SUMMER TRAINING REPORT

Aerial Delivery Research and Development Establishment (ADRDE)

Under the guidance of

Sh. Hotam Bansal (Scientist ‘E’)

Sh. Satyasheel (Technical Officer ‘A’)

Miss. Kriti Mittal (S. T.A ‘B’)

Department

Para 9 (Information Technology)

Submitted By:

Frontend Team:	**Backend Team:**	**Database Team:**
Alina Ali	Diya Gupta	Aditya Kumar
Kamil Khan	Naina Jain	Rishabh Yadav
Mehak Kushwah	Aniket Goyal	Ravi Singh
Jyoti Upadhyay		Gaurav

From B.tech-CSE (Eshan College of Engineering)

Duration: 19 June 2025 to 31 July 2025

Aerial Delivery Research and Development Establishment (ADRDE), Defense Research & Development Organisation (DRDO), Government of India

Agra Cantt-282001

Acknowledgment

We would like to express our sincere gratitude to **ADRDE, DRDO** for providing us with the opportunity to undertake this internship project on the *Venue Booking System*. It has been a valuable and enriching learning experience.

We extend our heartfelt thanks to our mentors and supervisors for their continuous guidance, constructive feedback, and encouragement throughout the development process. Their expertise and support played a pivotal role in the successful completion of this project.

We also acknowledge the contribution of our fellow team members, whose cooperation, dedication, and collaborative effort made this project possible.

Lastly, we thank all the users and stakeholders who provided valuable input during testing and development. Their feedback helped us shape the system to meet practical needs effectively.

This project has helped us enhance our technical skills, strengthen our teamwork abilities, and gain deeper insights into real-world software development.

Declaration

We hereby declare that the project titled **“Venue Booking System for ADRDE, DRDO”** is the outcome of our own efforts and has been carried out as part of our internship assignment.

This project has not been submitted, in part or full, for any other academic purpose or publication elsewhere. The information provided in this report is true to the best of our knowledge and has been prepared based on genuine research, development, and teamwork during the internship period.

We have acknowledged all sources of information and support that have contributed to the successful completion of this project.

Place: Agra Cantt

Date: __________

Certificate

This is to certify that the project report titled
“Venue Booking System for ADRDE, DRDO”
has been successfully completed by the project team as part of their internship program.

The project was carried out under the supervision and guidance of the concerned mentors, and it demonstrates a comprehensive understanding of full-stack development, including frontend design, backend implementation, and database integration. The work done is original and fulfills the requirements of the internship project.

The team has shown commendable effort, collaboration, and professionalism throughout the development cycle of the project.

Internship Coordinator

Name: Sh. Hotam Bansal
Department: Para 9

Signature: ___________

Project Guide/ Supervisor:

Name: Miss. Kriti Mittal
Department: Para 9

Signature: _____________

Table Of Content

1. Introduction

The **Venue Booking System** is a modern, structured application developed to streamline the process of reserving meeting venues at **ADRDE, DRDO**. This system addresses the frequent scheduling and coordination challenges that arise in institutional settings where multiple departments share limited physical and technical resources.

This report outlines the complete life cycle of the project—from planning and development to implementation and testing—providing insights into the technical architecture, team structure, challenges faced, and the final outcomes.

The Venue Booking System is built with a focus on:

- **Efficiency** in handling venue requests and approvals
- **Transparency** in communication among users, directors, and the IT team
- **Scalability** to support multiple departments and use cases
- **User-centric design** for ease of use across roles

1.1 Background

In organizations such as ADRDE, where meetings, presentations, and events are routine and involve diverse stakeholders, a robust venue booking system is essential. Previously, manual processes led to frequent miscommunication, double bookings, and logistical delays. This project was initiated to digitize and simplify the workflow, ensuring smooth coordination and resource allocation.

1.2 Objectives

The primary objectives of the Venue Booking System are:

- To develop a user-friendly web application that allows seamless booking of venues.

