**DOCUMENTATION INDEX**

* **Objectives & Scope**
* **Technology Stack**
* **Deployment & Usage Instructions**
* **Security Considerations**
* **Testing & Validation**
* **Conclusion**

**🍦 CRAVE ICE-CREAM: Project Documentation**

**1. Introduction**

The CRAVE ICE-CREAM project is a small-scale dynamic website that transforms a fun, static UI into a fully functional ordering platform. It demonstrates the combination of interactive frontend design with a Node.js backend and a simple admin dashboard for managing orders.

This project is suitable for learning full-stack development concepts while delivering an engaging, playful user experience.

**2. Objectives & Scope**

**Objectives**

* Provide a visually attractive and playful UI for browsing ice cream flavors.
* Enable customers to place ice cream orders seamlessly.
* Provide an admin interface for monitoring and managing orders.
* Ensure responsiveness and cross-device compatibility.

**Scope**

* **In-Scope:**
  + Ice cream browsing
  + Dynamic order placement
  + Admin order confirmation/cancellation
  + Persistent order storage
* **Out-of-Scope:**
  + User login/registration
  + Payment gateway integration
  + Delivery tracking

**3. System Architecture**

* **Frontend (UI Layer):**
  + Built with HTML, CSS, JavaScript.
  + Provides draggable flavor menu, floating emoji background, interactive cards, and order popup system.
* **Backend (Application Layer):**
  + Node.js + Express server (server.js).
  + Manages API endpoints for flavors and orders.
* **Data Layer:**
  + JSON files used as lightweight storage (flavors.json, orders.json).
* **Admin Dashboard:**
  + admin.html for order management.

**4. Technology Stack**

| **Layer** | **Technology** |
| --- | --- |
| Frontend | HTML5, CSS3, JavaScript (Vanilla JS) |
| Backend | Node.js, Express.js |
| Database | JSON file-based storage |
| Styling | Custom CSS animations, transitions |
| Deployment | Localhost or any Node-compatible host (Heroku, Render, Azure, etc.) |

**5. Features**

**Homepage UI**

✔ Vibrant ice cream-themed background  
✔ Floating emojis that animate infinitely  
✔ Draggable menu with flavors list  
✔ Flavor cards with hover overlays (price + “Order Now”)

**Order System**

✔ Orders stored dynamically in orders.json  
✔ Popup feedback after order placement  
✔ Prevents duplicate clicks with button disable feature

**Admin Dashboard**

✔ Table view of pending orders  
✔ Buttons for confirm ✅ and cancel ❌ actions  
✔ Instant popup notifications after each action

**6. Backend Services**

**Endpoints**

* GET /flavors → Fetches all flavors
* POST /order → Places new order
* GET /orders → Retrieves pending orders
* DELETE /orders/:id → Confirms or cancels an order

**Data Persistence**

* Orders automatically saved in orders.json.
* Flavors are read from flavors.json at startup.

**7. Deployment & Usage**

**Setup Instructions**

1. Install [Node.js](https://nodejs.org/).
2. Navigate to project directory:
3. cd crave-icecream
4. npm install express cors
5. Start server:
6. node server.js
7. Open index.html in browser (customer view).
8. Open admin.html in browser (admin view).

**Deployment Options**

* Host server on **Heroku / Render / Azure Web Apps**.
* Host frontend on **Netlify / GitHub Pages / Azure Storage**.
* Connect them via correct server URL in script.js and admin.html.

**8. Security Considerations**

* **CORS enabled** for local development.
* Input validation ensures orders require flavor & price.
* No authentication (future enhancement).
* Data stored in JSON, not production-grade database → suitable only for small-scale/demo use.

**9. Testing & Validation**

* **Frontend:**
  + Verified draggable menu works smoothly.
  + Confirmed hover overlay animations.
  + Checked popup dismissal after 3s.
* **Backend:**
  + Tested API endpoints using Postman.
  + Ensured orders persist across sessions.
* **Admin Panel:**
  + Confirmed orders removed after confirmation/cancellation.
  + Popup feedback tested successfully.

**10. Conclusion**

The **CRAVE ICE-CREAM** project delivers a **fun, interactive, full-stack web application**. It bridges static design with real backend functionality, offering both **user ordering** and **admin management** capabilities.

It is a strong **learning project** for web developers, showcasing:

* Dynamic UI interactions
* Node.js + Express backend integration
* JSON file-based persistence
* Full-stack workflow (user → server → admin)