EVENT REGISTRATION LAYOUT

Minor Project Report Submitted to

SRI PADMAVATI MAHILA VISVAVIDYALAYAM

In Partial fulfilment of the requirement for the

MASTER OF COMPUTER APPLICATIONS

III SEMESTER

By

T. SAGARIKA(2024MCA16097)

SHAIK LEKHA BEGUM (2024MCA16089)

Under the guidance of

D. NIKITHA RAO

Accredited by **NAAC with "A+"** Grade ISO 9001: 2015 Certified

DEPARTMENT OF COMPUTER SCIENCE

SRI PADMAVATI MAHILA VISVAVIDYALAYAM

(Women's University)

Tirupati-517502(A.P), Andhra Pradesh

DECEMBER, 2025

DEPARTMENT OF COMPUTER SCIENCE

SRI PADMAVATI MAHILA VISVAVIDYALAYAM

(Women's University)

Tirupati – 517501, Andhra Pradesh, India

Accredited with" A+" Grade by NAAC

CERTIFICATE

This is to certify that the project work entitled "**EVENT REGISTRATION LAYOUT"** is a bonafide record of work carried out by **T.SAGARIKA(2024MCA16097) and SHAIK LEKHA BEGUM(2024MCA16089)** in the **Department of Computer Science**, **Sri Padmavati Mahila Visvavidyalayam**, Tirupati in partial fulfilment of the requirements of III Semester of **MASTER OCOMPUTER APPLICATIONS**. The content of the Project Report has not been submitted to any other University / Institute for the award of any degree.

Guide ***Head of the Department***

DECLARATION

We hereby declare that MCA III Semester Minor Project entitled **"EVENT REGISTRATION LAYOUT"** was done at the **Department of Computer Science**, **Sri Padmavati Mahila Visvavidyalayam**, Tirupati, in the year 2025-2026 under the guidance of **D. NIKITHA RAO** in partial fulfilment of requirements of MCA III Semester.

We also declare that this project is our original contribution of the best of our knowledge and belief. We further declare that this work has not been submitted for the award of any other degree of this or any other university/Institution.

Signature of the Student(s)

ACKNOWLEDGEMENT

We are greatly indebted to our guide **D. NIKITHA RAO** for taking keen interest on my project work and providing valuable suggestions in all the possible areas of improvement.

We express my sincere thanks to the teaching staff of the Department of Computer Science for extending support and encouragement to me in all the stages of the project work.

We gratefully acknowledge and express our gratitude to the non-teaching staff of the Computer Science Department who supported us in preparing the project report.

Signature of the Student(s)

INDEX

SL.No. content Page No.

ABSTRACT

This project presents a complete **Event Registration Layout** designed to simplify the workflow of event organizers and enhance the user experience for participants. The system allows users to register for an event by entering their personal details, selecting event type, venue, duration, and number of attendees. A dynamic real-time **cost calculation module** instantly computes total payable amounts based on user selections.

A key feature of this system is the integration of **UPI-based digital payments** using an automatically generated QR code built through **QRCode.js**. The QR updates instantly whenever the event details or payable amount changes, ensuring accurate and seamless payment processing. The system also supports multiple payment methods, including UPI, credit card, debit card, and cash on delivery.

Upon successful registration and submission of the form, the system generates a **digital receipt** with all event and participant details, which the user can download. Additionally, an **Admin Dashboard** displays all registered users' details, including personal information, event selections, total cost, and timestamp—allowing efficient tracking and event management.

Overall, this project demonstrates a clean, user-friendly, and responsive interface combined with automated calculations, QR generation, and detailed dashboards, making it a complete and functional solution for modern event management requirements.

1. INTRODUCTION

1.2. University Profile

Sri Padmavati Mahila Visvavidyalayam (university for women) was founded in the year 1983 by N.T. Rama Rao, the Chief Minister of Andhra Pradesh, with the fervent desire to train women students as better builders of nation and to include skills of leadership in all aspects of life. The University was established under the Sri Padmavati Mahila Visvavidyalayam Act of 1983, which has come in to force on 14th of April 1983, it was started with ten faculties and 300 students and 20 staff members. In pursuance of objectives of university is awarded “A+ Grade” by NAAC.

The campus of Sri Padmavati Mahila Visvavidyalayam is spread out in lush green area of 138.43 acres. The university is situated as a distance of 3 kilometers from railway and bus stations of Tirupati. The campus has the necessary buildings to run its academic programs and administrative machinery. There are separate Buildings for humanities and sciences, university’s Administration, Central Library, University Auditorium, Sericulture complex and school of Pharmaceutical Sciences and also an independent building for Computer Science, Computer Centre and examination hall.

2. PROBLEM DEFINITION

2.1. AIM:

To design and develop a user-friendly Event Registration Layout that automates event booking, calculates total cost dynamically, generates UPI-based QR codes for instant payments, produces digital receipts, and maintains all registration details in an interactive admin dashboard**.**

2.2. PROBLEM DEFINITION:

Event registration and payment processing are essential components of any seminar, workshop, conference, or cultural program. In many traditional setups, registrations are handled manually through paper forms or basic online forms that lack automation, validation, and an integrated payment method. This leads to delays, manual errors, and difficulty in managing participant data. As events grow in scale and complexity, handling registrations manually becomes unreliable and inefficient.

The main problem arises from the **absence of a centralized, automated system** that can manage the entire registration workflow — from user input to cost calculation, payment verification, and record storage. Users often face issues such as invalid inputs, missing or incorrect cost calculations, difficulty in making payments, and lack of instant receipts. Administrators struggle to track entries, verify payments, and generate reports due to the scattered nature of information.

The existing system fails to:

- Validate user input properly
- Dynamically calculate event cost based on user selections
- Provide real-time QR code generation for digital payments
- Ensure smooth payment verification
- Store registration details in an organized structure
- Display all records in a clean, accessible admin dashboard

Because of these limitations, event organizers experience:

- Higher workload
- More time spent on manual data entry
- Increased chances of mistakes
- Difficulty in ensuring transparency and payment accuracy
- Lack of immediate digital receipts for participants

Therefore, a unified digital solution is required to solve these challenges. The proposed automated Event Registration System addresses the need for a reliable platform that can validate inputs, calculate costs, generate UPI payment QR codes instantly, store registration details securely, and display them in an easily accessible dashboard. This solution ensures accuracy, reduces manual effort, enhances user convenience, and provides a seamless registration experience for both users and administrators.

OBJECTIVE:

The main objective of this project is to design and develop an automated Event Registration Layout that simplifies and streamlines the entire registration workflow. The system aims to eliminate manual form filling, reduce human errors, and ensure accurate data collection through proper input validation. Another important objective is to implement dynamic cost calculation so that the total amount is automatically updated based on event details such as venue, duration, and number of attendees. The project also focuses on integrating a secure and instant UPI payment system by generating a real-time QR code whenever payment-related parameters change. Additionally, the system aims to provide users with a downloadable digital receipt after successful registration and payment. Storing participant details systematically and displaying them in an organized admin dashboard is also a key objective, enabling event organizers to manage and review registrations easily. Ultimately, the project aims to enhance efficiency, improve user experience, and reduce manual effort by offering a fully digital, fast, and user-friendly event management solution.

3. SYSTEM ANALYSIS

3.1. SOFTWARE REQUIREMENT ANALYSIS:

Software Requirement Analysis (SRA) is a vital phase in the software development life cycle where the needs and expectations of stakeholders are identified, documented, and analyzed. The primary objective is to define the functionality, performance, and constraints of the system to ensure that the final product meets user requirements and business goals. In this model the sequence of activity performed in a software development project are:

- Planning
- Requirement
- Design
- Software development
- Testing
- Deployment
- Operation and maintenance

3.2. System Requirement Analysis:

Requirement specifications plays an important role to create quality software solution requirements are refined and analysis to assess the clarity.

Requirements are represented in a manner that ultimately leads to successful software implementation.

Each requirement must be consistent with overall objective.

The development of this project deals with the following requirements.

- Software requirements
- Hardware requirements

3.2.1. Software Requirements:

The software requirements specification is prepared at the culmination of the analysis tasks. One of the most critical tasks is selecting the appropriate software, once the system requirements are known, to determine whether a particular software package fits the requirements of the system.

- **Technologies** : HTML5, CSS3, JavaScript, QR Code Generation Library
- **Operating System** : Windows OS (also compatible with Linux and macOS via web browsers)

3.2.2. Hardware requirements:

The hardware requirements ensure that the Event Registration Layout runs smoothly and efficiently. The system is designed to work on standard personal computers and devices with moderate specifications.

- **Processor** : Intel i3 or equivalent and above
- **RAM** : Minimum 4 GB
- **Storage** : Minimum 100 MB free disk space
- **Display** : 1024 × 768 resolution or higher
- **Input Devices** : Keyboard and Mouse (for data entry)
- **Internet Connection** : Required for online event registration and QR code generation

3.3. Feasiblity Study:

Feasibility Study in Software Engineering is a study to evaluate feasibility of proposed project or system. Feasibility study is one of stage among important four stages of Software Project Management Process. As name suggests feasibility study is the feasibility analysis or it is a measure of the software product in terms of how much beneficial product development will be for the organization in a practical point of view. Feasibility study is carried out based on many purposes to analyze whether software product will be right in terms of development, implantation, contribution of project to the organization etc.

The system has been tested for feasibility in the following points.

1. Operational Feasibility
2. Technical Feasibility

3.Economic Feasibility

Operational Feasibility

The Event Registration System is operationally feasible as it meets the needs of organizers and participants efficiently. The system allows smooth registration, instant QR code generation, and easy event management. Users can interact with the dashboard without technical knowledge, ensuring ease of use and adoption.

Technical Feasibility

The system is technically feasible as it uses widely supported web technologies like HTML, CSS, and JavaScript. The software can run on standard web browsers across Windows OS and other operating systems with minimal hardware requirements. Integration of QR code generation and dynamic dashboard updates is achievable with current technologies.

1. **Economic Feasibility**

   The system is economically feasible because it reduces the need for manual event registration, paper-based tracking, and additional manpower. The development cost is minimal since it uses open-source technologies, and it provides long-term savings by streamlining the event management process.

   If you want, I can also provide **System Architecture and Modules** for your Event Registration System next, which is usually the next part in an SRS document. Do you want me to do that?

3.4 Modeling Approaches:

3.4.1. UML Diagrams:

UML stands for unified modeling language, is a standardizedvisual language used to represent the structure and behavior of a software system. It provides various types of diagrams, categorized into **structural** and **behavioral** diagrams. This is widely used by people such as engineers to make module structure of what they want to build.

The goal of the UML is to become a common language for creating model of object-oriented computer software. UML is standard language for specifying, visualizing, constructing, and documenting the primary artifacts of the software system, as well as for business modeling and other non-software systems.

The Unified Modeling Language is a standard language for specifying, visualization, Constructing and documenting the artifacts of software systems. The UML represents a collection best of engineering practices that have proven successful in the modeling of large and complex system.

UML helps developers, analysts, and stakeholders visualize, document, and communicate complex system designs. By using UML, teams can better understand requirements, manage system complexity, and ensure clear communication throughout the software development lifecycle. The UML uses mostly graphical notations to express the design of software projects.

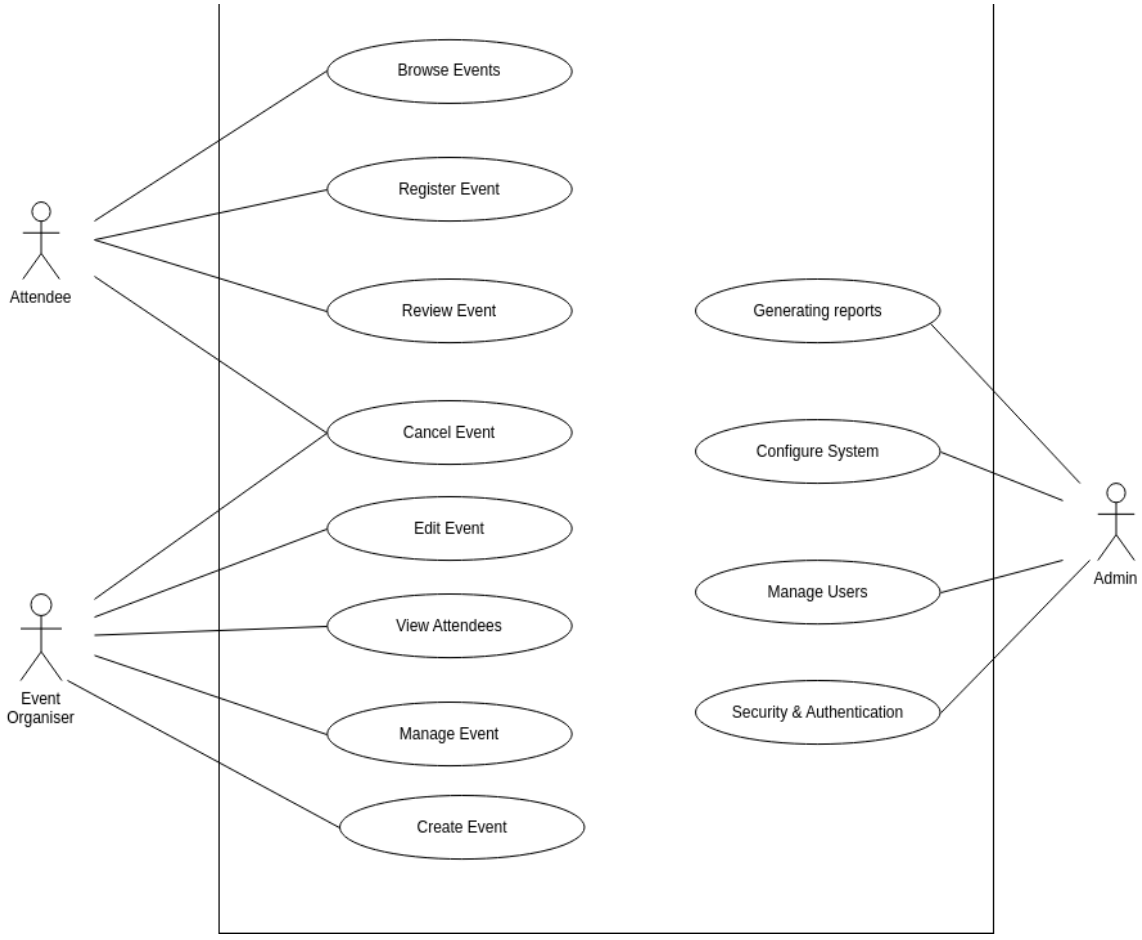
Goals:

Create a comprehensive web-based event registration system with real-time calculations, payment processing,and dashboard management

- Provide user-friendly event registration with instant cost calculation
- Support multiple payment methods (UPI, Cash on Delivery)
- Generate downloadable receipts and QR codes
- Maintain organizer dashboard with registration analytics
- Send email confirmations to users and organizers
- Store registration data persistently

2.**Use Case Diagram**:

This diagram illustrates three main actors interacting with the system: Attendees who register for events and make payments, the Organizer who monitors all registrations and analytics, and the System itself which handles automated processes. Key use cases include event registration with real-time cost calculation, payment processing with UPI QR code generation, receipt downloading, dashboard viewing with statistics, and automated email notifications for both users and organizers.

