Anshika Saini, 2026, 14:2
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Brew Haven A Browser-Based Café Ordering and Management System

Anshika Saini , Ayesha Garg , Nishant , Shivani Panchal , Ankur Kaushik
Department of CSE & IT
SD College of Engineering and Technology , Muzaffarnagar

Abstract- This paper presents BrewHaven, a café ordering and management system built entirely using HTML, CSS, and JavaScript—requiring no server, no database software, and no internet connection. The system consists of two self-contained files: a customer-facing ordering interface and an admin dashboard, both communicating exclusively through the browser's localStorage API. BrewHaven demonstrates that modern web technologies alone are sufficient to deliver a functional, visually polished digital ordering experience for small cafés. This paper describes the system architecture, design decisions, feature set, limitations, and proposed future enhancements.

Keywords: Web Application, Café Ordering System, JavaScript, localStorage, Single-Page Application, Admin Dashboard, Client-Side Architecture.

I. INTRODUCTION

The proliferation of digital ordering systems in large restaurant chains has highlighted a significant gap: small local cafés, particularly in emerging markets such as India, largely continue to rely on manual, paper-based ordering workflows. This approach introduces errors in order taking, imprecise billing, and a complete absence of operational analytics.
The primary barrier to digital adoption for small cafés is cost. Commercial Point-of-Sale (POS) software can cost thousands of rupees per month; server-based custom solutions require ongoing technical maintenance; and bespoke development is financially prohibitive for most small business owners.

BrewHaven was designed to address this gap directly. By constraining the system to run entirely within the browser — using only localStorage for persistence — the project eliminates all infrastructure costs and dependencies while still delivering a modern, responsive ordering experience.

A. Scope of This Paper

This paper covers: (i) the motivation and design rationale for BrewHaven; (ii) the technical architecture of the customer and admin interfaces; (iii) a comparison with existing alternatives; (iv) a frank assessment of limitations; and (v) proposed future extensions.

II. BACKGROUND AND RELATED WORK

A. Existing Café Ordering Approaches

Digital café ordering solutions currently fall into three broad categories: (1) subscription cloud POS systems (e.g., Square, Petpooja, Lightspeed), which are feature-rich but incur monthly costs and require continuous internet access; (2) custom web applications with server backends, providing full control at the cost of significant development and hosting expenditure; and (3) Progressive Web Apps (PWAs), which can work offline but typically still require a server for crossdevice data synchronisation. BrewHaven occupies a fourth, underexplored category: a zero-server, zero-cost, browser-confined system appropriate for single-device deployments.

B. The localStorage API

localStorage is a browser-native key-value storage mechanism that persists data across page reloads and browser restarts. Data is stored as serialised text (JSON), scoped to the origin of the document. Its principal constraint — that storage is local to a single browser instance on a single device — directly

shapes the architectural trade-offs of BrewHaven [5, 6].

C. Single-Page Application (SPA) Pattern

BrewHaven's customer interface follows the SPA paradigm, in which a single HTML document manages all view transitions through JavaScript-driven DOM manipulation, avoiding full-page reloads [4]. This yields an application-like interaction feel without any build tooling or JavaScript framework dependencies.

III. SYSTEM DESIGN

A. Architectural Overview

The system comprises exactly two files: index.html (customer interface) and admin.html (management dashboard). All CSS and JavaScript are inlined within each file, requiring no external dependencies, package managers, or build processes. Communication between the two interfaces is mediated entirely through localStorage, as illustrated in Fig. 1.

Fig. 1. System communication through localStorage.

B. localStorage Schema

Four storage keys govern the entire system state:

TABLE I — localStorage Key Schema

Key Name	Used By	What It Stores
brewhaven_orders	Both pages	All placed orders
bh_item_avail	Admin writes, Customer reads	Available menu items
bh_admin_sess	Admin only	Admin login session
bh_notifications	Admin only	Dashboard alerts

Table I. The four localStorage keys used by BrewHaven.

C. Customer Interface Screens

The customer page presents four sequential screens, each replacing the previous via JavaScript visibility toggling: (i) a Welcome Screen with café branding; (ii) a Table Selection Screen offering eight table positions plus a takeaway option; (iii) the main Menu Screen with category filtering, full-text search, cart management, and a special notes field; and (iv) a Success Screen displaying an order receipt with a celebratory confetti animation.

