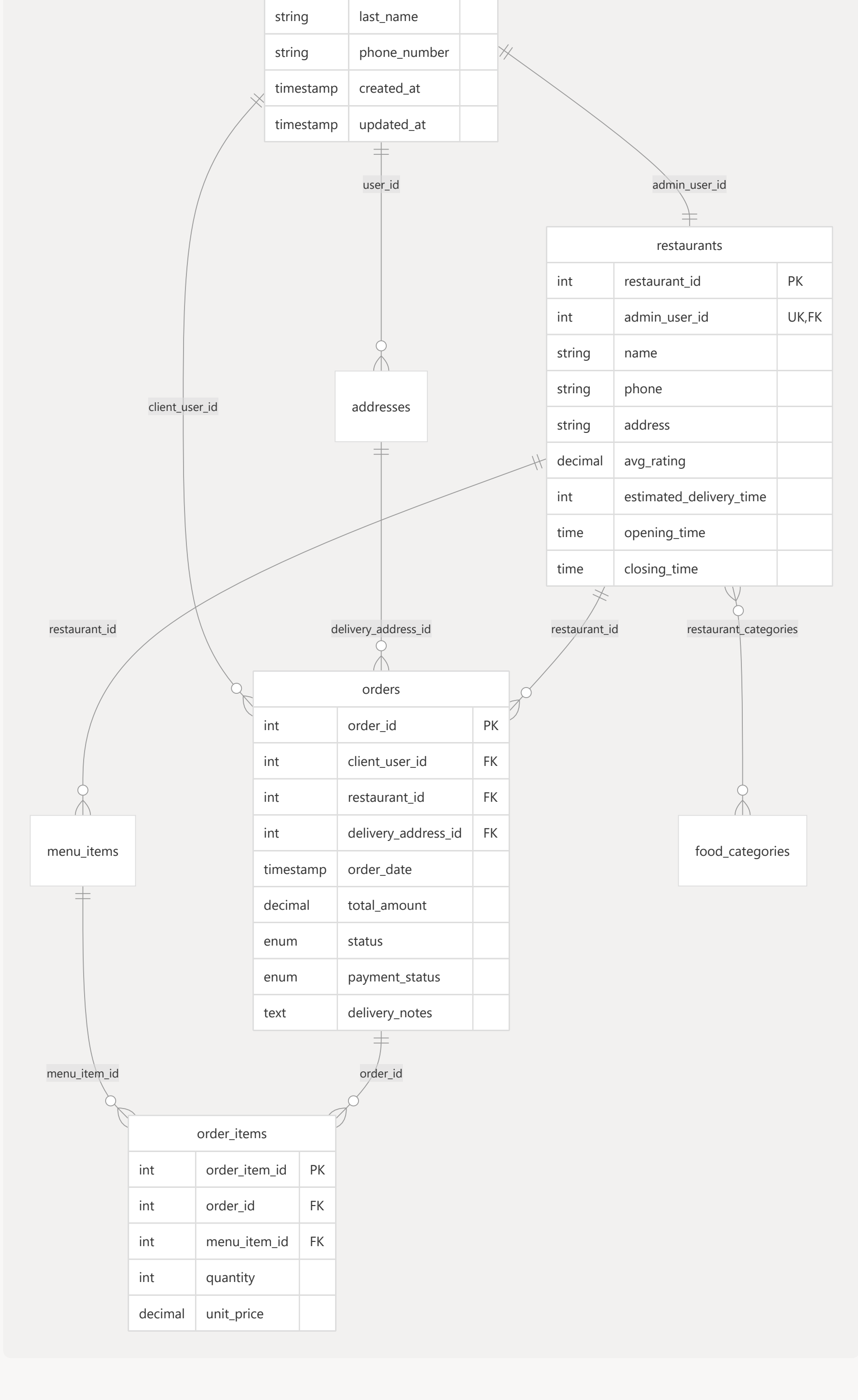
The system implements both `bcrypt` and `bcryptjs` for password hashing, with `express-session` managing user sessions and role-based middleware controlling access to protected routes.