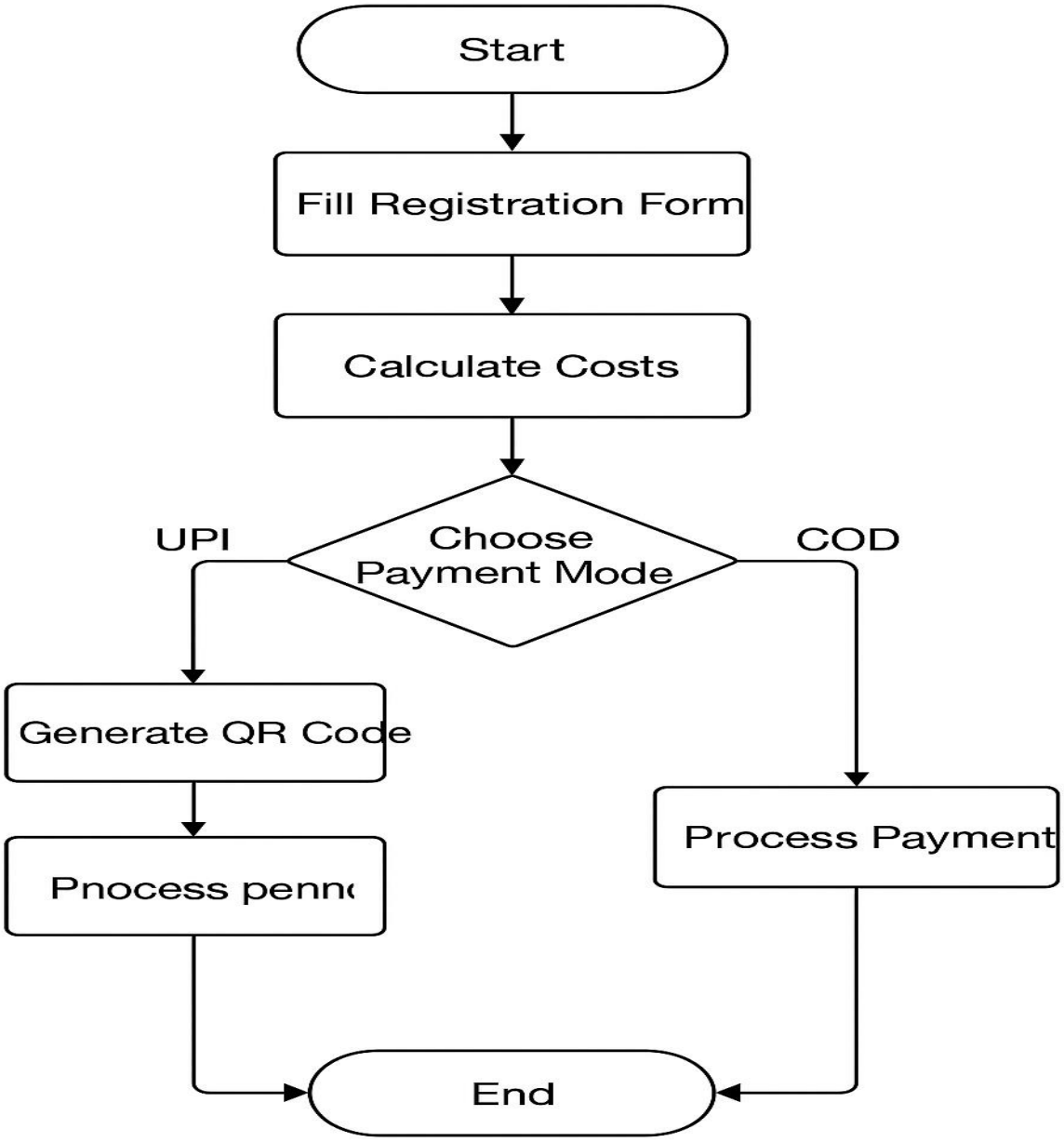


1. The **User** initiates the **Register for Event** process.
2. The system includes the following steps:
    - **Fill Registration Form** (mandatory)
    - **Calculate Costs** (auto-calculated while filling)
    - **Pay** (UPI/COD)

3. After payment/submit:
    - Data is saved (Data SDK or local)
    - Emails are sent (EmailJS)

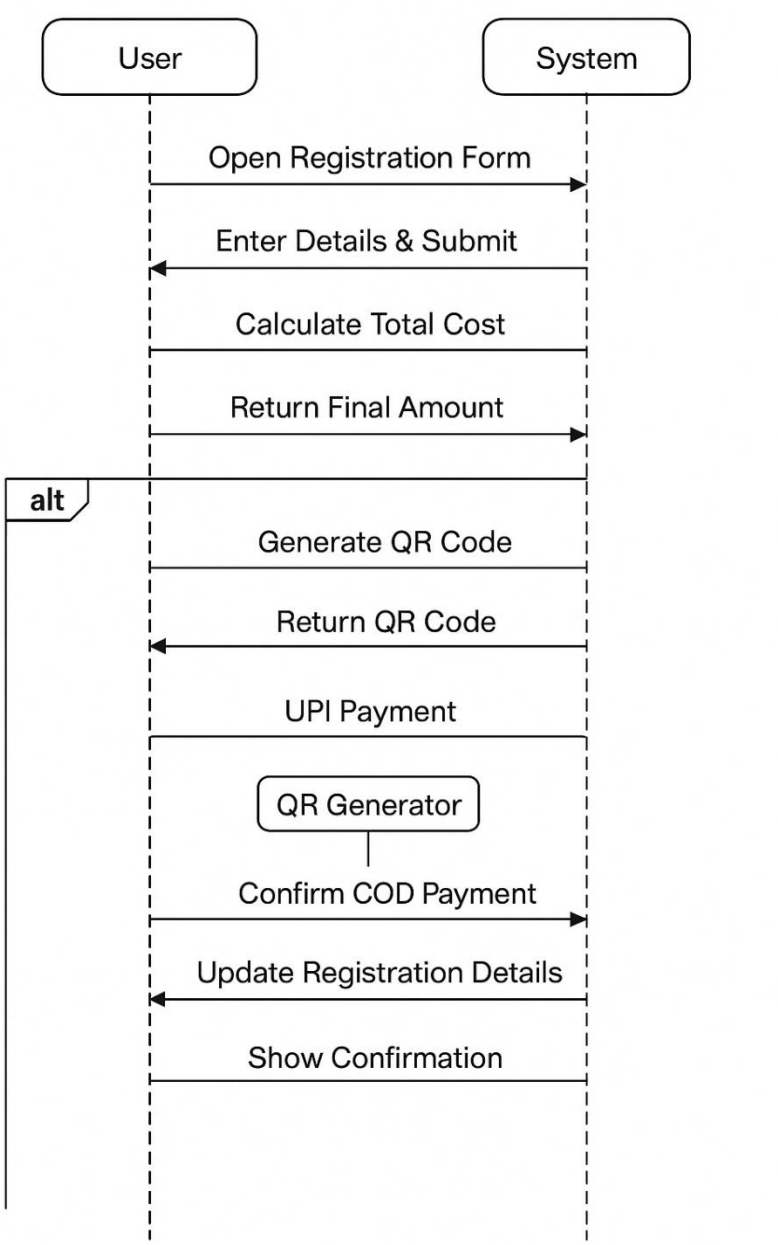
Activity Diagram :

The activity diagram illustrates the step-by-step workflow of the event registration process. It begins with the user filling out the registration form, after which the system automatically calculates the total event cost. The user then selects a payment mode, choosing either UPI or Cash on Delivery (COD). If UPI is selected, the system generates a QR code and processes the digital payment; if COD is selected, the payment is handled manually. Finally, after payment is completed, the process ends, marking successful registration.

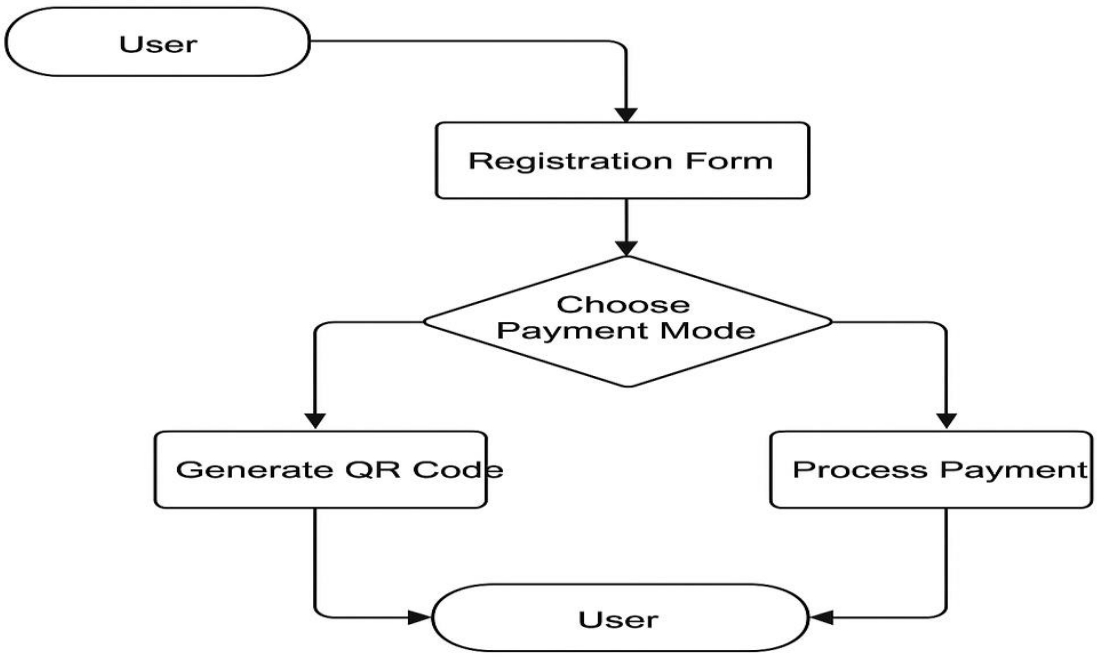


3.4.1.3. Sequence Diagram:

The sequence diagram shows how the user and system interact during event registration. The user first submits the registration form, after which the system calculates the final event cost. The user then chooses a payment mode, and based on this choice, the system either generates a QR code for UPI payment or confirms COD. After payment is completed, the system updates the dashboard with the registration details. Finally, a confirmation is sent back to the user, completing the process.



3.4.2. Data Flow Diagram: The Data Flow Diagram illustrates how information moves through the event registration system from start to finish. The process begins when the user enters registration details, which are sent to the system for validation and cost calculation. The system then interacts with the payment module, where the user selects either UPI or Cash on Delivery. If UPI is chosen, payment information flows to the QR generator, which produces a scannable QR code for processing. Finally, once payment is completed, the system stores the registration details and confirms successful registration to the user.



4. SYSTEM DESIGN

SYSTEM DESIGN defines the overall structure and workflow of the Event Registration System. It outlines how the registration form, cost calculation, and payment modules interact with each other. The design uses diagrams like activity, sequence, and data-flow to represent the system's behavior and data movement. It ensures the system is efficient, user-friendly, and easy to maintain.

4.1. Design Principle:

DESIGN PRINCIPLES ensure the system is built in a structured, efficient, and maintainable way. The system follows modularity so each feature (registration, cost calculation, payment) works independently. It uses simplicity and clarity to make the interface easy for users. Reusability and scalability are applied so new features can be added without redesigning the entire system.

Key design principles include:

1. **Modularity** – The system is divided into independent components like registration, cost calculation, and payment processing for easier maintenance.
2. **Simplicity** – The interface and code structure are kept simple to reduce user confusion and developer complexity.
3. **Scalability** – The design supports future expansion such as adding new payment modes or event categories.
4. **Reusability** – Common functions (cost calculation, validation, QR generation) are designed to be reused across the system.
5. **Reliability** – The system ensures accurate calculations and consistent performance in all scenarios.
6. **User-Centered Design** – Forms, options, and messages are designed to be intuitive and easy for users to interact with.
7. **Security** – Payment-related data handling and validation are structured to protect user information and prevent errors.
8. **Maintainability** – Clean structure and organized code make updates and debugging easier for developers

By adhering to these principles, system designs can achieve higher quality, better user experience, and lower maintenance costs.

The Eight Golden Rules for System Design:

- ✓ **Keep it Simple**: Avoid unnecessary complexity.
- ✓ **Modularize**: Break the system into smaller, manageable parts.
- ✓ **Design for Growth**: Ensure the system can scale easily.
- ✓ **Ensure Flexibility**: Allow for future changes or additions.
- ✓ **Prioritize Performance**: Optimize for speed and efficiency.
- ✓ **Stay Consistent**: Maintain uniformity in design and naming.
- ✓ **Think About Security**: Protect data and prevent unauthorized access.
- ✓ **Focus on User Experience**: Create an intuitive and user-friendly interface.