IV. CUSTOMER INTERFACE IMPLEMENTATION

A. Menu Structure

The menu catalogue consists of 42 items distributed across five categories, defined as a JavaScript data object rather than HTML markup. This architecture ensures that all rendering is generated programmatically, allowing menu updates without modifying the document structure. Table II summarises the categories

TABLE II — Menu Category Summary

Category	Items	Price Range (INR)
Hot Beverages	10	₹70 – ₹145
Cold Beverages	8	₹100 – ₹175
Bakery & Snacks	9	₹90 – ₹180
Mains	8	₹200 – ₹280
Desserts	7	₹140 – ₹220

Table II. Menu categories with item counts and INR price ranges.

B. Search and Filtering

Customers may filter items either by selecting a category tab or by entering text into the search field. Both mechanisms operate simultaneously: a customer may select 'Cold Beverages' and then search for 'mango' to narrow results to relevant items within that category. All filtering operates on the in-memory JavaScript data object, producing instantaneous updates.

C. Cart and Order Placement

Cart state is maintained in a JavaScript variable. Items added to the cart replace their 'Add' button with an inline quantity stepper. Upon order placement, a JSON order object is appended to the brewhaven_orders array in localStorage and a receipt is rendered on the success screen. The entire transaction is local and instantaneous, with no network latency.

D. Table Occupancy Synchronisation

The admin interface may mark tables as occupied, writing this state to localStorage. On the table selection screen, the customer interface reads this state and visually disables occupied tables, preventing duplicate table assignments [2].

V. ADMIN DASHBOARD IMPLEMENTATION

A. Authentication

Access to admin.html is gated by a login form. Credentials are stored within the JavaScript source and validated client-side; a session flag in localStorage maintains the authenticated state across page reloads. This mechanism is appropriate only when the device is physically secured from customer access.

B. Dashboard Analytics

The dashboard derives four key performance indicators directly from localStorage: total order count, total revenue, most-ordered item, andaverage order value. These are recalculated on each page load, providing the operator with a current operational snapshot without any external analytics infrastructure.

C. Order Status Management

Each order can be transitioned through three states — Preparing, Ready, and Completed — via per-card status buttons in the Orders tab. Status changes are written to localStorage, enabling kitchen and service staff to coordinate workflow.

D. Menu Availability Control

The Menu tab exposes a toggle control for each item. Disabling an item writes a false flag to bh_item_avail. The customer interface reads this flag on each menu render, visually disabling unavailable items and adjusting category item counts accordingly. This addresses a prevalent operational pain point in café management [1].

VI. DESIGN AND VISUAL STYLE

A. Colour and Typography

The interface employs a near-black background with warm amber and gold accent tones. This palette is consistent with research indicating that warm golden tones on dark backgrounds elevate perceived product quality and consumer willingness-to-spend in food-service contexts [9]. Playfair Display is used for headings (conveying premium café aesthetics); Inter is used for body text (optimising on-screen legibility).

B. Performance-Aware Animation

Three ambient visual layers — a particle canvas, glowing orbs, and a noise-texture overlay — are conditionally rendered based on a device performance probe executed at page load. On lower-capability devices, these layers are suppressed while full ordering functionality is retained. This constitutes graceful degradation [7]. The post-order confetti animation similarly reduces particle count on slow devices (100 → 45 shapes).

VII. COMPARATIVE ANALYSIS

BrewHaven occupies a distinct position in the ordering system landscape. Its defining advantages — zero cost, zero internet dependency, immediate deployability — are counterbalanced by the single-device constraint inherent to localStorage-based architectures. This trade-off is appropriate for single-

point ordering deployments, but precludes use cases requiring real-time cross-device data synchronisation. See Section VIII (full-width table on next page) for the detailed comparison.

VIII. LIMITATIONS

A. Single-Device Data Scope localStorage is scoped to the browser instance on a specific device. Consequently, orders placed via index.html on one device are not visible via admin.html on a different device. This is the system's most significant architectural constraint and makes BrewHaven unsuitable for multi-tablet deployments without architectural modification [5].

B. Data Durability
All order history resides exclusively in the browser's localStorage. Clearing browser data results in irreversible data loss. Furthermore, localStorage has an approximate 5–10 MB capacity limit, which at approximately 200 orders per day yields an estimated 250-day operational lifespan before manual clearing becomes necessary.

C. Security Model
Admin credentials are stored in plaintext within the JavaScript source and are accessible via browser developer tools. This model is acceptable for a physically secured, offline device but is unsuitable for any internet-accessible deployment.

D. Absence of Real-Time Updates
Without a server or WebSocket connection, the admin interface does not receive push notifications for new orders. Staff must manually refresh the admin page to observe newly placed orders. Item availability changes made by the admin are similarly not propagated to customers with open menu screens.