Sources:  Proy3_Pedidos/def proyecto 61-85  Proy3_Pedidos/package.json 18-26

Database Schema and Relationships

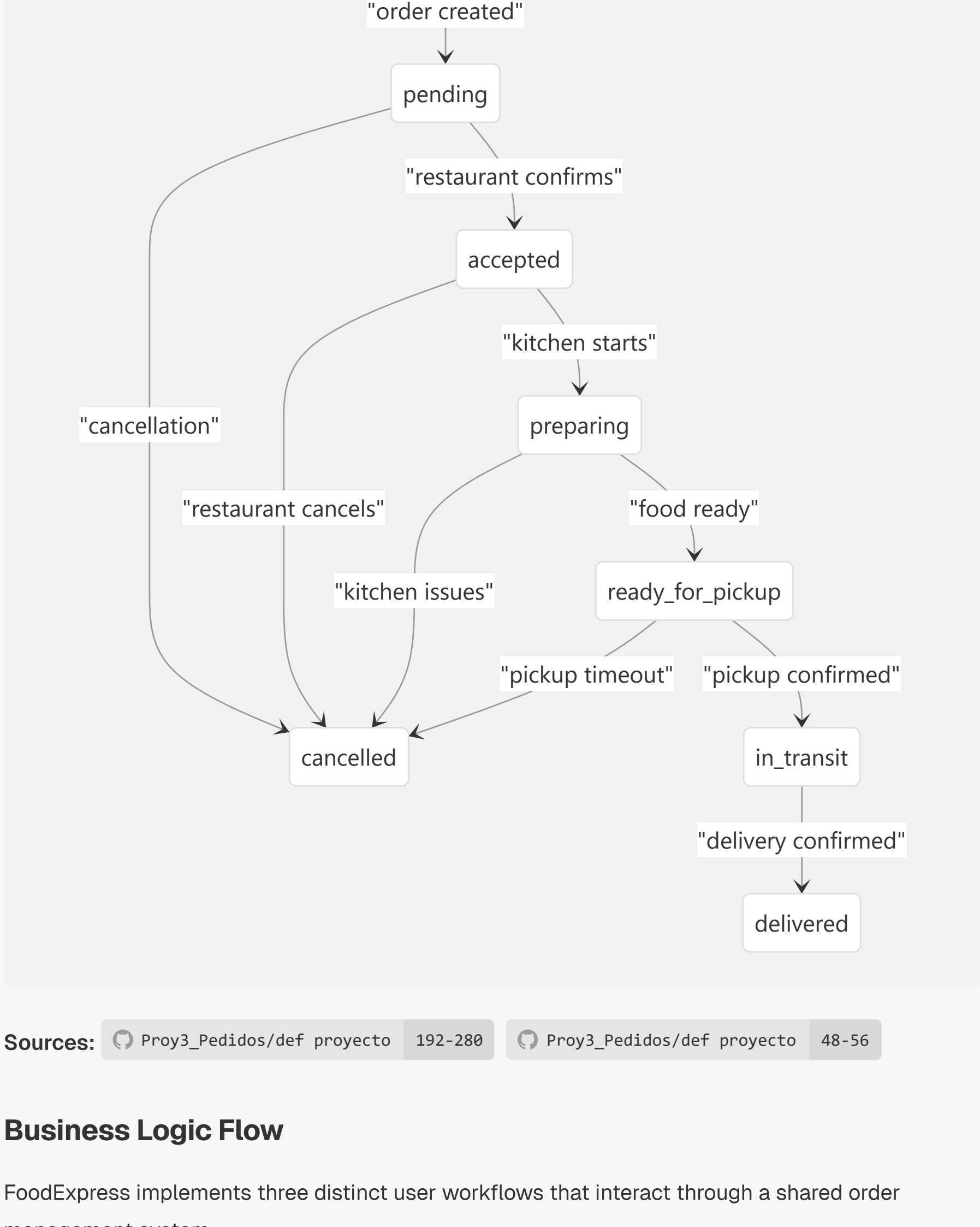
The FoodExpress database implements a complex relational schema designed to support multi-tenant restaurant operations with detailed order tracking and user management.