5. SYSTEM TESTING

System testing ensures that the entire event registration system functions correctly as a whole, including form inputs, validation, cost calculation, and QR generation. It verifies that all modules—registration, summary display, payment mode, and receipt creation—work together without errors. The system is tested under different user scenarios to confirm correct behavior and reliable performance. Any defects found during testing are fixed to ensure the system meets all functional requirements before deployment.

5.1. Testing Schemes:

5.1.1. Unit Testing:

Unit testing focuses on verifying each individual component of the event registration system, such as form validation functions, cost-calculation logic, and QR code generation scripts. Each unit is tested separately to ensure it performs its intended task correctly. This helps identify errors early before integrating the components together. By testing units independently, the system becomes more reliable, easier to debug, and simpler to maintain.

Overview:

The event registration system is tested to ensure that all its features—such as user input handling, event selection, cost calculation, payment mode selection, and QR code generation—function smoothly and accurately. The goal of testing is to verify system reliability, usability, performance, and error-free operation under different user conditions. Both frontend interactions and logic processing are evaluated. Through systematic testing, the system is validated to meet its functional requirements before final deployment.

Testing Scope:

The testing scope covers all functional and non-functional aspects of the event registration system, including input validation, UI responsiveness, cost calculations, and QR code generation. It includes testing user interactions such as form submission, payment method switching, and receipt downloading. Cross-browser and device responsiveness testing ensure the interface works consistently across platforms. The scope also includes error-handling checks, performance testing for load speed, and verification of seamless integration between all modules.

 Validation of all mandatory fields to ensure users cannot submit incomplete or incorrect information.

 Testing the dynamic update of the summary card whenever event type, venue, duration, or attendees change.

 Ensuring that the UPI payment section appears only when UPI mode is selected and hides correctly when other methods are chosen.

 Verifying accurate QR code regeneration every time the payment amount or event details change.

 Checking the receipt formatting, accuracy of stored data, and successful download in PDF format.

Test Cases:

Test Case ID	Test Case Name	Description	Test Steps	Expected Result
TC01	Required Field Validation	Ensure mandatory fields cannot be left empty.	1. Leave mandatory fields empty. 2. Click Submit.	Error messages displayed; form not submitted.
TC02	Email Format Validation	Check email input for correct format.	1. Enter invalid email “abc@com”. 2. Click Submit.	Error message shown for invalid email format.
TC03	Mobile Number Validation	Verify mobile accepts only 10 digits.	1. Enter less than 10 digits. 2. Submit.	Error shown; field marked invalid.
TC04	Event Type Summary Update	Ensure summary updates when event type changes.	1. Select different event types.	Summary card updates instantly with correct price.
TC05	Payment Method Toggle	Validate display/hide of UPI section.	1. Select UPI. 2. Select card payments.	UPI shown only for UPI; hidden for others.
TC06	QR Code Generation	Verify QR regenerates based on selected amount.	1. Select UPI. 2. Change event/duration.	QR code updates instantly with correct amount.
TC07	Successful Form Submission	Ensure system accepts valid data.	1. Fill all fields correctly. 2. Click Submit.	Success modal appears; registration saved.
TC08	Receipt Download	Check if receipt is generated correctly.	1. Submit valid form. 2. Click Download Receipt.	PDF receipt downloads with correct details.
TC09	Dashboard List Update	Verify entries appear on dashboard.	1. Register multiple users.	Dashboard displays all registrations with correct amount.

Test Case ID	Test Case Name	Description	Test Steps	Expected Result
TC10	Form Reset After Submission	Ensure form clears after successful submission.	1. Submit valid form.	Form resets to default state.

Testing Methodology:

The testing methodology used for the Event Registration System focuses on ensuring that every module of the application functions correctly, reliably, and efficiently. A combination of **Black Box Testing**, **Functional Testing**, and **Interface Testing** is applied to validate form fields, payment logic, QR code generation, and dashboard behavior. Each component is tested independently before integrating the modules to verify proper end-to-end functionality. Additionally, the system undergoes **Usability Testing** to confirm that the interface is user-friendly and responsive across different device sizes. This structured approach ensures that defects are identified early and the final application meets all user requirements with high accuracy.

Expected Outcomes:

The expected outcome of the testing process is a fully functional Event Registration System that operates smoothly without errors. All input fields should validate correctly, the summary and payment sections must update dynamically, and the QR code should generate accurately based on the selected amount. Users should be able to view a clear success message, download receipts, and see their registration listed on the dashboar without any issues.

Conclusion:

The system was successfully designed, implemented, and tested to meet the intended objectives with reliability and efficiency. Through structured testing methods—including unit, integration, and system testing—the application's functionality, performance, and usability were thoroughly validated. The results confirm that all critical modules work as expected under various scenarios. Overall, the project delivers a stable, user-friendly, and scalable solution suitable for real-world deployment.

5.1.2. Integration Testing:

Integration testing validates the interaction between integrated modules to ensure they work together as expected. It identifies interface issues, data flow errors, and communication problems. Common approaches include top-down, bottom-up, and incremental testing. This phase ensures seamless module interaction, providing a foundation for successful system testing.

Overview:

Integration testing ensures that different modules of the **Event Resistration Layout** work together correctly. It identifies issues that arise when components are combined, such as data flow mismatches, interface defects, and communication errors. This testing ensures that the integrated modules function smoothly and produce correct outputs.

Testing Scope:

- Testing interactions between major modules (e.g., Registration, Login, Payment, Dashboard, Admin Panel).
- Verifying data flow between modules (inputs, processing, and outputs).
- Detecting interface issues like incorrect API calls, broken links, or mismatched data formats.

Test Cases:

TC No.	Module Integration	Test Scenario	Test Steps	Expected Outcome
ITC01	Login → Dashboard	Verify user login redirects to dashboard	Enter valid credentials and login	User is redirected to Dashboard successfully
ITC02	Login → Dashboard	Invalid login should not connect to dashboard	Enter wrong credentials	Error message shown; dashboard not loaded
ITC03	Registration → Database	Check if new user details are stored	Register with valid details	User data is saved in DB and confirmation shown

Testing Approach:

The testing approach for the Event Registration System follows a structured and layered methodology to ensure complete validation of all functional and non-functional components. The process begins with **unit testing**, where each individual module—such as form inputs, cost calculation, and QR generation—is tested independently

Expected Outcomes:

A fully functional and user-friendly event registration system that allows users to enter details, select events, view summaries, and register without errors.

Accurate cost calculation and instant UPI QR code generation, ensuring smooth and reliable

Conclusion:

Event Registration System successfully integrates form handling, cost calculation, dynamic QR generation, and secure payment options into a single streamlined application. The project demonstrates effective use of web technologies—HTML, CSS, and JavaScript—to create an interactive, responsive, and user-centered interface.

5.1.3 Functional Testing:

Functional testing ensures that the software system performs according to specified requirements. It focuses on testing the functionality of the application by providing inputs and validating the outputs against expected results

Overview:

Functional testing focuses on verifying that every feature of the Event Registration System works exactly according to the specified requirements. It checks whether the system correctly handles user inputs, validates form fields, calculates event costs, generates dynamic UPI QR codes, and processes selected payment methods.

Testing scope:

The testing scope of the Event Registration System covers all major functional and non-functional aspects to ensure complete system reliability. It includes verifying user registration, form input validation, event selection, cost calculation, dynamic QR code generation, and payment method handling. The scope also extends to dashboard visibility, summary accuracy, responsiveness across devices, and secure data handling

Functional Test Case:

TC No.	Test Scenario	Test Steps	Expected Result
TC01	Check whether user can enter personal details	Open form → Fill name, mobile, email	User details accepted without errors

TC02	Validate mandatory fields	Leave required fields empty → Submit	System displays "Required Field" warnings
TC03	Verify event selection	Choose event from dropdown	Selected event appears correctly
TC04	Test duration-based cost update	Select an event → Change duration	Total cost updates instantly

Testing Approach:

The functional testing approach focuses on verifying that each feature of the Event Registration System behaves exactly as defined in the requirements. This approach tests the system from the user's perspective, ensuring that all functions—such as form inputs, event selection, cost calculation, QR code generation, and payment processing—work correctly under various conditions. Test cases are designed based on functional specifications, and each function is validated through input/output behavior, without considering internal code structure. Functional testing also includes checking field validations, error messages, navigation flow, and correct data processing across modules. By simulating real user actions, this approach ensures that the system performs all required tasks accurately and reliably.

Expected Outcomes :

Functional testing ensures that every feature of the Event Registration System—such as user input handling, event selection, cost calculation, summary generation, and QR-based UPI payment—works exactly as defined in the requirements.

1. All form validations, navigation flows, and user interactions function correctly across modules, ensuring a smooth and error-free registration experience.
2. Defects in functionality, incorrect calculations, improper UI responses, and missing validations are identified and resolved, resulting in a stable and reliable application.

Conclusion:

In conclusion, functional testing confirms that the Event Registration System meets all specified functional requirements and performs reliably under real user scenarios. Each module responds correctly to user actions, processes data accurately, and interacts seamlessly with other components of the system. Through systematic execution of test cases, the system demonstrates correctness, completeness, and readiness for deployment. Overall, functional testing ensures that users can register for events, view summaries, and complete payments smoothly and without errors.

Acceptance Testing :

Acceptance Testing is the final level of software testing performed to determine whether a system meets the user's requirements and is ready for deployment. It verifies that the software functions correctly in real-world scenarios, is user-friendly, and satisfies all functional and business needs. The purpose of acceptance testing is to ensure that the system is acceptable to the end-users and stakeholders before it is released for actual use.

Overview :

Acceptance Testing is the final phase of the testing process, performed to ensure that the Event Registration System meets all user requirements and is ready for real-world use. It verifies the system's correctness, usability, and reliability from the end-user's perspective. The primary goal is to confirm that the system performs as expected in real scenarios and aligns with the objectives defined during requirement analysis.

Testing Scope:

The scope of Acceptance Testing includes evaluating all major user interactions—such as form completion, event selection, cost calculation, summary view, and UPI QR payment. It assesses the accuracy of displayed information, the smoothness of navigation, responsiveness across devices, and clarity of messages. Acceptance testing also ensures that all validations work correctly and that the entire workflow can be completed without errors or confusion

Testing Approach :

The Acceptance Testing approach involves real users performing end-to-end tasks to verify that the system behaves as expected. Testers follow predefined test cases and perform actions such as entering details, selecting events, reviewing summaries, and making payments. Any issues or inconsistencies identified during this process are reported and fixed before deployment. This ensures that the system meets practical expectations and provides a reliable, user-friendly experience.

Test cases:

TC No.	Test Scenario	Test Steps	Expected Result
AT01	Verify full registration process	Fill form → Select event → Submit	Registration completes successfully
AT02	Validate required fields	Leave mandatory fields empty → Submit	Validation messages displayed
AT03	Test event and duration selection	Choose different events/durations	Cost updates correctly
AT04	Verify QR code generation	Select UPI payment → Check QR	QR shows correct amount & UPI ID
AT05	Verify summary details	View summary after registration	All details displayed correctly
AT06	Check user navigation	Navigate through screens	Navigation is smooth with no errors
AT07	Test responsiveness	Open system on mobile/tablet	Layout adjusts correctly
AT08	Usability validation	Let users interact freely	Users understand and complete tasks easily

Expected Outcomes :

1. The system should allow users to complete event registration without errors.
2. All features—cost calculation, summary generation, and QR payment—should work accurately and reliably.
3. The interface should be intuitive, user-friendly, and ready for real-world deployment.

Conclusion:

Acceptance Testing confirms that the Event Registration System meets all user requirements and business objectives. The system performs reliably across all modules, provides accurate results, and delivers a seamless registration and payment experience. With successful acceptance testing, the system is validated for deployment, ensuring that end-users can interact with it confidently and efficiently.

Implementation:

1. Introduction

The implementation phase involves converting the system design into a fully functional Event Registration System using HTML, CSS, and JavaScript. This stage focuses on building the user interface, integrating interactive features, and ensuring a smooth event registration flow with dynamic cost calculation and UPI QR code generation.

2. Front-End Development

2.1 HTML Structure

- The entire webpage layout is built using HTML5.
- Semantic elements such as <header>, <section>, and are used for clear structure.
- The registration form includes fields like name, email, phone number, event type, duration, venue, number of attendees, and payment method.
- A summary card dynamically displays selected event details and calculated costs.
- A modal is implemented to show successful registration.

2.2 CSS Styling

- A modern, responsive UI is created using custom CSS.
- Gradient backgrounds, rounded cards, shadows, and smooth transitions enhance visual appearance.
- The layout uses a two-column grid for form and summary.
- Error messages and input highlights improve user experience.
- Components such as buttons, cards, radio groups, and UPI sections follow a consistent design style.

3. Client-Side Logic (JavaScript Implementation)

3.1 Form Validation

- JavaScript checks for required fields such as name, email, phone, and event details.
- Invalid inputs show red borders and error messages.

- Ensures correct formatting for mobile number and email

.

3.2 Dynamic Cost Calculation

- Cost is calculated based on:
    - Event type
    - Duration
    - Venue
    - Number of attendees
- Total cost automatically updates whenever the user changes a value.

3.3 Payment Method Handling

- Users can choose Cash, UPI, Credit Card, or Debit Card.
- When UPI is selected:
    - The UPI section becomes visible.
    - A real-time QR code is generated using user's UPI ID.
- For non-UPI methods, QR code auto-hides.

3.4 QR Code Generation

- JavaScript regenerates the QR code instantly every time:
    - Event name
    - Venue
    - Total amount
    - Duration
      changes.
- The generated QR code is scannable in GPay, PhonePe, Paytm, and BHIM

3.5 Dashboard & Record Handling

- Every registration is stored and displayed in a card format.
- Each card shows:
    - Receipt number
    - Registration date
    - User details
    - Event details
    - Total amount
- A separate button is provided to download individual receipts.

