SELF INTENSIVE TRAINING ON FULL STACK

DEVELOPMENT Stage -1

(Planning And Requirement Gathering)

Name	SABTHAGIRI E K S
Roll No	7376221EC284
Seat No	230
Project ID	12
Module Name	VENUE MANAGEMENT

TECHNICAL COMPONENTS:

COMPONENT	**MEVN STACK**
Frontend	Vue (JS Library for building user interfaces)
Backend	Node.js with Express.js
Database	MongoDB (NOSQL Database)
API	Open API

1. Introduction

1.1. Purpose:

The goal of this document is to provide a full account of the Venue Booking. It will describe the system's goal and characteristics, its interfaces, what the system will perform, the limitations under which it must function, and how the system will respond to external stimuli.

1.2. Scope of Project:

- Students or faculties will be able to request venues to the infra team through this software system, which will act as a portal for them. .
- Administrators possess the authority to accept or decline Venues. Once a location has been authorized, collisions must be avoided.If a different user needs the same. Venue at the same time and date that the request is given to the admin, who can then approve or reject it. The admin can also modify the venue and provide feedback on user requests. After logging in, the user can also view their history.

2. System Overview:

2.1. Users:

1. **Students**:

They can upload necessary information, track the progress of their application, and examine their past venue bookings in addition to submitting applications for venue reservations.

2. Admins:

Review submitted venue booking applications, approve or reject applications (with remarks),

2.2. Features:

Venue Registration

1. Users have the ability to register new locations with information like name, address, capacity, and amenities.

Venue Booking

1.Users can book available venues for specific dates and time slots.

2.The system should prevent double booking of a venue for the same date and time.

3.If a user attempts to book an already reserved venue, the request is sent to the administrator for review.

Administrator Review

1.Administrators can review booking requests for venues that have conflicts.

2.Administrators have the option to approve or deny the conflicting booking request.

3.Administrators can change the venue assigned to a booking request and provide comments or remarks.

Notifications

Users should receive notifications regarding the status of their venue registration and booking requests (approved, denied, or modified by the administrator).

User Management

1.The system should have user authentication and authorization mechanisms.

2.Users can be assigned different roles (e.g., regular user, administrator) with varying levels of access and permissions.

Reporting and Analytics

1.The system should provide reporting and analytics features for administrators to monitor venue utilization, booking trends, and other relevant metrics.

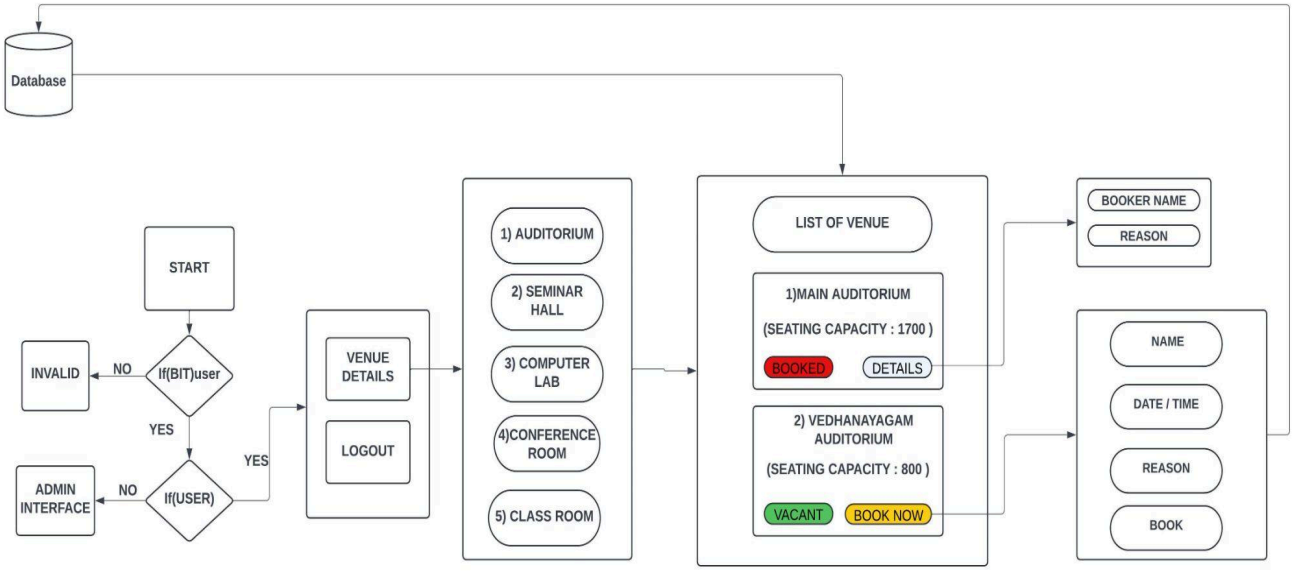
User Interface

1.The system should have a user-friendly interface for users to register venues, book venues, and view their booking history.