E. No Integrated Payment Processing
BrewHaven manages order capture but not payment processing. Postorder payment is handled externally. Integrating a payment gateway (e.g., Razorpay, Stripe) would require a server-side component to securely handle payment verification [3].

IX. FUTURE WORK

A. WebSocket-Enabled Multi-Device Sync
The introduction of a lightweight Node.js server with WebSocket support would resolve the single-device limitation. Real-time order events could be pushed from the customer interface to the admin dashboard and a dedicated kitchen display screen, completing the full operational workflow [3].

B. Progressive Web App Conversion
Packaging BrewHaven as a PWA would enable installation on mobile devices, background push notifications for new orders, and enhanced offline capability — significantly improving staff experience without a full architectural rebuild [4].

C. Cloud Database Integration BrewHaven
Migrating data persistence from localStorage to a cloud database (e.g., Firebase Realtime Database, Supabase) would address both data durability and multi-device access, with both platforms offering free tiers sufficient for small café deployment volumes [6].

D. QR Code Table Detection
Printing unique QR codes at each table, encoding URLs of the form index.html?table=N, would enable automatic table identification at session start, eliminating the manual table selection step and reducing the risk of table misidentification.

E. Kitchen Display Screen
A third HTML file designed as a dedicated kitchen display — showing incoming orders with countdown timers and status controls — would complete the full customer-to-kitchen workflow when paired with a WebSocket server component.

X. CONCLUSION

BrewHaven demonstrates that a functional, polished, and practically useful digital café ordering system can be constructed using only three foundational web technologies — HTML, CSS, and JavaScript — without any server infrastructure, database software, or financial outlay.

The system provides a complete ordering workflow for customers and a practical management interface for operators, constrained only by the inherent limitations of browser-local storage. These limitations are acknowledged and do not impair utility in the most common small café deployment scenario: a single managed tablet.

More broadly, BrewHaven illustrates that modern browser APIs are sufficiently capable to support real-world business tooling, and that the complexity and cost typically associated with digital ordering systems are not fundamental requirements but rather artefacts of architectural choices that can be revisited.

TABLE III — Comparative Analysis of Café Ordering System Options

Feature	BrewHaven	Cloud POS	Custom Web App	Paper System
Monthly Cost	Zero	₹1,500–₹5,000	₹2,000+ hosting	Near zero
Internet Needed?	No	Yes (always)	Yes (for sync)	No
Multi-Device?	No	Yes	Yes	Yes (no sync)
Real-Time Updates?	No	Yes	Yes	No
Offline Working?	Yes (fully)	No/Limited	Partial	Yes
Data Safety	Low (browser)	High (cloud)	High (server)	Low (paper)
Setup Difficulty	Very easy	Easy	Hard	None
Analytics	Basic	Advanced	Custom	None
Security	Basic	Enterprise	Developer-defined	Physical only

Table III. Feature-by-feature comparison of BrewHaven against three alternative ordering approaches

REFERENCES

1. J. Kim and S. Park, "The Impact of Digital Ordering Systems on Restaurant Operations and Customer Satisfaction," J. Foodservice Bus. Res., vol. 23, no. 4, pp. 312–329, 2020.
2. B. Moreira, "Self-Service Kiosk Adoption in Quick Service Restaurant Chains: A Case Study," Int. J. Hosp. Manage., vol. 89, p. 102574, 2020.
3. R. Patel and A. Mehta, "Real-Time Hotel Room Service Ordering Using Firebase and React," in Proc. ICCCNT 2021, IEEE, pp. 1–6, 2021.
4. L. Zhao, W. Chen, and Y. Liu, "QR-Code Triggered Progressive Web App for Restaurant Table Ordering," in Proc. ISCTIS 2020, IEEE, pp. 44–49, 2020.
5. W3C, Web Storage Specification (2nd ed.), World Wide Web Consortium, 2016. [Online]. Available: https://www.w3.org/TR/webstorage/
6. P. Dhawan and M. Kapoor, "Client-Side Storage Strategies for Offline-First Web Applications," ACM SIGWEB Newsl., vol. 2022, no. 1, pp. 1–9, 2022.
7. J. Nielsen, "10 Usability Heuristics for User Interface Design," Nielsen Norman Group, 1994.

8. H. Liu and J. Kwon, "Animated UI Feedback and Error Reduction in Digital Menu Interfaces," Comput. Hum. Behav., vol. 128, p. 107103, 2022.
9. C. Spence, "On the Psychology of Colour and Taste Perception," Flavour J., vol. 4, no. 21, 2015.
10. W. E. Hick, "On the Rate of Gain of Information," Q. J. Exp. Psychol., vol. 4, no. 1, pp. 11–26, 1952.