**REQUIREMENT SPECIFICATION DOCUMENT  
FOR THE DEVELOPMENT OF AN EVENT VENUE BOOKING SYSTEM**

A Project Proposal Presented to the

Faculty of Datamex College of Saint Adeline, Inc.

In Partial Fulfillment of the Requirements for the  
Degree of Bachelor of Science in Information Technology

By:

Albania, John Maverick B.

September 2025

**REQUIREMENT SPECIFICATION DOCUMENT**

**INTRODUCTION**

The purpose of this document is to define the requirements for the Event Venue Booking System. It serves as a clear reference for all stakeholders, including the project owner, client representatives, and technical advisers. This ensures that the system will be developed with the agreed functionalities and limitations in mind. The document will also be the basis for the design, development, testing, and future maintenance of the system.

The Event Venue Booking System is a modern web-based system that simplifies client booking for events. Clients can visit the website to view available venues and submit booking requests online. The system has a real-time availability checker to prevent double bookings, a centralized booking calendar for staff and administrators, and an automated email notification system to keep clients informed about their reservation status. By digitizing the booking process, the system speeds up reservations, increases accuracy, and makes booking more convenient. This helps reduce scheduling conflicts and improve overall client satisfaction.

This requirements specification outlines the functional and non-functional needs of the Event Venue Booking System. It describes the booking process for clients, management features for administrators and staff, data requirements, assumptions, and limitations. Clients can submit booking requests online without creating an account, All they need is a Google account. Meanwhile, administrators and staff handle approvals, scheduling, and reporting on the backend. Payment gateway integration is not included in the initial release, but it may be considered for future system upgrades.

**FUNCTIONAL REQUIREMENTS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Requirement ID** | **Requirement Description** | **Priority** | **Dependencies** | **Acceptance Criteria** |
| JM-01 | Clients can access the website and navigate to the venue booking page. | High | None | Venue booking page loads and displays venue options. |
| JM-02 | Clients can view available venues with details capacity, description, availability. | High | JM-01 | Venue details are displayed and updated in real time. |
| JM-03 | Clients can submit a booking request by filling in their details name, email, contact, event date/time, event type. | High | JM-02 | Booking request is stored in the system and marked as “Pending.” |
| JM-04 | Clients receive email notifications about the status of their booking (Pending, Approved, Rejected). | Medium | JM-03 | Clients receive confirmation email after admin/staff action. |
| JM-05 | Admin can log in with full system access. | High | None | Admin is redirected to admin dashboard upon successful login. |
| JM-06 | Admin can approve, reject, or cancel booking requests. | High | JM-03 | Booking status changes correctly and notification is sent to the client. |
| JM-07 | Admin can manage venue information (add, edit, update availability, delete). | High | JM-05 | Venue list updates are reflected on the booking page. |
| JM-08 | Admin can manage staff accounts (add, edit, deactivate). | Medium | JM-05 | Staff account changes are saved in the system. |
| JM-09 | Staff can log in with limited system access. | High | JM-08 | Staff is redirected to staff dashboard after login. |
| JM-10 | Staff can view booking requests and the booking calendar. | Medium | JM-09, JM-03 | Booking requests and calendar display correctly. |
| JM-11 | Staff can update limited booking details (e.g., event notes, minor changes). | High | JM-10 | Booking details update is saved and visible. |
| JM-12 | The system provides a centralized booking calendar accessible by Admin and Staff. | Medium | JM-03 | Calendar reflects updated booking information. |
| JM-13 | The system generates simple reports on venue usage and bookings. | High | JM-12 | Reports are generated and exportable. |
| JM-14 | The system prevents double booking by checking venue availability in real time. | High | JM-03 | If a venue is already reserved for the same slot, system blocks the booking. |

**Table 1: System Functionality**

**NON-FUNCTIONAL REQUIREMENTS**

**Performance**

* The system must display available venues within 2 seconds after the client loads the booking page.
* The system must process booking requests and update availability in real time to avoid double booking.

**Usability**

* The client interface must be simple and intuitive, allowing users to submit a booking request in three steps or less Select Venue Fill Form Submit.
* Venue information should be presented clearly, including capacity, description, and availability status.
* Automated email notifications must be formatted clearly with booking details and status.

**Reliability**

* Booking requests and approvals must be logged reliably in the database with no data loss.
* Backup of booking records should occur daily to ensure recovery in case of failures.

**Security**

* All sensitive data (such as client email and contact number) must be stored securely and encrypted where necessary.
* Admin and staff access must be protected by authentication (username and password).
* Unauthorized access to booking management functions must be prevented through role-based access control.