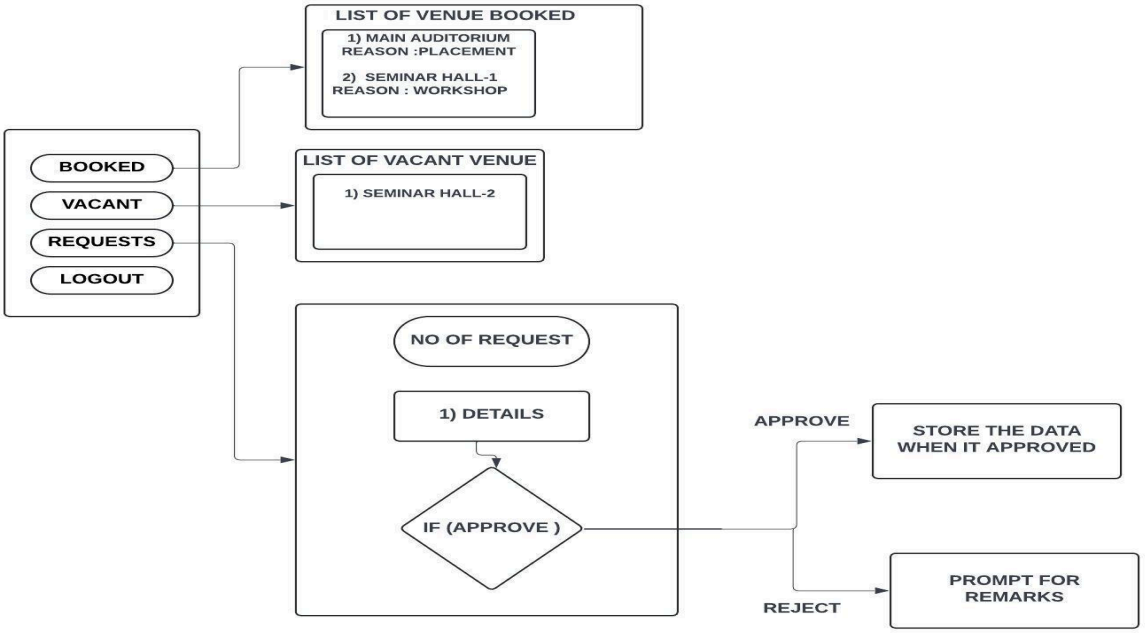
2.Administrators should have a separate interface or dashboard to manage venues, review booking requests, and access reporting and analytics features.

FLOW CHART :

USER INTERFACE



ADMIN INTERFACE



3. System Requirements Specification:

3.1 Functional Requirements:

User Management

1.User registration and authentication (login/logout)

2.User roles (admin, regular user) with different permissions

Venue Management

1.Venue registration by users (with details like name, location, capacity, amenities)

2.Venue approval/rejection by administrators

Booking Management

1.Book available venues for specific dates and time slots

Administrator Functions

1.Review conflicting booking requests

2.Approve, deny, or modify (change venue, add comments) conflicting bookings

Notification Management

1.Notify users about venue registration status (approved/rejected)

2.Notify users about booking request status (approved/denied/modified)

3.2. Non-Functional Requirements:

● **Performance:** The system must respond to user actions within 2 seconds to ensure efficient usability and must handle a concurrent user load of at least 100 users without significant performance degradation.

● **Security:** User data must be encrypted during transmission and storage, and access to sensitive functionalities should be restricted to authorized admin users through secure authentication mechanisms.

● **Usability:** The user interface should be intuitive and user-friendly, with clear and concise error messages provided to guide users in case of input errors or system failures.

● **Reliability:** The system should be available 24/7 with minimal downtime and should have a backup and recovery mechanism in place to prevent data loss in case of system failures or crashes.

● **Scalability:** The system should be designed to accommodate an increasing number of users and data volume over time, and it should be scalable to support additional features and functionalities as per future requirements.