- To implement a multi-tier approval mechanism for validation by directors and the IT department.
- To facilitate real-time venue availability checks and feedback submission.
- To ensure proper management of IT infrastructure (e.g., projectors, microphones, VC setup) during events.
- To establish a centralized and auditable record of all bookings and resource usage

2. Team Structure

The project team consists of 11 members divided into three specialized sub-teams:

Team Member and their Role’s

Front-end Team:

- Alina Ali (Front-end Designer & Developer)
- Kamil Khan (Front-end Developer)
- Mehak Kushwah (Documentation)
- Jyoti Upadhyay (Resource Gatherer)

Back-end Team:

- Diya Gupta (Backend API Developer)
- Naina Jain (Admin Panel & Testing)
- Aniket Goyal (Version Control)

Database Team:

- Aditya Kumar (Database & Models)
- Rishabh Yadav (Authentication & Security)
- Ravi Singh (Media Upload & Management)
- Gaurav (Scheduling & Calendar Integration)

3. Project Overview

The **Venue Booking System** is a centralized web application developed to streamline the process of booking meeting spaces within ADRDE, DRDO. It enables users to request, track, and manage venue bookings efficiently while incorporating a multi-level approval mechanism involving Directors and the IT Team. The system ensures real-time availability checks, structured workflows, and improved coordination for hosting organizational meetings and events.

Key Functionalities

- **Booking Requests**:
  Users can raise venue booking requests by selecting available time slots and required facilities.
- **Availability Check**:
  Before submitting a booking, users can view the live availability status of venues to avoid scheduling conflicts.
- **Approval Workflow**:
  The system integrates a multi-tiered approval process where booking requests are sequentially reviewed by Directors and confirmed by the IT team.
- **Event Execution Support**:
  Once approved, the IT team prepares the venue with necessary infrastructure to ensure smooth execution of events.
- **Post-Event Feedback**:
  After the event, users can submit feedback, allowing the organization to maintain quality and address any issues.

3.1 Use Cases

- **Use Case 1: Booking by User**
  A user logs in, checks venue availability, fills out the booking form, and submits a request.

- **Use Case 2: Director’s Approval**
  The Director receives the request, verifies the event details, and either approves or rejects it.
- **Use Case 3: IT Team Confirmation**
  Upon Director approval, the IT team is notified to finalize arrangements and confirm the booking.

4. Technologies Used

4.1 Frontend Technologies

- **Framework**: React with Vite
  React is a leading JavaScript library for building dynamic user interfaces. Vite offers a lightning-fast development server and optimized build process.
- **Languages**: JavaScript, HTML
  JavaScript drives the functionality of the application, while HTML defines its structure.
- **Styling**: Tailwind CSS
  A utility-first CSS framework that enables rapid UI development with responsive design and custom styling.
- **Libraries & Dependencies**:
  - **react-router-dom**: Manages routing and navigation between different pages.
  - **axios**: Handles HTTP requests to communicate with the backend API.
  - **react-icons**: Provides a wide range of icons to enhance UI/UX.

4.2 Backend Technologies

- **Framework**: Django
  A high-level Python framework promoting clean, pragmatic design and rapid development of secure web applications.
- **API Development**: Django REST Framework
  Used for building robust RESTful APIs to connect the frontend with backend services.
- **Version Control**: Git
  A distributed version control system for tracking and managing changes in the source code during collaborative development.

4.3 Database Technologies

- **Database System**: PostgreSQL
  PostgreSQL is a powerful, open-source, object-relational database system known for its reliability, scalability, and SQL compliance.
- **ORM**: Django Models
  Django's built-in ORM (Object Relational Mapper) simplifies database interactions, allowing developers to manage database schema using Python classes.
- **Schema Management**: Django Migrations
  Automatic database schema updates and versioning through migration files.
- **Relational Structure**:
  The system employs a relational database model, with key tables such as:
  - **User Table**: Stores user credentials and roles.
  - **Venue Table**: Captures information about available venues, their location, and capacity.
  - **Booking Table**: Manages booking details, including time slots, venue IDs, and user requests.
  - **Feedback Table**: Records post-event feedback submitted by users.
- **Data Integrity**:
  Constraints and relationships are enforced through foreign keys and model validations to ensure consistency and accuracy of data.

5. Functionalities

The **Venue Booking System** is built to cater to the unique needs of three key roles within ADRDE, DRDO: Users, Directors, and the IT Team. Each role has access to specific functionalities designed to simplify their responsibilities while maintaining a smooth workflow from booking request to event execution.

5.1 User Role (Sector Employees)

Users are the primary initiators of booking requests. Their functionalities include:

- **Venue Booking**
  Users can browse available venues and submit booking requests by selecting dates, times, and facilities.
- **Availability View**
  Users can view real-time availability of venues to avoid overlaps and conflicts.
- **Booking History**
  A log of past bookings is available for reference and accountability.
- **Feedback Submission**
  After using a venue, users can rate their experience and provide suggestions for improvement.
- **Contact IT Support**
  Users can reach out to the IT team for any technical assistance related to venue infrastructure.