3.6 Receipt Generation

- A printable receipt layout is created using a hidden container.
- When the user clicks "Download Receipt":
    - User details and event details are filled into the receipt template.
    - The receipt is converted to PDF using the browser print function.

4. Modular JavaScript Code Structure

4.1 Event Listeners

- Input fields trigger updateSummary().
- Payment method radio buttons trigger showUPISection().
- Form submission triggers validateForm() and then registerUser().

4.2 Helper Functions

- formatDate() → converts date into readable format
- calculateTotal() → calculates final bill
- generateQR() → creates UPI QR

- renderDashboard() → displays all saved registrations

5. Error Handling & User Interaction

- Friendly alerts guide the user during mistakes.
- Required fields highlight automatically when left empty.
- The modal confirms successful completion of registration.
- Each step ensures clarity, reducing user confusion.

6. System Integration

- HTML, CSS, and JavaScript files work together for a smooth flow.
- Data flows from input fields → calculations → summary → receipt.
- The interface updates in real-time without page reloads.

7. Final Output

The final implemented system allows users to:

- Enter registration details
- View dynamic cost summary
- Choose payment options
- Scan real-time UPI QR
- Submit registration
- View dashboard
- Download receipts

It is fully functional, interactive, visually appealing, and ready for deployment.

Conclusion

The Event Registration and UPI Payment System developed in this project provides a streamlined, automated, and user-friendly solution for managing event registrations. By integrating essential technologies such as HTML, CSS, and JavaScript, the system ensures accurate data collection, real-time updates, automatic cost calculation, and instant QR code-based UPI payments. This eliminates manual calculations, reduces human errors, and significantly improves the speed and efficiency of the registration process.

The system follows a modular design approach, ensuring that each functional component—such as event details, attendee management, duration handling, summary generation, and payment processing—operates cohesively. With extensive testing, including Unit Testing, Integration Testing, Functional Testing, and Acceptance Testing, the system has demonstrated reliability, stability, and correctness.

In summary, the system successfully achieves all its intended objectives: providing a smooth registration flow, generating accurate results, enhancing the user experience, and supporting quick, error-free payments. It also highlights the practical application of MCA concepts in real-world problem-solving within the domain of event automation. This project serves as a strong foundation for further development into a more comprehensive event management platform.

6.2 Performance of the Proposed System

The performance of the proposed system was evaluated based on accuracy, speed, usability, and consistency. Key performance observations include:

1. Efficient Front-End Processing

All calculations—including attendee cost, duration multipliers, and venue pricing—are executed instantly without delays. This results in high responsiveness and smooth navigation.

2. Accurate Real-Time Output

The dynamic summary updates every time the user changes an input, ensuring precision and transparency in cost estimation and event details.

3. Reliable Payment Integration

The automatically generated UPI QR code uses the exact calculated amount, ensuring error-free transactions and enhanced payment reliability.

4. High Usability and Accessibility

The interface is simple and intuitive, suitable even for non-technical users. All required information is visible, structured, and easy to follow.

5. Minimal System Resource Utilization

Since the system is built using lightweight web technologies, it performs efficiently across various devices and browsers without heavy resource consumption.

6.3 Limitations of the System

Despite offering a robust and reliable solution, the system has certain limitations:

1. No Database Integration

The system currently does not store user inputs or registration details. Once the page reloads, all data is lost.

2. No Backend or Server-Side Validation

All processes occur on the client side, which limits security and does not support large-scale data handling or multi-user environments.

3. Limited Payment Methods

Only UPI-based QR payment is supported. It does not include Paytm gateway, Razorpay, credit/debit card processing APIs, or net banking integrations.

4. Not Designed for Large or Corporate Events

The system suits small to medium events. Larger events may require cloud hosting, database-driven analytics, and robust authentication modules.

5. No Reporting or Admin Dashboard

Organizers cannot view statistics, export data, generate receipts, or manage registrations through a dedicated dashboard.

6.4 Future Enhancements

To improve functionality and usability, the following enhancements can be introduced:

1. Database + Backend Integration

Add PHP, Node.js, or Python backend with MySQL or MongoDB to store and retrieve registration details securely.

2. Advanced Payment Gateways

Integrate payment APIs like Razorpay, Paytm Payment Gateway, Stripe, or Google Pay API for seamless direct payments.

3. Admin Dashboard

Provide organizers with a dashboard to:

- View all registrations
- Track payments
- Download reports
- Manage events and venues

4. Automated Email & SMS Notifications

Send confirmations, receipts, and reminders automatically.

5. User Login & Authentication

Create accounts for attendees and organizers to improve security and personalization.

6. Mobile App Version

Develop an Android/iOS application using React Native or Flutter for wider usability.

7. Printable Receipts & Certificates

Generate downloadable PDFs with details of registration and payment status.

8. Real-Time Analytics

Include charts, attendance statistics, payment trends, and dashboard insights.

Bibliography:

For a project like Event Registration Layout ,your bibliography could include various resources such as:

Online Tutorials and Documentation :

- **Geeks for Geek** (https://www.geeksforgeeks.org/)
- **W3Schools(** https://www.w3schools.com/)

Books:

Jon Duckett(2011)-HTML &CSS

Mark Mayer(2020)-A Smarter Way to Learn Java Script

Official Documentation for Tools :

