07ASN – Digital Game Store System

Jordan’s Premier Digital Gaming Marketplace

Prepared by
Hasan Mansour
Coding Academy by Orange

Date
20 April 2025

Table of Contents

1. Introduction

07ASN GamesStore-System is a full-featured digital marketplace designed to transform an already-profitable social-media gaming operation into a professional web platform. Leveraging the PERN stack (PostgreSQL, Express.js, React.js, Node.js) and styled with Tailwind CSS, it specializes in:

- **Digital Games**
- **Verified Gaming Accounts**
- **Instant Code Delivery**

As Jordan's first dedicated online hub for digital-gaming products and services, 07ASN delivers enterprise-grade security, seamless user experiences, and 24/7 transaction capability.

2. Problem Statement

Despite strong revenue via Instagram and Facebook, the business faced:

- **Fragmented Customer Experience**
  Order requests and support were handled via DMs and comments, leading to miscommunication and lost sales.
- **Trust & Professionalism Gaps**
  Lack of a branded website deterred customers from high-value purchases.
- **Manual Delivery Overhead**
  Sending codes/accounts by hand caused delays, errors, and inconsistent fulfillment times.
- **Scalability Constraints**
  Managing inventory, payments, and listings on social platforms became error-prone as volume grew.

3. Proposed Solution

The 07ASN platform addresses these challenges by:

1. **Centralizing Sales & Support**
   A single portal for browsing, purchasing, order-tracking, and customer tickets.
2. **Professional Brand Presence**
   Custom domain and polished UI/UX (React + Tailwind) to boost credibility.

3. **Automated Instant Delivery**
   Backend logic (Node.js + Express + Sequelize) triggers sub-30-second code/account emails upon purchase.
4. **Robust Security & Scalability**
   JWT-based auth, OWASP-aware input validation, HTTPS enforcement, and scalable, stateless services.
5. **Administrative Efficiency**
   Admin dashboard for inventory management, listing approvals, analytics, and engagement tracking.

4. Framework Skills

- **React JS** – Built a dynamic, component-based front end using hooks and Context API for state.
- **Tailwind CSS** – Applied utility-first styling for a responsive, mobile-first design.
- **Node JS** – Powered the REST API runtime for user auth, product management, and order processing.
- **Express JS** – Structured routing and middleware for secure endpoint handling.
- **Sequelize (ORM)** – Defined models and queries against PostgreSQL, including migrations and transactions.
- **PostgreSQL** – Served as the relational datastore for users, products, orders, and logs.

5. Functional Specifications

5.1 Business Goals

- Launch a professional web presence to expand market reach in Jordan.
- Increase conversions and customer trust.
- Ensure instant, reliable delivery of all digital goods.

5.2 User Stories

- **Guest**
  - Browse games by platform (PS, Xbox, Steam, PC).
  - Filter by price, popularity, and genre.
- **Registered User**
  - Sign up / log in (email + password + optional 2FA).
  - One-click purchase of games/accounts.
  - Automatic delivery email with unique code.

  - View order history, wishlist, and wallet.
- **Admin**
  - Create / edit / delete products.
  - Approve or reject marketplace listings.
  - Access sales analytics and customer metrics.

6. Technical Specifications

Architecture Overview

- **Front-end**: React, styled with Tailwind CSS, state via Redux Toolkit.
- **Back-end**: Node.js + Express.js.
- **Database**: PostgreSQL (Sequelize ORM).
- **Third-Party**: JWT for auth; Cloudinary for media; PayPal for payments.

6.2 Non-Functional Requirements

- **Performance**: API responses < 200 ms under typical load.
- **Scalability**: Stateless services to allow horizontal scaling.
- **Security**: Input validation (OWASP), JWT auth, HTTPS only.
- **Responsiveness**: Fully mobile-friendly; tested on iOS/Android.

7. Code Excerpts & Justification

7.1 Authentication Middleware :

Stateless JWT sessions improve scalability and eliminate server-side session storage.

7.2 Product Listing Endpoint

Database-level filtering removes sold-out products before they reach the UI, optimizing performance.

7.3 React Component

A reusable component pattern with Tailwind for rapid styling; click handler integrates with global cart context for simplicity.

8. Test Case for Representative Feature

Test Case ID	**TC-001**
Description	Verify that purchasing a game triggers immediate code delivery email.
Preconditions	User is logged in; product inventoryCount > 0.
Test Steps	1. Navigate to product page. 2. Click “Buy Now”. 3. Confirm checkout. 4. Observe email inbox.
Input Data	User credentials; Game ID 1234.
Expected Result	Order status → “Delivered”; email arrives within 30 s with unique code.
Actual Result	After clicking *Buy Now*, the order was “Pending” then updated to “Delivered” in 12 seconds, and an email with code `XYZ-123-ABC` arrived at 14:32:07.
Pass/Fail	**Pass** if every expectation was met (delivery happened within 30 s, correct email format, order status updated, etc.). **Fail** if something differed (no email, wrong code, delay >30 s, order stuck in “Pending,” etc.).
Notes/Screenshots	See Appendix A: backend log excerpt and email screenshot.