5.2 Director Role

Directors act as the approval authority for venue booking requests. Their responsibilities include:

- **Request Review and Approval**
  Directors can view all incoming booking requests and approve or reject them based on relevance and availability.
- **Request Editing**
  Directors may make changes to booking details before approving a request.
- **Communication with IT Team**
  Directors can coordinate with the IT team to discuss specific event requirements.
- **View Booking History**
  A historical record of all approved or rejected requests is available for audit and planning purposes.

5.3 IT Team Role

The IT Team is responsible for managing approved bookings and ensuring venues are technically equipped. Their features include:

- **Request Management**
  View all approved requests and prepare the venue with required technical resources (AV setup, networking, etc.).
- **User Management**
  Approve or deny user registrations to maintain secure access to the system.
- **Event Logistics Coordination**
  Ensure that all resources (projectors, microphones, internet) are available at the time of the event.
- **Reports & Feedback Access**
  Generate and review usage reports, feedback summaries, and historical data for system improvement.

- **Full Booking History**
  Track all transactions, bookings, and communications for reference and troubleshooting.

6. Purpose

The purpose of the **Venue Booking System** is to provide a **centralized, efficient, and transparent** platform for managing venue reservations and associated IT support at ADRDE, DRDO.

This system eliminates the need for manual coordination by:

- Allowing users to check real-time availability and request bookings.
- Implementing a **multi-level approval workflow** to ensure all bookings are properly authorized.
- Enabling the **IT team to coordinate technical support and logistics** seamlessly.
- Maintaining a record of all bookings, approvals, and feedback for accountability.

Through automation and structured communication, the system **reduces scheduling conflicts**, improves inter-departmental coordination, and ensures **optimal resource utilization**.

7. Backend Development

The backend of the Venue Booking System is built using **Django** with **Django REST Framework**, ensuring robust API handling and smooth integration with the frontend.

7.1 API Development

- Built RESTful APIs to manage bookings, venues, users, and feedback.
- Key endpoints:
  - /api/bookings/: Create/view bookings.
  - /api/venues/: Check venue availability.
  - /api/feedback/: Submit feedback.
- Used **JWT** for secure authentication.

7.2 Database Development

- **Database**: PostgreSQL.
- Used Django ORM for model definitions:
  - Venue, Booking, Feedback, and extended User model.
- Schema managed through Django migrations.

7.3 Admin Panel & Testing

- Django Admin used to manage venues, users, and feedback.
- Manual testing with **Postman** and Python requests for API validation.

7.4 Version Control & Docs

- Project documented with **README** and **Swagger** for API reference.

8. Database Development

8.1 Models and Schema

The database structure was designed using Django’s ORM (Object Relational Mapper), ensuring a consistent and scalable schema.

Key Models:

- **User**:
  Extended to support multiple roles (User, Director, IT Team).
- **Venue**:
  Contains information like name, location, capacity, and current availability.
- **Booking**:
  Links users to venues with fields like date, time, purpose, and booking status.
- **Feedback**:
  Stores user feedback including comments and ratings for events.

Schema Relationships:

- One-to-Many: A venue can have many bookings.
- One-to-One: Each booking may have one feedback entry.
- Foreign Keys: Used to establish relationships between models.

8.2 Migrations

Django migrations are used to apply schema changes to the PostgreSQL database.

- **Initial Migrations**: Created after defining models for the first time.
- **Subsequent Migrations**: Handled any modifications to model fields or relationships.
- **Commands Used**:

  - python manage.py makemigrations
  - python manage.py migrate

These ensure version-controlled and reliable updates to the database structure.

8.3 Data Handling

Efficient data handling was achieved through Django’s model methods and QuerySet API.

- **Data Validation**: Performed at the model level using field validators.
- **CRUD Operations**: Django’s built-in functions were used to create, read, update, and delete records.
- **Admin Panel Integration**: Allowed easy manipulation of data for testing and admin use.
- **Security**: Role-based access controls prevent unauthorized data manipulation.

9. Frontend Development

9.1 User Interface Design

The frontend was built using **React** with **Vite**, optimized for performance and modular design. The team focused on creating clean, intuitive layouts using **Tailwind CSS**, ensuring consistency and visual clarity.