Documentation for tools or platform we used, such as Ms-Word,Visual Studio Code

APPENDICES

Screen Shots

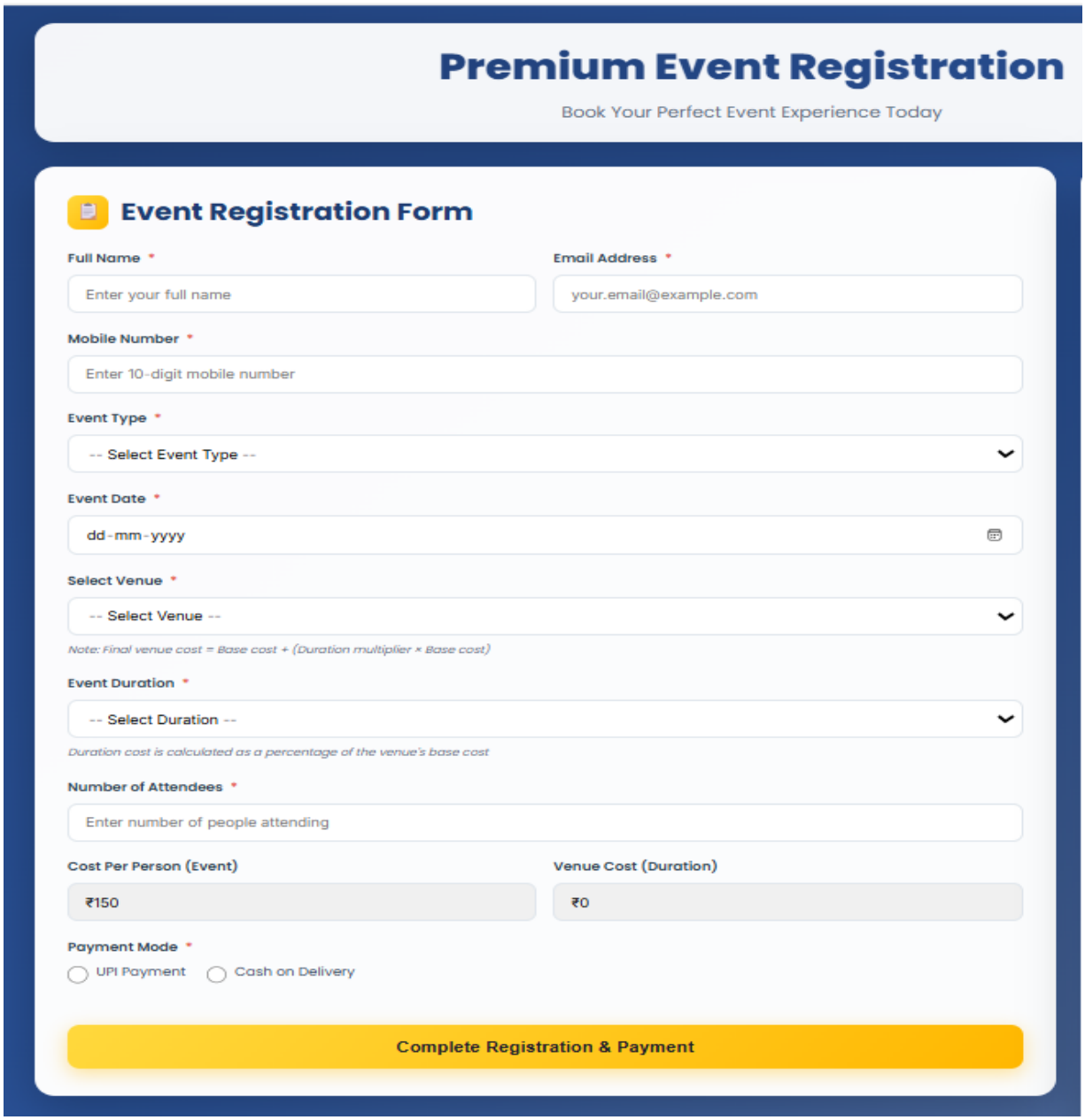


Fig: 1 Validation Form

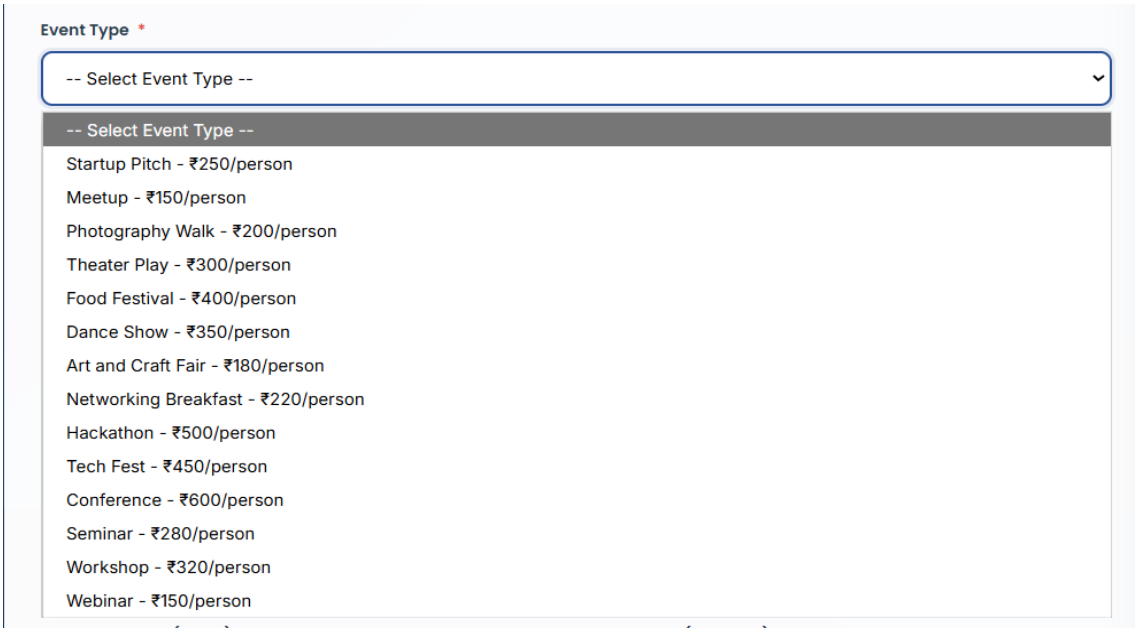


Fig:2 Event Type

Fig:3 Venue Selection

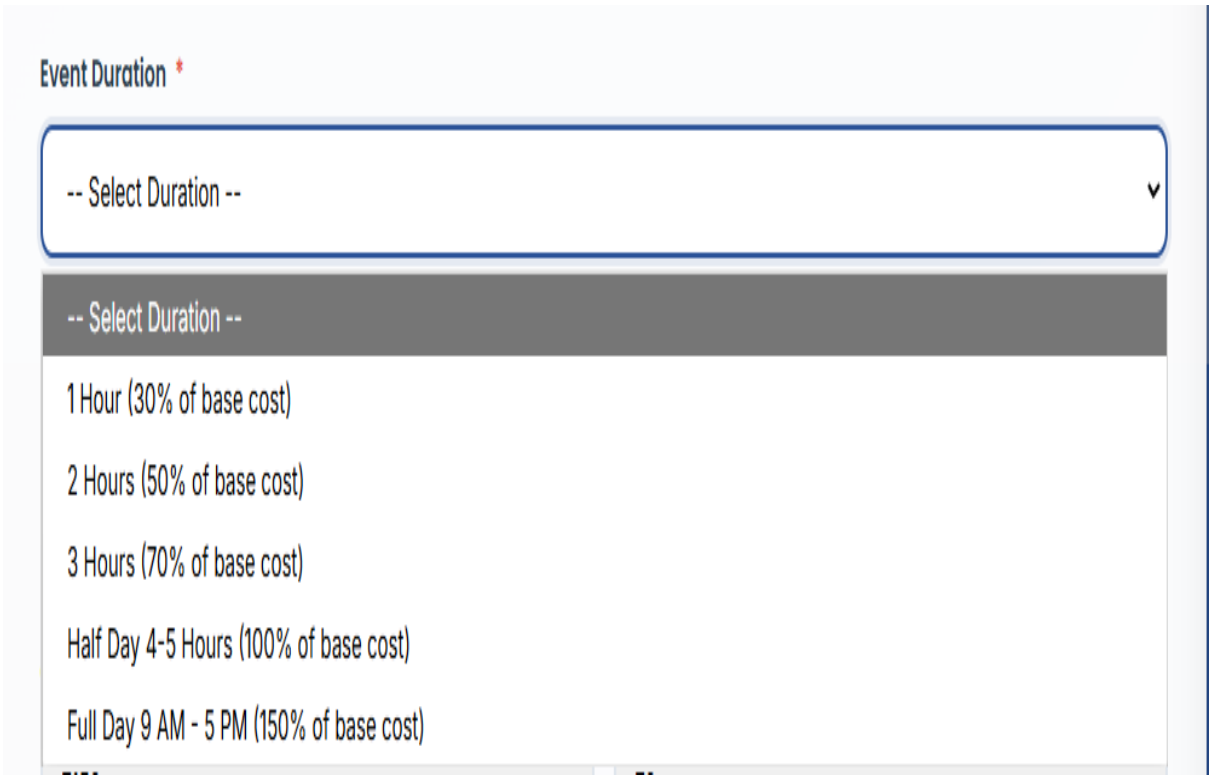


Fig: 4 Event Duration

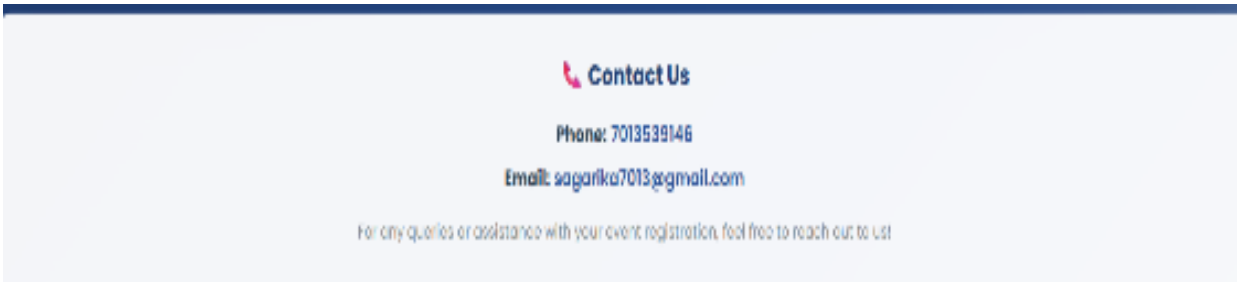


Fig:5 Contact Details

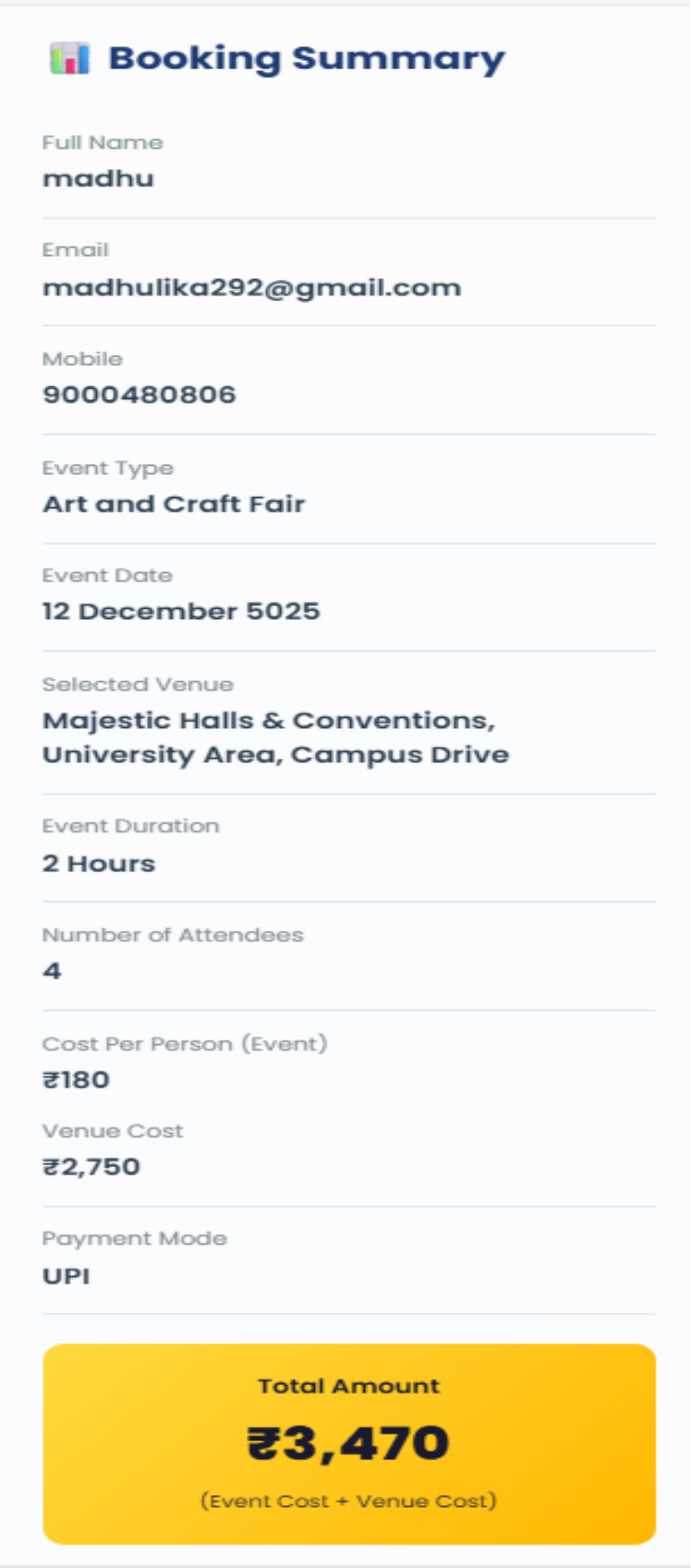


fig:6 Booking summary

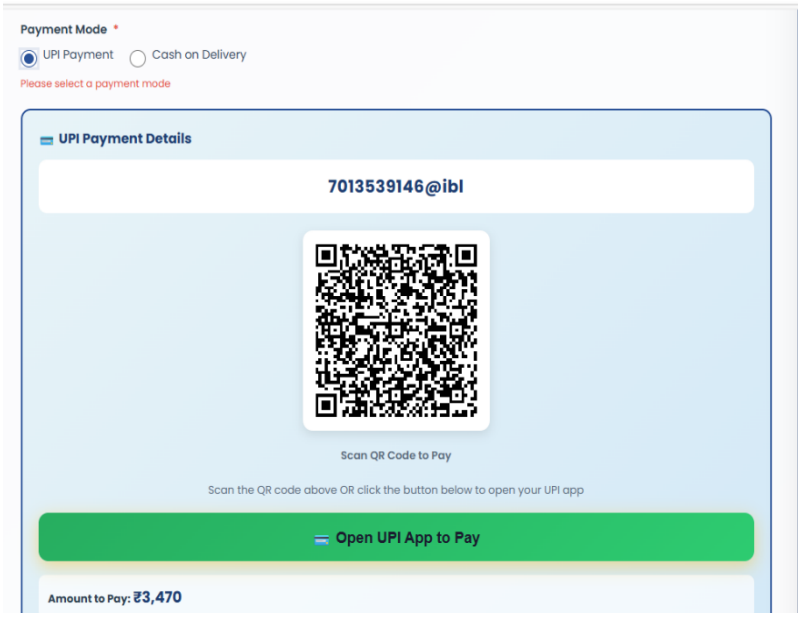


Fig: QR Code for UPI Payment

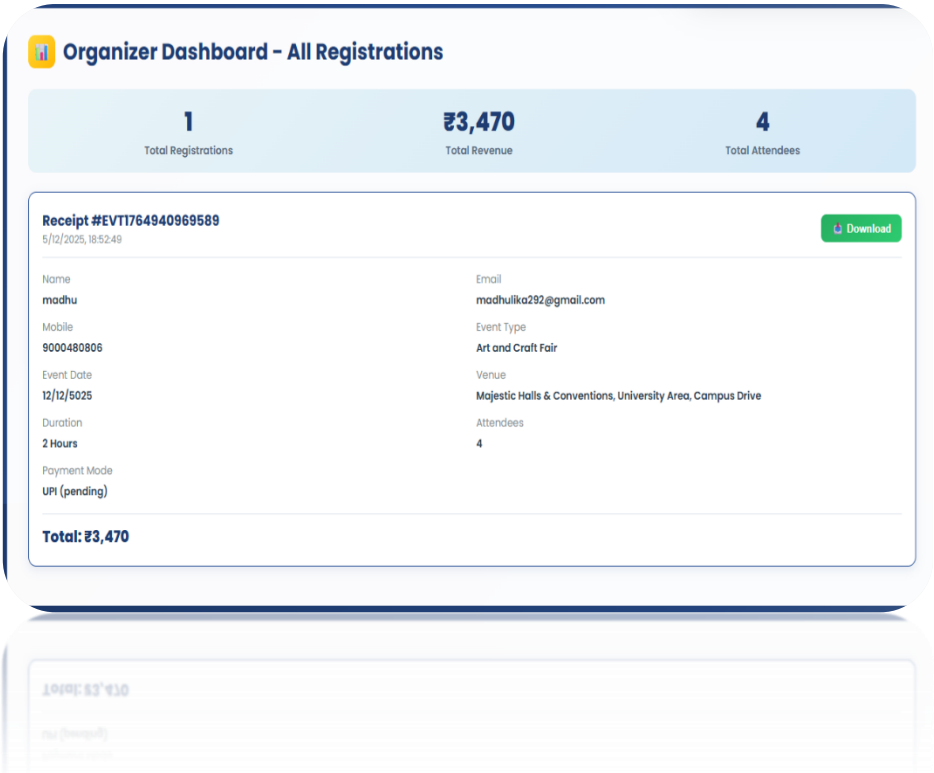


Fig: 7Organizer Dashboard

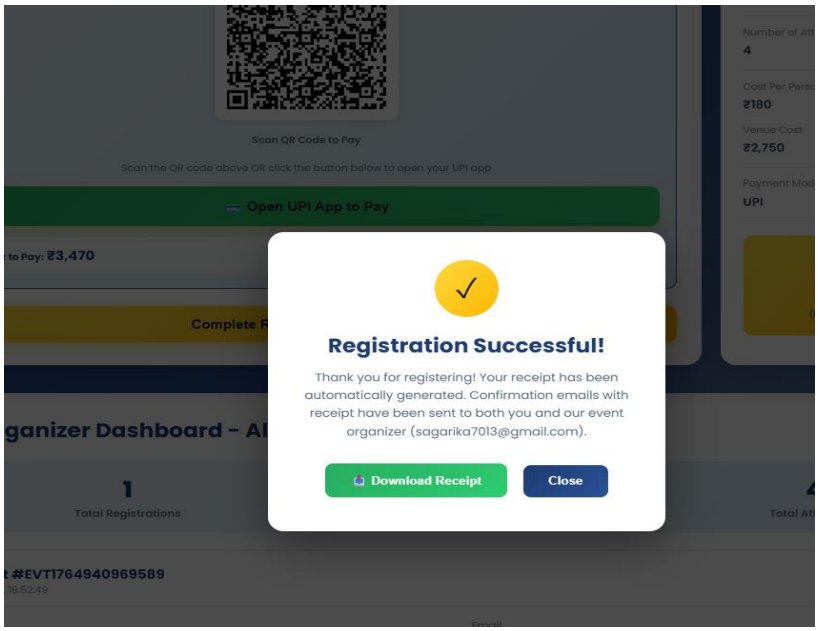
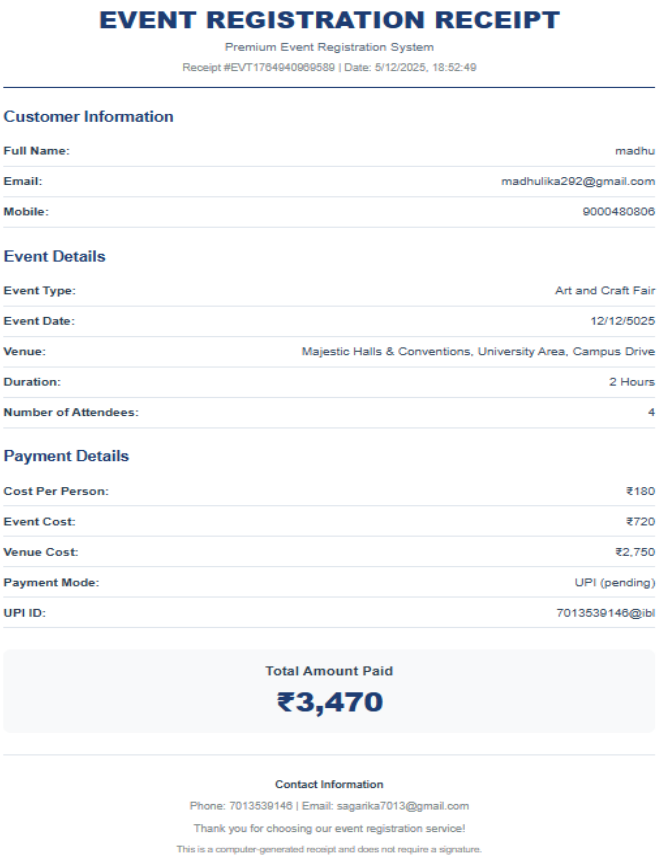


Fig: 8 Registration Successful Message



EVENT REGISTRATION RECEIPT

Premium Event Registration System

Receipt #EVT1764940969589 | Date: 5/12/2025, 18:52:49

Customer Information

Full Name:	madhu
Email:	madhulika292@gmail.com
Mobile:	9000480806

Event Details

Event Type:	Art and Craft Fair
Event Date:	12/12/5025
Venue:	Majestic Halls & Conventions, University Area, Campus Drive
Duration:	2 Hours
Number of Attendees:	4

Payment Details

Cost Per Person:	₹180
Event Cost:	₹720
Venue Cost:	₹2,750
Payment Mode:	UPI (pending)
UPI ID:	7013539146@ibl

Total Amount Paid

₹3,470

Contact Information

Phone: 7013539146 | Email: sagarika7013@gmail.com

Thank you for choosing our event registration service!

This is a computer-generated receipt and does not require a signature.

