Convention Center Event Management System – Finalized Requirements Document

1. Purpose

This document defines a **lean, clean, cost-effective, and future-ready Convention Center Event Management System** for **Web and Mobile App**. It enables a **single owner** to manage **multiple convention centers**, track income and expenses, and generate essential financial reports efficiently. The owner has full control across all centers.

---

2. Scope

- Web application (internal use only)
- Mobile application (Android & iOS, internal use only)
- Centralized backend and database
- Multiple convention centers under a single owner
- Optional future public module: read-only calendar per center

Users can **enter and view data from both web and app**, scoped to their assigned center(s).

---

3. User Roles

Admin (Owner / Towner): Manage users, events, income, expenses across centers; access dashboards per center or consolidated.

Staff (User): Enter data and view events for assigned center; no access to other centers or consolidated financial reports.

---

4. Modules & Functional Requirements

> The system is designed as a **white-label product**, allowing the same platform to be sold to different convention centers with their own branding (logo, name, colors) without code changes.

4.1 User & Role Management

- Admin-approved user registration and profile management.
- Role assignment per center (Admin, Staff).
- Secure login/logout and password management.
- Role-based access control for events, income, expenses, and reports.

4.2 Event Management

- Create, edit, and track events: Date, Time, Type, Client Name, Address, Contact, Advance Amount, Total Booked Amount, linked center.
- Event lifecycle: Blocked → Booked → Completed.
- Event status tracking and reporting per center.

4.3 Income & Expense Management

- Record event-related and general expenses, categorized (Food, Decoration, Electricity, Cleaning, Staff, Others).
- Track income received, advance amounts, balance calculations per event and center.
- Expense analytics to identify high-cost categories and event-wise expense comparison.

4.4 Client Management & Analytics

- Automatic client record creation from event entries.
- Client search by name or contact.
- Client history: total events, first/last event date, total revenue, outstanding balances.
- Reports: repeat vs new clients, top revenue-generating clients.

4.5 Reporting & Analytics

- Event-wise income vs expense and profit/loss per center.
- Monthly and annual summaries.
- Charts and graphs for quick insights.
- Export options: PDF and Excel.

4.6 Calendar Module

- Internal monthly/yearly calendar per center.
- Event filtering by center, date, or type.
- Future public read-only calendar with date, type, and status only.

4.7 Notifications & Alerts

- In-app notifications: new events, completed events, added expenses, pending balances.
- Push notifications for mobile app: event reminders, same-day alerts.
- Email notifications: confirmations and summaries.
- Dashboard alerts: over-budget events, high expenses, pending payments.

---

5. Dashboards

Admin: Total Events, Income, Expenses, Net Balance, Pending Payments per center or consolidated.

Staff: Assigned Events, Event Status, Recent Entries per center.

---

6. Non-Functional Requirements

- Secure internal access only
- Fast API responses (< 2s)
- Mobile-friendly, simple forms
- Optimized for staff usage
- Data scoped per center

---

7. Technology Stack (Lean & Future-Ready)

Frontend: React.js (Web), **React Native with Expo (Mobile)**, Material UI / Ant Design, Victory/Recharts, Redux Toolkit/Zustand, React Hook Form + Yup.

Mobile App Notes (Expo): - Built using Expo (managed workflow) - Faster setup on Windows - No mandatory Android Studio or local emulator - Supports push notifications, OTA updates - Can eject to bare React Native later if required

Backend: Laravel PHP 10+, REST API, Sanctum authentication, role-based access, services for events/income/expenses/reports with center scoping, queues and scheduler for automation.

Laravel PHP 10+, REST API, Sanctum authentication, role-based access, services for events/income/expenses/reports with center scoping, queues and scheduler for automation.

Database: PostgreSQL 14+

- Strong relational integrity and ACID compliance
- Better support for complex reporting and analytics
- Optimized for financial calculations and aggregations
- Excellent performance for GROUP BY, window functions, and CTEs
- Native JSON support for future extensibility
- Tables: Centers, Events, Users, User_Centers, Incomes, Expenses, Clients, Expense_Categories, Notifications
- `center_id` foreign key enforced across all operational and financial tables
- Soft deletes and timestamps enabled
- Single currency system

White-Label / Branding Support (Built-in): - Center-level configuration for: - Logo - Center name - Primary color / theme - Branding loaded dynamically based on logged-in center - Same application binary used for all customers - No code change required for branding updates

Hosting & Infrastructure: - Cloud-based or VPS hosting (AWS EC2 / DigitalOcean / similar) for internal apps. - Managed MySQL instance with automated daily backups. - Scalable for future multi-center growth. - Optional object storage for documents/images. - SSL/TLS, firewall rules, restricted admin access, basic CI/CD pipeline.

Notifications & Communication: Internal email, in-app, push, and dashboard alerts per center; future public calendar read-only; voice calls, IVR, and social media notifications excluded.

---

8. Assumptions

- Single owner managing multiple centers
- Internal users only
- Single currency system
- Internet connectivity available
- Hosting will be cloud-based or VPS with security and backup measures in place

---

9. Future Enhancements (Out of Scope)

- GST / Tax calculations
- Vendor management
- Accounting system integration
- Public booking requests and lead management

---

10. Database Schema (Proposed – Lean & Scalable)

The database is designed for **single owner + multiple centers**, **white-label branding**, and **clear financial reporting**, while keeping joins and complexity minimal.

10.1 centers

Stores convention center and white-label configuration.

- id (PK)
- name
- logo_url
- primary_color
- address
- contact_phone
- status (active / inactive)
- created_at
- updated_at

10.2 users

System users (Admin or Staff).

- id (PK)
- name
- email (unique)
- phone
- password
- role (admin / staff)
- status (active / inactive)
- created_at
- updated_at

10.3 user_centers

Maps users to centers (supports one user working in multiple centers if needed later).

- id (PK)
- user_id (FK → users.id)
- center_id (FK → centers.id)
- created_at

10.4 clients

Auto-created from event entries; reused for analytics.

- id (PK)
- center_id (FK → centers.id)
- name
- phone
- address
- created_at
- updated_at

10.5 events

Core event booking table.

- id (PK)
- center_id (FK → centers.id)
- client_id (FK → clients.id)
- event_date

- start_time
- end_time
- event_type
- status (blocked / booked / completed / cancelled)
- advance_amount
- booked_amount
- created_by (FK → users.id)
- created_at
- updated_at

10.6 incomes

Tracks money received for events.

- id (PK)
- center_id (FK → centers.id)
- event_id (FK → events.id)
- amount_received
- received_date
- payment_mode (cash / bank / UPI / other)
- created_by (FK → users.id)
- created_at

10.7 expense_categories

Master table for expense grouping.

- id (PK)
- name (Food, Decoration, Electricity, etc.)
- created_at

10.8 expenses

Tracks event-level and general expenses.

- id (PK)
- center_id (FK → centers.id)
- event_id (nullable, FK → events.id)
- category_id (FK → expense_categories.id)
- amount
- expense_date
- description
- created_by (FK → users.id)

- created_at

10.9 notifications (internal)

Stores in-app notification records.

- id (PK)
- center_id (FK → centers.id)
- user_id (FK → users.id)
- type (event / expense / payment)
- message
- is_read (boolean)
- created_at

---

11. Key Design Notes

- `center_id` exists in all financial and operational tables for **strict data isolation**.
- White-label branding is driven entirely from the **centers** table.
- No duplicate financial data is stored; reports are calculated.
- Schema supports scaling to SaaS without restructuring.
- Optimized for MySQL with indexes on:
- center_id
- event_date
- client phone
- status