Core Entity Relationships



Order Status State Machine

The system implements a comprehensive order lifecycle with seven distinct states:

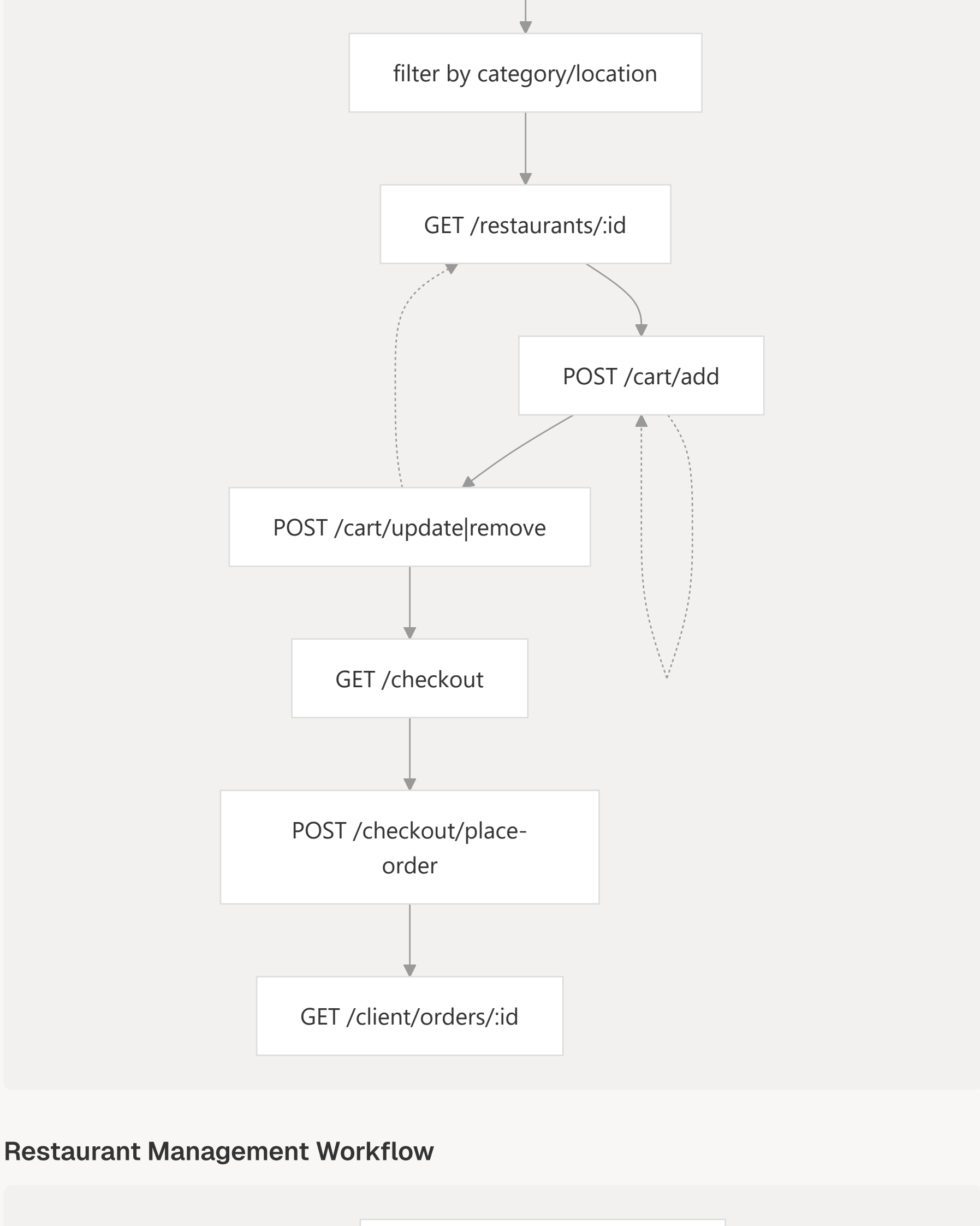


Sources:  Proy3_Pedidos/def proyecto 