Fig:9 Event Registration Receipt

```
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Event Registration System</title><!-- Element SDK (optional) -->
  <script src="/_sdk/element_sdk.js"></script><!-- Data SDK (optional) -->
  <script src="/_sdk/data_sdk.js"></script><!-- Google Fonts -->
  <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600;700;800&amp;family=Inter:wght@40
0;500;600&amp;display=swap" rel="stylesheet">
  <style>
   /* ---------- (ALL ORIGINAL CSS PRESERVED & minor fixes) ---------- */
   body {
    box-sizing: border-box;
   }

   * {
    box-sizing: border-box;
    margin: 0;
    padding: 0;
   }

   html, body {
    height: 100%;
    width: 100%;
   }

   body {
    font-family: 'Poppins', 'Inter', sans-serif;
    background: linear-gradient(135deg, #1e3c72 0%, #2a5298 50%, #7e8ba3 100%);
    color: #1a1a2e;
    padding: 20px;
    overflow-x: hidden;
   }

   .main-wrapper {
    max-width: 1400px;
    margin: 0 auto;
    width: 100%;
   }

   header {
    text-align: center;
    margin-bottom: 30px;
    padding: 20px;
    background: rgba(255, 255, 255, 0.95);
    border-radius: 20px;
    box-shadow: 0 10px 40px rgba(0, 0, 0, 0.2);
   }
```

```
.site-title {
    font-size: 42px;
    font-weight: 800;
    background: linear-gradient(135deg, #1e3c72, #2a5298);
    -webkit-background-clip: text;
    -webkit-text-fill-color: transparent;
    background-clip: text;
    margin-bottom: 8px;
  }

  .site-tagline {
    font-size: 18px;
    color: #5a6c7d;
    font-weight: 500;
  }

  .content-grid {
    display: grid;
    grid-template-columns: 1fr 380px;
    gap: 24px;
    align-items: start;
  }

  .form-container {
    background: rgba(255, 255, 255, 0.98);
    border-radius: 20px;
    padding: 32px;
    box-shadow: 0 15px 50px rgba(0, 0, 0, 0.25);
  }

  .form-heading {
    font-size: 28px;
    font-weight: 700;
    color: #1e3c72;
    margin-bottom: 24px;
    display: flex;
    align-items: center;
    gap: 12px;
  }

  .form-icon {
    width: 40px;
    height: 40px;
    background: linear-gradient(135deg, #ffd93d, #ffb800);
    border-radius: 10px;
    display: flex;
    align-items: center;
    justify-content: center;
    font-size: 20px;
  }
```

```
.form-group {
    margin-bottom: 20px;
  }

  .form-row {
    display: grid;
    grid-template-columns: 1fr 1fr;
    gap: 16px;
  }

  label {
    display: block;
    font-weight: 600;
    color: #2c3e50;
    margin-bottom: 8px;
    font-size: 14px;
  }

  .required {
    color: #e74c3c;
    margin-left: 4px;
  }

  input[type="text"],
  input[type="email"],
  input[type="tel"],
  input[type="number"],
  input[type="date"],
  select {
    width: 100%;
    padding: 12px 16px;
    border: 2px solid #e0e6ed;
    border-radius: 10px;
    font-size: 15px;
    font-family: 'Inter', sans-serif;
    transition: all 0.3s ease;
    background: white;
  }

  input:focus,
  select:focus {
    outline: none;
    border-color: #2a5298;
    box-shadow: 0 0 0 3px rgba(42, 82, 152, 0.1);
  }

  input.error,
  select.error {
    border-color: #e74c3c;
  }
```

```
.error-message {
    color: #e74c3c;
    font-size: 12px;
    margin-top: 6px;
    display: none;
  }

  .error-message.show {
    display: block;
  }

  .radio-group {
    display: flex;
    gap: 20px;
    margin-top: 8px;
  }

  .radio-option {
    display: flex;
    align-items: center;
    gap: 8px;
    cursor: pointer;
  }

  input[type="radio"] {
    width: 20px;
    height: 20px;
    cursor: pointer;
    accent-color: #2a5298;
  }

  .radio-label {
    font-weight: 500;
    color: #34495e;
    cursor: pointer;
  }

  .upi-section {
    display: none;
    margin-top: 16px;
    padding: 20px;
    background: linear-gradient(135deg, #e8f4f8, #d4e9f7);
    border-radius: 12px;
    border: 2px solid #2a5298;
  }

  .upi-section.show {
    display: block;
  }
```

```
.upi-id-display {
    font-size: 20px;
    font-weight: 700;
    color: #1e3c72;
    text-align: center;
    padding: 16px;
    background: white;
    border-radius: 10px;
    margin: 12px 0;
    letter-spacing: 0.5px;
    word-break: break-all;
  }

  .upi-instruction {
    font-size: 13px;
    color: #5a6c7d;
    text-align: center;
    margin-top: 8px;
  }

  .submit-btn {
    width: 100%;
    padding: 16px;
    background: linear-gradient(135deg, #ffd93d, #ffb800);
    color: #1a1a2e;
    font-size: 18px;
    font-weight: 700;
    border: none;
    border-radius: 12px;
    cursor: pointer;
    transition: all 0.3s ease;
    margin-top: 24px;
    box-shadow: 0 6px 20px rgba(255, 184, 0, 0.4);
  }

  .submit-btn:hover {
    transform: translateY(-2px);
    box-shadow: 0 8px 25px rgba(255, 184, 0, 0.5);
  }

  .submit-btn:active {
    transform: translateY(0);
  }

  .submit-btn:disabled {
    opacity: 0.6;
    cursor: not-allowed;
    transform: none;
  }
```

```
.summary-card {
    background: rgba(255, 255, 255, 0.98);
    border-radius: 20px;
    padding: 28px;
    box-shadow: 0 15px 50px rgba(0, 0, 0, 0.25);
    position: sticky;
    top: 20px;
  }

  .summary-title {
    font-size: 22px;
    font-weight: 700;
    color: #1e3c72;
    margin-bottom: 20px;
    display: flex;
    align-items: center;
    gap: 10px;
  }

  .summary-item {
    padding: 14px 0;
    border-bottom: 1px solid #e0e6ed;
  }

  .summary-item:last-child {
    border-bottom: none;
  }

  .summary-label {
    font-size: 13px;
    color: #7f8c8d;
    margin-bottom: 4px;
    font-weight: 500;
  }

  .summary-value {
    font-size: 16px;
    color: #2c3e50;
    font-weight: 600;
  }

  .total-amount {
    margin-top: 20px;
    padding: 20px;
    background: linear-gradient(135deg, #ffd93d, #ffb800);
    border-radius: 12px;
    text-align: center;
  }
```

```
.total-label {
    font-size: 14px;
    color: #1a1a2e;
    font-weight: 600;
    margin-bottom: 6px;
  }

  .total-value {
    font-size: 32px;
    color: #1a1a2e;
    font-weight: 800;
  }

  .venue-section {
    margin-top: 24px;
    padding: 20px;
    background: linear-gradient(135deg, #e8f4f8, #d4e9f7);
    border-radius: 12px;
  }

  .venue-title {
    font-size: 16px;
    font-weight: 700;
    color: #1e3c72;
    margin-bottom: 12px;
  }

  .venue-list {
    list-style: none;
    padding: 0;
  }

  .venue-item {
    padding: 10px 0;
    font-size: 14px;
    color: #34495e;
    border-bottom: 1px dashed #b8d4e8;
  }

  .venue-item:last-child {
    border-bottom: none;
  }

  .venue-name {
    font-weight: 600;
    color: #2a5298;
  }
```

```
.modal {
  display: none;
  position: fixed;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background: rgba(0, 0, 0, 0.7);
  z-index: 1000;
  align-items: center;
  justify-content: center;
}

.modal.show {
  display: flex;
}

.modal-content {
  background: white;
  border-radius: 20px;
  padding: 40px;
  max-width: 500px;
  width: 90%;
  text-align: center;
  box-shadow: 0 20px 60px rgba(0, 0, 0, 0.4);
}

.success-icon {
  width: 80px;
  height: 80px;
  background: linear-gradient(135deg, #ffd93d, #ffb800);
  border-radius: 50%;
  display: flex;
  align-items: center;
  justify-content: center;
  margin: 0 auto 20px;
  font-size: 40px;
}

.modal-title {
  font-size: 28px;
  font-weight: 700;
  color: #1e3c72;
  margin-bottom: 12px;
}
.modal-message {
  font-size: 16px;
  color: #5a6c7d;
  margin-bottom: 24px;
  line-height: 1.6;
}
```

```
.modal-btn {
  padding: 14px 32px;
  background: linear-gradient(135deg, #1e3c72, #2a5298);
  color: white;
  border: none;
  border-radius: 10px;
  font-size: 16px;
  font-weight: 600;
  cursor: pointer;
  transition: all 0.3s ease;
  margin: 8px;
}

.modal-btn:hover {
  transform: translateY(-2px);
  box-shadow: 0 6px 20px rgba(30, 60, 114, 0.4);
}

.download-btn {
  background: linear-gradient(135deg, #27ae60, #2ecc71);
}

.download-btn:hover {
  box-shadow: 0 6px 20px rgba(46, 204, 113, 0.4);
}

.receipt-container {
  display: none;
  position: fixed;
  left: -9999px;
  background: white;
  width: 800px;
  padding: 40px;
  font-family: 'Inter', Arial, sans-serif;
}

.receipt-header {
  text-align: center;
  border-bottom: 3px solid #1e3c72;
  padding-bottom: 20px;
  margin-bottom: 30px;
}

.receipt-title {
  font-size: 32px;
  font-weight: 800;
  color: #1e3c72;
  margin-bottom: 8px;
}
```

```
.receipt-subtitle {
    font-size: 16px;
    color: #5a6c7d;
  }
  .receipt-info {
    margin-bottom: 30px;
  }
  .receipt-row {
    display: flex;
    justify-content: space-between;
    padding: 12px 0;
    border-bottom: 1px solid #e0e6ed;
  }
  .receipt-label {
    font-weight: 600;
    color: #2c3e50;
  }
  .receipt-value {
    color: #34495e;
  }
  .receipt-total {
    background: #f8f9fa;
    padding: 20px;
    margin-top: 20px;
    border-radius: 10px;
    text-align: center;
  }
  .receipt-total-label {
    font-size: 18px;
    font-weight: 600;
    color: #2c3e50;
    margin-bottom: 8px;
  }
  .receipt-total-amount {
    font-size: 36px;
    font-weight: 800;
    color: #1e3c72;
  }
  .receipt-footer {
    margin-top: 30px;
    padding-top: 20px;
    border-top: 2px solid #e0e6ed;
    text-align: center;
    color: #7f8c8d;
    font-size: 14px;
  }
```

```
.event-costs-info {
    margin-top: 16px;
    padding: 16px;
    background: #f8f9fa;
    border-radius: 10px;
    font-size: 13px;
    color: #5a6c7d;
  }

  .cost-note {
    font-weight: 600;
    color: #2a5298;
    margin-bottom: 8px;
  }

  .registration-card {
    background: white;
    border: 2px solid #e0e6ed;
    border-radius: 12px;
    padding: 20px;
    margin-bottom: 16px;
    transition: all 0.3s ease;
  }

  .registration-card:hover {
    border-color: #2a5298;
    box-shadow: 0 4px 12px rgba(42, 82, 152, 0.1);
  }

  .registration-header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 16px;
    padding-bottom: 12px;
    border-bottom: 2px solid #e0e6ed;
  }

  .registration-receipt-number {
    font-size: 18px;
    font-weight: 700;
    color: #1e3c72;
  }

  .registration-date {
    font-size: 13px;
    color: #7f8c8d;
  }
```

```
.registration-details {
    display: grid;
    grid-template-columns: repeat(2, 1fr);
    gap: 12px;
    margin-bottom: 16px;
  }

  .registration-detail {
    font-size: 14px;
  }

  .registration-detail-label {
    color: #7f8c8d;
    font-weight: 500;
    margin-bottom: 4px;
  }

  .registration-detail-value {
    color: #2c3e50;
    font-weight: 600;
  }

  .registration-footer {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding-top: 12px;
    border-top: 2px solid #e0e6ed;
  }

  .registration-total {
    font-size: 20px;
    font-weight: 800;
    color: #1e3c72;
  }

  .download-single-btn {
    padding: 8px 16px;
    background: linear-gradient(135deg, #27ae60, #2ecc71);
    color: white;
    border: none;
    border-radius: 8px;
    font-size: 14px;
    font-weight: 600;
    cursor: pointer;
    transition: all 0.3s ease;
  }
  .download-single-btn:hover {
    transform: translateY(-2px);
    box-shadow: 0 4px 12px rgba(46, 204, 113, 0.3);
  }
```

```
  /* Responsive */
  @media (max-width: 1024px) {
    .content-grid {
      grid-template-columns: 1fr;
    }

    .summary-card {
      position: static;
    }
  }

  @media (max-width: 640px) {
    body {
      padding: 12px;
    }

    .site-title {
      font-size: 32px;
    }

    .form-container {
      padding: 24px;
    }

    .form-row {
      grid-template-columns: 1fr;
    }

    .radio-group {
      flex-direction: column;
      gap: 12px;
    }
  }
</style>

<!-- QRCode.js (Option B: offline JS QR generation) -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/qrcodejs/1.0.0/qrcode.min.js"></script>

<!-- EMAILJS SDK (for sending emails) -->
<script type="text/javascript" src="https://cdn.jsdelivr.net/npm/emailjs-com@2/dist/email.min.js"></script>
<script>
  // IMPORTANT: Replace the placeholders below with your EmailJS keys
  // emailjs.init('YOUR_EMAILJS_PUBLIC_KEY');
  // Example: emailjs.init('user_xxxxxxx');
  // We'll initialize in the main script after DOM load to keep things explicit.
</script>
</head>
```

```
<body>
  <div class="main-wrapper">
   <header role="banner">
    <h1 class="site-title" id="site_title">Premium Event Registration</h1
    <p class="site-tagline" id="site_tagline">Book Your Perfect Event Experience Today</p>
   </header>
   <main role="main" class="content-grid"><!-- Registration Form -->
    <section class="form-container">
     <h2 class="form-heading" id="form_heading"><span class="form-icon">📋</span> Event Registration
Form</h2>
     <form id="registrationForm" novalidate><!-- Personal Information -->
      <div class="form-row">
       <div class="form-group"><label for="fullName">Full Name <span class="required">*</span></label>
<input type="text" id="fullName" name="fullName" placeholder="Enter your full name" required>
        <div class="error-message" id="error_fullName">
         Please enter your full name
        </div>
       </div>
       <div class="form-group"><label for="email">Email Address <span class="required">*</span></label>
<input type="email" id="email" name="email" placeholder="your.email@example.com" required>
        <div class="error-message" id="error_email">
         Please enter a valid email address
        </div>
       </div>
      </div>
      <div class="form-group"><label for="mobile">Mobile Number <span class="required">*</span></label>
<input type="tel" id="mobile" name="mobile" placeholder="Enter 10-digit mobile number" required>
       <div class="error-message" id="error_mobile">
        Please enter a valid 10-digit mobile number
       </div>
      </div><!-- Event Details -->