**Scalability**

* The system must be able to handle at least 100 simultaneous booking requests without significant performance degradation.
* It should be designed to allow easy integration of additional features in the future (e.g., payment gateway, public client portal with accounts).

**Maintainability**

* The system code must be modular and documented to allow updates and fixes with minimal downtime.
* System logs should track errors and key events to support maintenance and troubleshooting.
* The architecture should support future enhancements like SMS notifications or external venue integrations.

**USE CASES**

**UC-01:** Client Submits Booking Request In this use case, the client visits the website and goes to the booking page. The client chooses an available venue, then fills in their personal details like name, email, contact number, and event information. After completing the form, they submit their booking request. The system then saves the request and marks it as “Pending.” If the chosen venue is already booked, the system will show a message saying “Venue Unavailable.”

**UC-02:** Admin Approves or Rejects Booking  
 Here, the admin checks the booking requests made by clients. Before doing so, the admin needs to log in first. After reviewing the details, the admin can either approve or reject the request. Once done, the system updates the booking status and automatically sends an email to the client about the result. If there’s a scheduling conflict or double booking, the system won’t allow the admin to approve until it’s fixed.

**UC-03:** Staff Views Booking Calendar In this use case, the staff logs in to the system to view the booking calendar. The calendar shows all ongoing and upcoming bookings, making it easier to monitor the schedule. If some booking details are missing or need small updates, the staff can add notes or minor information to keep the records complete.

**UC-04:** Admin Manages Venues This use cases explains how the admin handles venue information. After logging in, the admin can add new venues, edit existing ones, or remove those that are no longer available. Any updates made by the admin will automatically appear on the client booking page. If there’s missing or wrong information, the system will ask the admin to correct it before saving.

**UC-05:** Client Receives Booking Confirmation Email After the client submits a booking request, the system automatically sends an email with the booking details and its current status whether it’s Approved, Rejected, or still Pending. This keeps the client informed about their request. If the email doesn’t go through for some reason, the system logs the error and notifies the admin to resend it.

**DATA REQUIREMENTS**

**Data Entities**

**Client** - users who will access the website to make venue bookings.

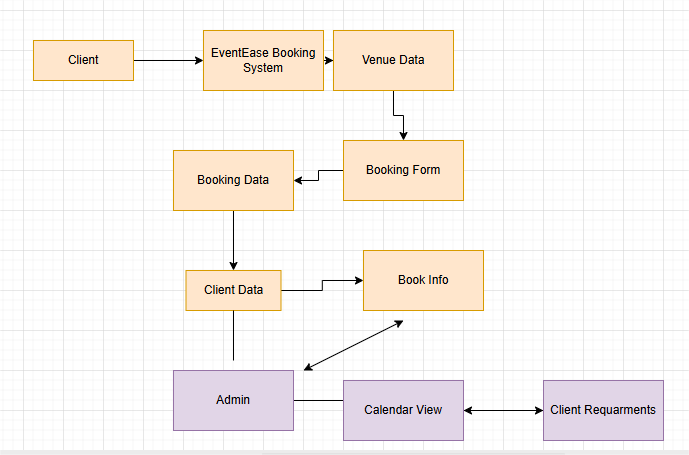
**Admin** - has full control over the system, including approving/rejecting bookings, managing venues, and managing staff accounts.

**Staff** - has limited access, mainly to assist in booking management and monitoring.

**Venue** - the list of event venues available for reservation (e.g., halls and function rooms).

**Booking** - the reservation record created by clients and managed by admin or staff.

**Attributes**

****

**Images 1:** Data Flow

**ASSUMPTIONS AND CONSTRAINTS**

**Assumptions**

* Staff will be trained to use the system.
* The system will be used only for in-house venues.

**Constraints**

* No payment gateway integration in Phase 1.
* No external venue integration.

**GLOSSARY**

**Double Booking**: Two or more events scheduled for the same venue at the same time.

**Admin Dashboard**: Section of the system where admin users manage system settings and accounts.

**Centralized Calendar**: Shared calendar displaying all confirmed reservations.

**REVISION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Description | Author |
| 1.0 | 2025-08-26 | Initial draft based on project proposal | John Maverick B. Albania |

**Table 2: System version**

**APPENDIX**

Sample database schema Venues, Bookings, Users

Wireframe sketches Reservation form, Calendar, Admin Dashboard

References from existing event booking systems and agile methodology articles