Key Screens Designed:

- **Login & Registration Page**
- **Venue Availability Checker**
- **Booking Form**
- **Director Approval Dashboard**
- **Feedback Submission Page**
- **Admin Panel (for IT Team)**

Wireframes were created prior to development to define layouts and element positioning, improving alignment with user expectations.

9.2 User Experience Considerations

To ensure a smooth user experience:

- **Clear Navigation**: Implemented using react-router-dom for multi-page routing.
- **Form Validations**: Real-time validation feedback using React hooks.
- **Status Notifications**: Visual feedback provided through modals and toast alerts after user actions like submitting a booking or receiving approval.
- **Accessibility**: Semantic HTML and readable font sizes were used to improve accessibility across devices.

9.3 Responsive Design

The system was designed to function across various devices, from desktops to smartphones.

- **Tailwind CSS** breakpoints were used to create fluid layouts.
- Responsive testing was conducted on:
    - Mobile (Android, iOS)
    - Tablet (iPad, Chrome DevTools)
    - Desktop (multiple resolutions)
- Layouts were adjusted to maintain usability on small screens, especially for the booking and approval dashboards.

10. Challenges Faced

10.1 Technical Challenges

- **Frontend-Backend Integration**: Synchronizing data flow between React and Django APIs required careful handling of asynchronous requests and CORS configuration.
- **Authentication Handling**: Securing login and session management across roles (user, director, IT team) was complex and required token-based authentication.
- **Database Consistency**: Ensuring consistent state between booking requests, approvals, and feedback demanded precise model relationships and validations.
- **Real-Time Availability Check**: Developing a system that checks for up-to-date venue availability during booking involved optimizing API calls and handling race conditions.

10.2 Collaboration & Communication

- **Merge Conflicts**: As multiple developers worked on shared branches, Git merge conflicts were frequent and required coordination.
- **Task Management**: Coordinating tasks across frontend, backend, and database teams took effort, especially during cross-functional features like booking workflows.
- **Testing Coordination**: Aligning test cases across teams to ensure end-to-end reliability was time-consuming.

10.3 Design & Usability

- **UI Consistency**: Maintaining a uniform look and feel across screens required repeated iteration and CSS refactoring.
- **Responsive Design**: Tailoring the app for various screen sizes involved frequent testing and layout adjustments.

11. Future Enhancements

11.1 Feature Improvements

- **Calendar Integration**
  Integrate the booking system with popular calendar services (like Google Calendar or Outlook) to allow automatic scheduling, reminders, and synchronization of events.
- **Notification System**
  Implement email and in-app notifications to alert users, directors, and IT personnel about booking status updates, approvals, and event reminders.
- **Booking Cancellation/Modification**
  Enable users to cancel or modify their booking requests with approval workflows, improving flexibility.
- **Multi-Venue Booking**
  Add support for booking multiple venues in a single request to handle large-scale or parallel events.

11.2 Technical Enhancements

- **Role-Based Access Enhancements**
  Expand role permissions to allow more granular access control, such as department-wise access to specific venues.
- **Real-Time Data Updates**
  Use WebSockets or similar technologies for live updates on booking status and venue availability.
- **Advanced Analytics Dashboard**
  Develop a reporting module to visualize booking trends, user engagement, and feedback insights for management.

11.3 Scalability and Performance

- **Load Handling**
  Optimize the backend and database to handle a larger number of concurrent users and requests as the organization scales.
- **Cloud Deployment**
  Move the application to a cloud platform (like AWS or Azure) for better scalability, reliability, and maintenance.
- **Mobile Application**
  Create a mobile version of the platform to improve accessibility and user convenience.

12. Conclusion

The **Venue Booking System** effectively addresses the complexities of managing meeting spaces in a structured and efficient manner within ADRDE, DRDO. By integrating a multi-tier approval workflow and role-specific functionalities, the system reduces scheduling conflicts, enhances transparency, and streamlines communication between users, directors, and the IT team.

The project showcases how modern technologies such as **React**, **Django**, and **PostgreSQL** can be leveraged to create scalable, user-friendly, and robust applications. The system not only simplifies the booking process but also ensures data consistency and administrative oversight through secure access, comprehensive logs, and feedback mechanisms.

While the current version of the system fulfills the core requirements, the roadmap for future development opens opportunities for greater automation, intelligent analytics, and real-time engagement, setting a strong foundation for continuous improvement and adoption at a larger organizational scale.

THANK YOU!!!