      <div class="form-group"><label for="eventType">Event Type <span class="required">*</span></label>
<select id="eventType" name="eventType" required> <option value="">-- Select Event Type --</option>
<option value="Startup Pitch" data-cost="250">Startup Pitch - ₹250/person</option> <option value="Meetup"
data-cost="150">Meetup - ₹150/person</option> <option value="Photography Walk" data-
cost="200">Photography Walk - ₹200/person</option> <option value="Theater Play" data-cost="300">Theater
Play - ₹300/person</option> <option value="Food Festival" data-cost="400">Food Festival -
₹400/person</option> <option value="Dance Show" data-cost="350">Dance Show - ₹350/person</option>
<option value="Art and Craft Fair" data-cost="180">Art and Craft Fair - ₹180/person</option> <option
value="Networking Breakfast" data-cost="220">Networking Breakfast - ₹220/person</option> <option
value="Hackathon" data-cost="500">Hackathon - ₹500/person</option> <option value="Tech Fest" data-
cost="450">Tech Fest - ₹450/person</option> <option value="Conference" data-cost="600">Conference -
₹600/person</option> <option value="Seminar" data-cost="280">Seminar - ₹280/person</option> <option
value="Workshop" data-cost="320">Workshop - ₹320/person</option> <option value="Webinar" data-
cost="150">Webinar - ₹150/person</option> </select>
       <div class="error-message" id="error_eventType">
        Please select an event type
       </div>
      </div>
```

```
<div class="form-group"><label for="eventDate">Event Date <span class="required">*</span></label> <input type="date" id="eventDate" name="eventDate" required>
      <div class="error-message" id="error_eventDate">
       Please select an event date
      </div>
     </div>
     <div class="form-group"><label for="venue">Select Venue <span class="required">*</span></label> <select id="venue" name="venue" required> <option value="">-- Select Venue --</option> <option value="Grand Royale Banquet Hall, MG Road, Sector 14" data-base-cost="5000">Grand Royale Banquet Hall - MG Road, Sector 14 (Base: ₹5,000)</option> <option value="Crystal Palace Convention Center, Jubilee Hills, Phase 2" data-base-cost="8000">Crystal Palace Convention Center - Jubilee Hills, Phase 2 (Base: ₹8,000)</option> <option value="The Heritage Auditorium, Old City, Heritage Lane" data-base-cost="4500">The Heritage Auditorium - Old City, Heritage Lane (Base: ₹4,500)</option> <option value="Skyline Events Arena, Tech Park, IT Corridor" data-base-cost="7000">Skyline Events Arena - Tech Park, IT Corridor (Base: ₹7,000)</option> <option value="Emerald Gardens Function Hall, Green Valley, Ring Road" data-base-cost="6000">Emerald Gardens Function Hall - Green Valley, Ring Road (Base: ₹6,000)</option> <option value="Metropolitan Conference Hall, Business District, Tower 3" data-base-cost="9000">Metropolitan Conference Hall - Business District, Tower 3 (Base: ₹9,000)</option> <option value="Sapphire Celebration Center, Lake View, Waterfront" data-base-cost="7500">Sapphire Celebration Center - Lake View, Waterfront (Base: ₹7,500)</option> <option value="Imperial Ballroom &amp;amp; Events, Downtown, Central Plaza" data-base-cost="10000">Imperial Ballroom &amp;amp; Events - Downtown, Central Plaza (Base: ₹10,000)</option> <option value="Platinum Pavilion, Airport Road, Mile 5" data-base-cost="6500">Platinum Pavilion - Airport Road, Mile 5 (Base: ₹6,500)</option> <option value="Majestic Halls &amp;amp; Conventions, University Area, Campus Drive" data-base-cost="5500">Majestic Halls &amp;amp; Conventions - University Area, Campus Drive (Base: ₹5,500)</option> </select>
      <div class="error-message" id="error_venue">
       Please select a venue
      </div>
      <div style="margin-top: 8px; font-size: 12px; color: #5a6c7d; font-style: italic;">
       Note: Final venue cost = Base cost + (Duration multiplier × Base cost)
      </div>
     </div>
     <div class="form-group"><label for="duration">Event Duration <span class="required">*</span></label> <select id="duration" name="duration" required> <option value="">-- Select Duration --</option> <option value="1 Hour" data-multiplier="0.3">1 Hour (30% of base cost)</option> <option value="2 Hours" data-multiplier="0.5">2 Hours (50% of base cost)</option> <option value="3 Hours" data-multiplier="0.7">3 Hours (70% of base cost)</option> <option value="Half Day (4-5 Hours)" data-multiplier="1.0">Half Day 4-5 Hours (100% of base cost)</option> <option value="Full Day (9 AM - 5 PM)" data-multiplier="1.5">Full Day 9 AM - 5 PM (150% of base cost)</option> </select>
      <div class="error-message" id="error_duration">
       Please select event duration
      </div>
      <div style="margin-top: 8px; font-size: 12px; color: #5a6c7d; font-style: italic;">
       Duration cost is calculated as a percentage of the venue's base cost
      </div>
     </div>
```

```
  <div class="form-group"><label for="attendees">Number of Attendees <span
class="required">*</span></label> <input type="number" id="attendees" name="attendees" min="1"
placeholder="Enter number of people attending" required>
      <div class="error-message" id="error_attendees">
       Please enter number of attendees (minimum 1)
      </div>
     </div>
     <div class="form-row">
      <div class="form-group"><label for="costPerPerson">Cost Per Person (Event)</label> <input type="text"
id="costPerPerson" name="costPerPerson" value="₹150" readonly style="background: #f0f0f0; cursor: not-
allowed;">
      </div>
      <div class="form-group"><label for="venueCost">Venue Cost (Duration)</label> <input type="text"
id="venueCost" name="venueCost" value="₹0" readonly style="background: #f0f0f0; cursor: not-allowed;">
      </div>
     </div><!-- Payment Mode -->
     <div class="form-group"><label>Payment Mode <span class="required">*</span></label>
      <div class="radio-group">
       <div class="radio-option"><input type="radio" id="upi" name="paymentMode" value="UPI" required>
<label for="upi" class="radio-label">UPI Payment</label>
       </div>
       <div class="radio-option"><input type="radio" id="cod" name="paymentMode" value="Cash on
Delivery"> <label for="cod" class="radio-label">Cash on Delivery</label>
       </div>
      </div>
      <div class="error-message" id="error_paymentMode">
       Please select a payment mode
      </div>
     </div><!-- UPI Section -->
     <div class="upi-section" id="upiSection">
      <div class="venue-title">
       UPI Payment Details
      </div>
      <div class="upi-id-display" id="upiIdDisplay">
       7013539146@ibl
      </div><!-- QR Code -->
      <div style="text-align: center; margin: 20px 0;">
       <div style="background: white; padding: 16px; border-radius: 12px; display: inline-block; box-shadow: 0
4px 12px rgba(0,0,0,0.1);">
        <!-- QR will be inserted here by QRCode.js -->
        <div id="qrHolder" style="width:200px; height:200px;"></div>
       </div>
       <p style="font-size: 13px; color: #5a6c7d; margin-top: 12px; font-weight: 600;">Scan QR Code to
Pay</p>
```

```
 </div>
      <p class="upi-instruction">Scan the QR code above OR click the button below to open your UPI app</p><button type="button" class="submit-btn" id="payNowBtn" style="margin-top: 16px; background: linear-gradient(135deg, #27ae60, #2ecc71);"> 💳 Open UPI App to Pay </button>
      <div style="margin-top: 16px; padding: 12px; background: rgba(255, 255, 255, 0.7); border-radius: 8px; font-size: 13px; color: #2c3e50;"><strong>Amount to Pay:</strong> <span id="upiAmountDisplay" style="font-size: 18px; font-weight: 700; color: #1e3c72;">₹0</span>
      </div>
     </div><button type="submit" class="submit-btn" id="submit_btn"> <span id="submit_button_text">Complete Registration &amp;amp; Payment</span> </button>
    </form>
   </section><!-- Summary Card -->
   <aside class="summary-card">
    <h3 class="summary-title">📊 Booking Summary</h3>
    <div class="summary-item">
     <div class="summary-label">
      Full Name
     </div>
     <div class="summary-value" id="summary_name">
      —
     </div>
    </div>
    <div class="summary-item">
     <div class="summary-label">
      Email
     </div>
     <div class="summary-value" id="summary_email">
      —
     </div>
    </div>
    <div class="summary-item">
     <div class="summary-label">
      Mobile
     </div>
     <div class="summary-value" id="summary_mobile">
      —
     </div>
    </div>
    <div class="summary-item">
     <div class="summary-label">
      Event Type
     </div>
     <div class="summary-value" id="summary_event">
      —
     </div>
    </div>
    <div class="summary-item">
     <div class="summary-label">
      Event Date
     </div>
     <div class="summary-value" id="summary_date">
```

```
 </div>
</div>
<div class="summary-item">
 <div class="summary-label">
  Selected Venue
 </div>
 <div class="summary-value" id="summary_venue">
  —
 </div>
</div>
<div class="summary-item">
 <div class="summary-label">
  Event Duration
 </div>
 <div class="summary-value" id="summary_duration">
  —
 </div>
</div>
<div class="summary-item">
 <div class="summary-label">
  Number of Attendees
 </div>
 <div class="summary-value" id="summary_attendees">
  —
 </div>
</div>
<div class="summary-item">
 <div class="summary-label">
  Cost Per Person (Event)
 </div>
 <div class="summary-value" id="summary_cost">
  ₹150
</div>
<div class="summary-item">
 <div class="summary-label">
  Venue Cost
 </div>
 <div class="summary-value" id="summary_venue_cost">
  ₹0
 </div>
</div>
<div class="summary-item">
 <div class="summary-label">
  Payment Mode
 </div>
 <div class="summary-value" id="summary_payment">
  —
 </div>
</div>
```

```
<div class="total-amount">
     <div class="total-label">
      Total Amount
     </div>
     <div class="total-value" id="total_amount">
      ₹0
     </div>
     <div style="font-size: 12px; margin-top: 8px; opacity: 0.9;">
      (Event Cost + Venue Cost)
     </div>
    </div>
   </aside>
  </main><!-- Organizer Dashboard -->
  <section style="margin-top: 40px; padding: 32px; background: rgba(255, 255, 255, 0.98); border-radius:
20px; box-shadow: 0 15px 50px rgba(0, 0, 0, 0.25);">
   <div style="display: flex; justify-content: space-between; align-items: center; margin-bottom: 24px;">
    <h2 style="font-size: 28px; font-weight: 700; color: #1e3c72; display: flex; align-items: center; gap:
12px;"><span style="width: 40px; height: 40px; background: linear-gradient(135deg, #ffd93d, #ffb800);
border-radius: 10px; display: flex; align-items: center; justify-content: center; font-size: 20px;">📊</span>
Organizer Dashboard - All Registrations</h2>
   </div>
   <div style="margin-bottom: 20px; padding: 16px; background: linear-gradient(135deg, #e8f4f8, #d4e9f7);
border-radius: 12px;">
    <div style="display: flex; justify-content: space-around; text-align: center;">
     <div>
      <div style="font-size: 32px; font-weight: 800; color: #1e3c72;" id="totalRegistrations">
       0
      </div>
      <div style="font-size: 14px; color: #5a6c7d; font-weight: 600;">
       Total Registrations
      </div>
     </div>
     <div>
      <div style="font-size: 32px; font-weight: 800; color: #1e3c72;" id="totalRevenue">
       ₹0
      </div>
      <div style="font-size: 14px; color: #5a6c7d; font-weight: 600;">
       Total Revenue
      </div>
     </div>
     <div>
      <div style="font-size: 32px; font-weight: 800; color: #1e3c72;" id="totalAttendees">
       0
      </div>
      <div style="font-size: 14px; color: #5a6c7d; font-weight: 600;">
       Total Attendees
      </div>
     </div>
    </div>
   </div>
   <div id="registrationsList" style="margin-top: 24px;">
```

```
    <p style="text-align: center; color: #7f8c8d; padding: 40px;">No registrations yet. Registrations will appear here automatically.</p>
   </div>
  </section><!-- Footer -->
  <footer role="contentinfo" style="margin-top: 40px; padding: 30px; background: rgba(255, 255, 255, 0.95); border-radius: 20px; text-align: center; box-shadow: 0 10px 40px rgba(0, 0, 0, 0.2);">
   <h3 style="font-size: 20px; font-weight: 700; color: #1e3c72; margin-bottom: 16px;">📞 Contact Us</h3>
   <p style="font-size: 16px; color: #2c3e50; margin-bottom: 8px;"><strong>Phone:</strong> <a href="tel:7013539146" style="color: #2a5298; text-decoration: none; font-weight: 600;">7013539146</a></p>
   <p style="font-size: 16px; color: #2c3e50; margin-bottom: 16px;"><strong>Email:</strong> <a href="mailto:sagarika7013@gmail.com" style="color: #2a5298; text-decoration: none; font-weight: 600;">sagarika7013@gmail.com</a></p>
   <p style="font-size: 14px; color: #7f8c8d; margin-top: 12px;">For any queries or assistance with your event registration, feel free to reach out to us!</p>
  </footer>
 </div><!-- Success Modal -->
 <div class="modal" id="successModal" role="dialog" aria-labelledby="modal_title" aria-modal="true">
  <div class="modal-content">
   <div class="success-icon">
    ✓
   </div>
   <h2 class="modal-title" id="modal_title">Registration Successful!</h2>
   <p class="modal-message" id="success_message">Thank you for registering! Your receipt has been automatically generated. Confirmation emails with receipt have been sent to both you and our event organizer (sagarika7013@gmail.com).</p><button class="modal-btn download-btn" id="downloadReceipt">📥 Download Receipt</button> <button class="modal-btn" id="closeModal">Close</button>
  </div>
 </div><!-- Hidden Receipt Template -->
 <div class="receipt-container" id="receiptContainer">
  <div class="receipt-header">
   <div class="receipt-title">
    EVENT REGISTRATION RECEIPT
   </div>
   <div class="receipt-subtitle">