192-288  Proy3_Pedidos/def proyecto 48-56

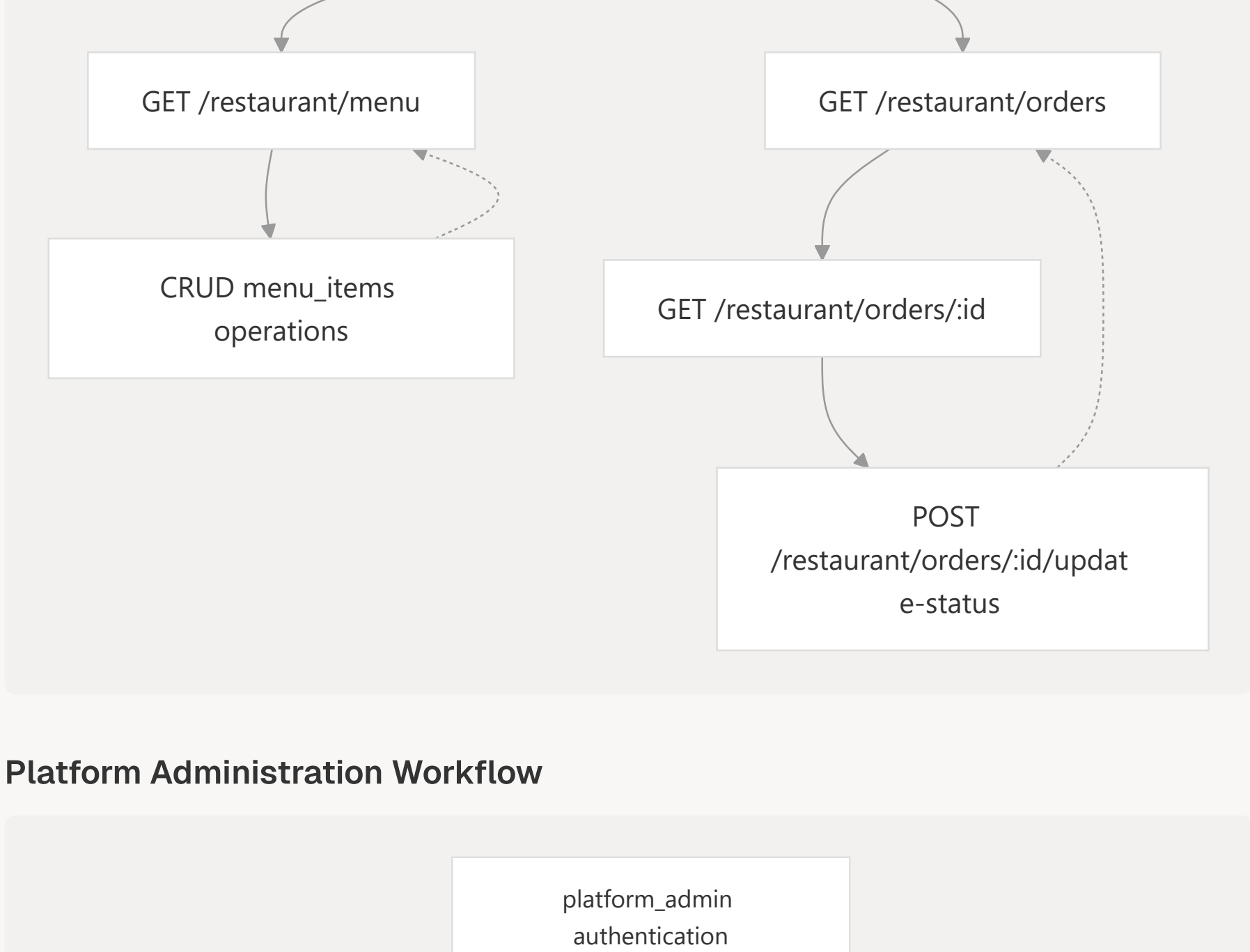
Business Logic Flow

FoodExpress implements three distinct user workflows that interact through a shared order management system.

