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

Architecture & Dependencies

> Relevant source files

This document details the comprehensive technology stack and dependency management for Project 3 (Proy3_Pedidos), the FoodExpress food delivery platform. It covers the Express.js application architecture, database integration, authentication systems, and development tooling that enable the multi-role order management system.

For information about the database models and schema design, see <u>Database Models & Schema</u>. For environment configuration details, see **Environment & Configuration**.

Core Technology Stack

framework. The application leverages MySQL with Sequelize ORM for data persistence and implements comprehensive authentication and session management.

The FoodExpress platform is built on a modern Node.js stack with Express.js as the primary web



Core Framework Dependencies

The application's foundation rests on express@4.18.2 Proy3_Pedidos/package.json 23 as the web server framework, with ejs@3.1.9 Proy3_Pedidos/package.json 22 providing server-side templating capabilities for dynamic HTML generation.

Database Integration

Database connectivity is handled through mysq12@3.9.1 Proy3_Pedidos/package.json for Proy3_Pedidos/package.json 31 MySQL database connections, while sequelize@6.37.7 Proy3_Pedidos/package.json 33 is provides ORM functionality. Additionally, sqlite3@5.1.7 included for potential local development or testing scenarios.

Sources: Proy3_Pedidos/package.json 17-33

The application implements a layered dependency structure that separates concerns between

Dependency Architecture

authentication, data persistence, request handling, and development tooling.



Authentication Stack

```
The platform implements dual password hashing with both bcrypt@6.0.0
Proy3_Pedidos/package.json 18 and bcryptjs@3.0.2 Proy3_Pedidos/package.json 19 for
cryptographic security. JWT token management is handled by jsonwebtoken@9.0.2
                                while session persistence uses express-session@1.18.2
 Proy3_Pedidos/package.json 26
                                with connect-flash@0.1.1 Proy3_Pedidos/package.json 20
 Proy3_Pedidos/package.json 24
for user feedback messaging.
```

Middleware Chain

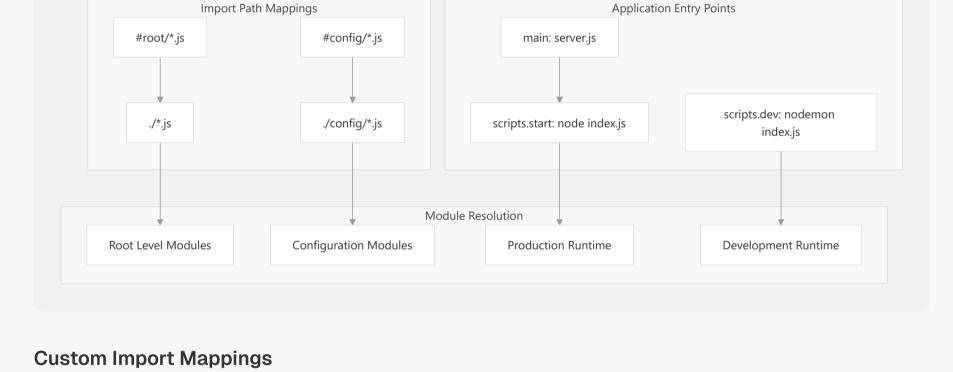
Proy3_Pedidos/package.json 28 for HTTP logging, express-validator@7.0.1 Proy3_Pedidos/package.json 25 for input validation, multer@2.0.2 Proy3_Pedidos/package.json 29 for file upload handling, and method-override@3.0.0 Proy3_Pedidos/package.json 27 for HTTP method support. Sources: Proy3_Pedidos/package.json 18-32

Request processing flows through a comprehensive middleware stack including morgan@1.10.1

Module System and Imports

package configuration.

The application uses modern ES6 module syntax with custom import mappings defined in the



The package defines custom import path mappings Proy3_Pedidos/package.json 6-9 with #root/*.js resolving to root-level modules and #config/*.js resolving to configuration-specific

modules in the config directory. **Application Entry Points** The main application entry point is configured as server.js Proy3_Pedidos/package.json 5

though the actual startup scripts reference index.js for both production (node index.js) and

development (nodemon index.js) environments Proy3_Pedidos/package.json 11-12

Development Environment

Sources: Proy3_Pedidos/package.json 5-12

The development environment is optimized for rapid iteration with hot-reloading and concurrent process management.

Tool Version **Purpose** 3.0.3 Development server with auto-restart nodemon

concurrently	9.2.0	Multi-process execution	
dotenv	16.4.5	Environment variable management	

Proy3_Pedidos/package.json 37 for automatic server

Environment

Production

Development

Dependency Categories

Development Dependencies

Proy3_Pedidos/package.json 17-38

restarts on file changes, while concurrently@9.2.0 Proy3_Pedidos/package.json 36 enables running multiple development processes simultaneously.

Development mode uses nodemon@3.0.3

connections, API keys, and application settings.

Command

node index.js

nodemon index.js

Environment Configuration Environment variable management is handled through dotenv@16.4.5 Proy3_Pedidos/package.json 21 providing secure configuration management for database

Script start

Version Constraints

dev

Script Commands

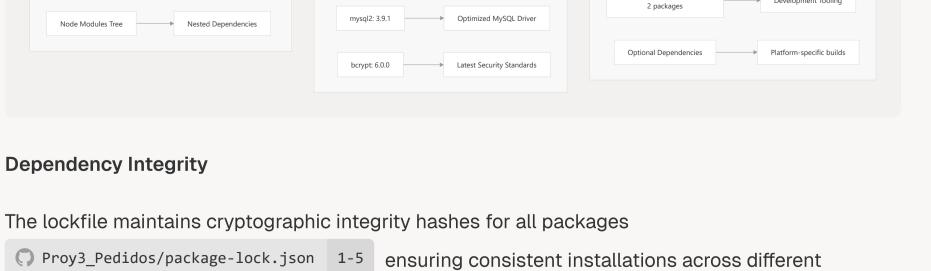
Sources: Proy3_Pedidos/package.json	35-38	<pre>Proy3_Pedidos/package.json</pre>	10-12
Package Lock and Depender	ncy Re	esolution	
The application uses npm's lockfile versi	ion 3 for	mat for precise dependency res	olution a

Key Package Versions

Sources: Proy3_Pedidos/package-lock.json 1-40

reproducible builds across environments.

Lock File Structure package-lock.json



Key dependencies maintain specific version ranges with express pinned to the 4.x series for API stability, sequelize using the latest 6.x features for modern ORM capabilities, and security-critical packages like bcrypt using the most recent stable releases.

environments and preventing supply chain attacks through package verification.