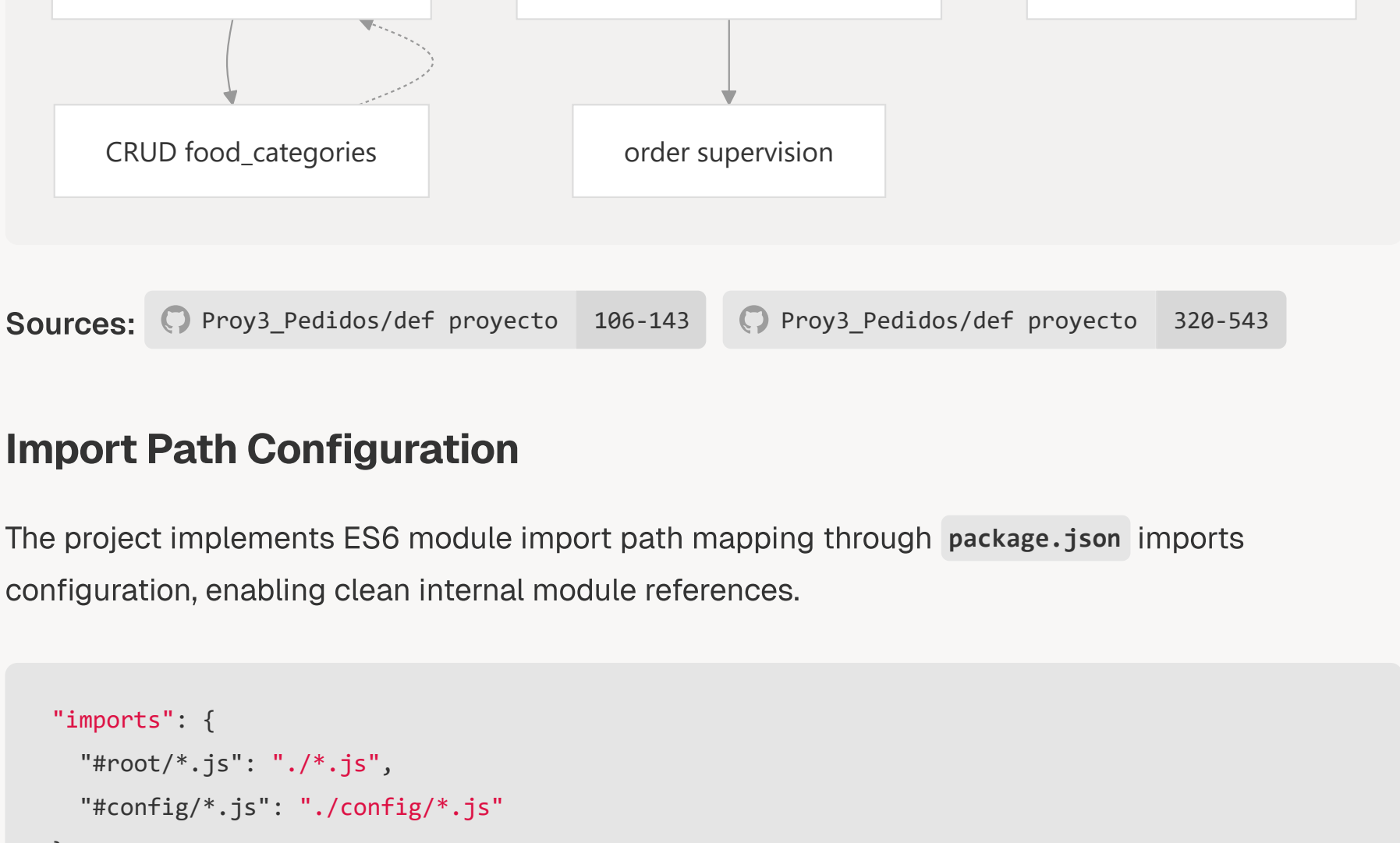
Client Order Workflow

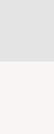
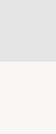


Restaurant Management Workflow



Platform Administration Workflow




Sources:  Proy3_Pedidos/def proyecto 186-143  Proy3_Pedidos/def proyecto 328-543

Import Path Configuration

The project implements ES6 module import path mapping through `package.json` imports configuration, enabling clean internal module references.

```
"imports": {
  "#root/*.*js": ".//*.*js",
  "#config/*.*js": "./config/*.*js"
}
```

This configuration allows modules to import using hash-prefixed paths like `#config/db.js` instead of relative paths, improving code maintainability and refactoring capabilities.

Sources:  Proy3_Pedidos/package.json 6-9

Development and Production Scripts

The application provides separate execution modes optimized for different environments:

| Script | Command | Purpose |
|--------------------|-------------------------------|-------------------------------|
| <code>start</code> | <code>node index.js</code> | Production execution |
| <code>dev</code> | <code>nodemon index.js</code> | Development with auto-restart |

The development dependencies include `nodemon` for automatic server restarts during development and `concurrently` for coordinated multi-process execution within the monorepo structure.

Sources: Proy3_Pedidos/package.json 10-13 Proy3_Pedidos/package.json 35-37