    Premium Event Registration System
   </div>
   <div style="margin-top: 12px; font-size: 14px; color: #7f8c8d;">
    Receipt #<span id="receipt_number"></span> | Date: <span id="receipt_date"></span>
   </div>
  </div>
  <div class="receipt-info">
   <h3 style="color: #1e3c72; margin-bottom: 16px; font-size: 20px;">Customer Information</h3>
   <div class="receipt-row"><span class="receipt-label">Full Name:</span> <span class="receipt-value" id="receipt_fullname"></span>
   </div>
   <div class="receipt-row"><span class="receipt-label">Email:</span> <span class="receipt-value" id="receipt_email"></span>
   </div>
```

```
<div class="receipt-row"><span class="receipt-label">Mobile:</span> <span class="receipt-value"
id="receipt_mobile"></span>

</div>
  </div>
  <div class="receipt-info">
   <h3 style="color: #1e3c72; margin-bottom: 16px; font-size: 20px;">Event Details</h3>
   <div class="receipt-row"><span class="receipt-label">Event Type:</span> <span class="receipt-value"
id="receipt_event"></span>
   </div>
   <div class="receipt-row"><span class="receipt-label">Event Date:</span> <span class="receipt-value"
id="receipt_eventdate"></span>
   </div>
   <div class="receipt-row"><span class="receipt-label">Venue:</span> <span class="receipt-value"
id="receipt_venue"></span>
   </div>
   <div class="receipt-row"><span class="receipt-label">Duration:</span> <span class="receipt-value"
id="receipt_duration"></span>
   </div>
   <div class="receipt-row"><span class="receipt-label">Number of Attendees:</span> <span class="receipt-
value" id="receipt_attendees"></span>
   </div>
  </div>
  <div class="receipt-info">
   <h3 style="color: #1e3c72; margin-bottom: 16px; font-size: 20px;">Payment Details</h3>
   <div class="receipt-row"><span class="receipt-label">Cost Per Person:</span> <span class="receipt-value"
id="receipt_costperperson"></span>
   </div>
   <div class="receipt-row"><span class="receipt-label">Event Cost:</span> <span class="receipt-value"
id="receipt_eventcost"></span>
   </div>
   <div class="receipt-row"><span class="receipt-label">Venue Cost:</span> <span class="receipt-value"
id="receipt_venuecost"></span>
   </div>
   <div class="receipt-row"><span class="receipt-label">Payment Mode:</span> <span class="receipt-value"
id="receipt_payment"></span>
   </div>
   <div class="receipt-row" id="upi_row" style="display: none;"><span class="receipt-label">UPI ID:</span>
<span class="receipt-value">7013539146@ibl</span>
   </div>
  </div>
  <div class="receipt-total">
   <div class="receipt-total-label">
    Total Amount Paid
   </div>
   <div class="receipt-total-amount" id="receipt_total"></div>
  </div>
  <div class="receipt-footer">
   <p style="margin-bottom: 12px; font-weight: 600; color: #2c3e50;">Contact Information</p>
   <p>Phone: 7013539146 | Email: sagarika7013@gmail.com</p>
   <p style="margin-top: 16px;">Thank you for choosing our event registration service!</p>
```

```
    <p style="margin-top: 8px; font-size: 12px;">This is a computer-generated receipt and does not require a signature.</p>
   </div>
  </div>

  <!-- ------------------- SCRIPT: logic, QR, EmailJS integration ------------------- -->
  <script>
   (function() {
    // --------------------------
    // Basic utilities and state
    // --------------------------
    let allRegistrations = []; // in-memory store; Data SDK will override if available
    let qrInstance = null;
    let currentReceiptData = null;
    let upiClicked = false;

    // DOM refs
    const form = document.getElementById('registrationForm');
    const fullNameInput = document.getElementById('fullName');
    const emailInput = document.getElementById('email');
    const mobileInput = document.getElementById('mobile');
    const eventTypeSelect = document.getElementById('eventType');
    const eventDateInput = document.getElementById('eventDate');
    const venueSelect = document.getElementById('venue');
    const durationSelect = document.getElementById('duration');
    const attendeesInput = document.getElementById('attendees');
    const costPerPersonInput = document.getElementById('costPerPerson');
    const venueCostInput = document.getElementById('venueCost');
    const upiRadio = document.getElementById('upi');
    const codRadio = document.getElementById('cod');
    const upiSection = document.getElementById('upiSection');
    const submitBtn = document.getElementById('submit_btn');
    const qrHolder = document.getElementById('qrHolder');
    const upiIdDisplay = document.getElementById('upiIdDisplay');
    const upiAmountDisplay = document.getElementById('upiAmountDisplay');

    // summary refs
    const summaryName = document.getElementById('summary_name');
    const summaryEmail = document.getElementById('summary_email');
    const summaryMobile = document.getElementById('summary_mobile');
    const summaryEvent = document.getElementById('summary_event');
    const summaryDate = document.getElementById('summary_date');
    const summaryVenue = document.getElementById('summary_venue');
    const summaryDuration = document.getElementById('summary_duration');
    const summaryAttendees = document.getElementById('summary_attendees');
    const summaryCost = document.getElementById('summary_cost');
    const summaryVenueCost = document.getElementById('summary_venue_cost');
    const summaryPayment = document.getElementById('summary_payment');
    const totalAmount = document.getElementById('total_amount');
    const successModal = document.getElementById('successModal');
    const closeModalBtn = document.getElementById('closeModal');
```

```
  const downloadReceiptBtn = document.getElementById('downloadReceipt');

    // set min date to today
    const today = new Date().toISOString().split('T')[0];
    eventDateInput.setAttribute('min', today);

    // Initialize EmailJS if user set key in global (replace below placeholder)
    // emailjs.init('YOUR_EMAILJS_PUBLIC_KEY'); // <-- uncomment and set your key

    // ---------------------------
    // QR helper (QRCode.js)
    // ---------------------------
    function renderQRCode(upiString) {
      // clear existing
      qrHolder.innerHTML = '';
      try {
        qrInstance = new QRCode(qrHolder, {
          text: upiString,
          width: 200,
          height: 200,
          correctLevel: QRCode.CorrectLevel.H
        });
      } catch (err) {
        console.error('QR generation failed', err);
      }
    }

    // Build UPI URI string (standard)
    function buildUpiString(amount) {
      const pa = encodeURIComponent('7013539146@ibl');
      const pn = encodeURIComponent('Event Host');
      const am = encodeURIComponent(amount);
      const cu = 'INR';
      const tn = encodeURIComponent('Event Registration Payment');
      return `upi://pay?pa=${pa}&pn=${pn}&am=${am}&cu=${cu}&tn=${tn}`;
    }

    // ---------------------------
    // Calculation & summary
    // ---------------------------
    function getSelectedCostPerPerson() {
      const selectedEvent = eventTypeSelect.options[eventTypeSelect.selectedIndex];
      return selectedEvent && selectedEvent.dataset && selectedEvent.dataset.cost ?
parseInt(selectedEvent.dataset.cost, 10) : 150;
    }

    function getVenueBaseCost() {
      const selectedVenue = venueSelect.options[venueSelect.selectedIndex];
      return selectedVenue && selectedVenue.dataset && selectedVenue.dataset.baseCost ?
parseInt(selectedVenue.dataset.baseCost, 10) : 0;
    }
```

```
    function getDurationMultiplier() {
      const selectedDuration = durationSelect.options[durationSelect.selectedIndex];
      return selectedDuration && selectedDuration.dataset && selectedDuration.dataset.multiplier ?
parseFloat(selectedDuration.dataset.multiplier) : 0;
    }

    function calculateCosts() {
      const costPerPerson = getSelectedCostPerPerson();
      const attendees = Math.max(0, parseInt(attendeesInput.value || '0', 10));
      const eventCost = costPerPerson * attendees;

      const venueBase = getVenueBaseCost();
      const durationMultiplier = getDurationMultiplier();
      const venueCost = Math.round(venueBase * durationMultiplier);

      const total = eventCost + venueCost;
      return {
        costPerPerson,
        attendees,
        eventCost,
        venueCost,
        total
      };
    }

    function updateSummary() {
      summaryName.textContent = fullNameInput.value.trim() || '—';
      summaryEmail.textContent = emailInput.value.trim() || '—';
      summaryMobile.textContent = mobileInput.value.trim() || '—';

      const selectedEvent = eventTypeSelect.value || '—';
      summaryEvent.textContent = selectedEvent;

      if (eventDateInput.value) {
        const date = new Date(eventDateInput.value);
        summaryDate.textContent = date.toLocaleDateString('en-IN', {
          day: 'numeric',
          month: 'long',
          year: 'numeric'
        });
      } else {
        summaryDate.textContent = '—';
      }

      summaryVenue.textContent = venueSelect.value || '—';
      summaryDuration.textContent = durationSelect.value || '—';
      summaryAttendees.textContent = attendeesInput.value || '—';
      const paymentMode = document.querySelector('input[name="paymentMode"]:checked');
      summaryPayment.textContent = paymentMode ? paymentMode.value : '—';
      const costs = calculateCosts();
      summaryCost.textContent = '₹' + costs.costPerPerson;
```

```
    costPerPersonInput.value = '₹' + costs.costPerPerson;
     summaryVenueCost.textContent = '₹' + costs.venueCost.toLocaleString('en-IN');
     venueCostInput.value = '₹' + costs.venueCost.toLocaleString('en-IN');
     totalAmount.textContent = '₹' + costs.total.toLocaleString('en-IN');

     upiAmountDisplay.textContent = '₹' + costs.total.toLocaleString('en-IN');

     // Generate QR when UPI is selected and total > 0
     if (upiRadio.checked && costs.total > 0) {
       const ustr = buildUpiString(costs.total);
       renderQRCode(ustr);
     } else {
       // clear QR
       qrHolder.innerHTML = '';
     }
   }

   // ---------------------------
   // Validation
   // ---------------------------
   function validateEmail(email) {
     const re = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
     return re.test(email);
   }

   function validateMobile(mobile) {
     const re = /^[6-9]\d{9}$/;
     return re.test(mobile);
   }

   function showError(fieldId, show) {
     const field = document.getElementById(fieldId);
     const error = document.getElementById('error_' + fieldId);
     if (show) {
       if (field) field.classList.add('error');
       if (error) error.classList.add('show');
     } else {
       if (field) field.classList.remove('error');
       if (error) error.classList.remove('show');
     }
   }

   function validateForm() {
     let isValid = true;
     if (!fullNameInput.value.trim()) { showError('fullName', true); isValid = false; } else {
showError('fullName', false); }
     if (!emailInput.value.trim() || !validateEmail(emailInput.value.trim())) { showError('email', true); isValid =
false; } else { showError('email', false); }
     if (!mobileInput.value.trim() || !validateMobile(mobileInput.value.trim())) { showError('mobile', true);
isValid = false; } else { showError('mobile', false); }
```

```
if (!eventTypeSelect.value) { showError('eventType', true); isValid = false; } else { showError('eventType', false); }
      if (!eventDateInput.value) { showError('eventDate', true); isValid = false; } else { showError('eventDate', false); }
      if (!venueSelect.value) { showError('venue', true); isValid = false; } else { showError('venue', false); }
      if (!durationSelect.value) { showError('duration', true); isValid = false; } else { showError('duration', false); }
      if (!attendeesInput.value || parseInt(attendeesInput.value, 10) < 1) { showError('attendees', true); isValid = false; } else { showError('attendees', false); }
      const paymentMode = document.querySelector('input[name="paymentMode"]:checked');
      if (!paymentMode) { showError('paymentMode', true); isValid = false; } else { showError('paymentMode', false); }
      return isValid;
    }

    // ---------------------------
    // Inline message helper
    // ---------------------------
    function showInlineMessage(message, duration = 3000) {
      const messageDiv = document.createElement('div');
      messageDiv.textContent = message;
      messageDiv.style.cssText = 'position: fixed; top: 20px; right: 20px; background: #1e3c72; color: white; padding: 16px 24px; border-radius: 10px; font-weight: 600; z-index: 9999; box-shadow: 0 4px 12px rgba(0,0,0,0.3);';
      document.body.appendChild(messageDiv);
      setTimeout(() => {
        messageDiv.style.opacity = '0';
        messageDiv.style.transition = 'opacity 0.3s ease';
        setTimeout(() => messageDiv.remove(), 300);
      }, duration);
    }

    // ---------------------------
    // Receipt generation & download
    // ---------------------------
    function generateAndDownloadReceipt(registration) {
      const receiptHTML = `<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Receipt #${registration.receipt_number}</title>
  <style>
    body { font-family: Arial, sans-serif; padding: 40px; max-width: 800px; margin: 0 auto; }
    .receipt-header { text-align: center; border-bottom: 3px solid #1e3c72; padding-bottom: 20px; margin-bottom: 30px; }
    .receipt-title { font-size: 32px; font-weight: 800; color: #1e3c72; margin-bottom: 8px; }
    .receipt-subtitle { font-size: 16px; color: #5a6c7d; }
    .receipt-info { margin-bottom: 30px; }
    .receipt-row { display: flex; justify-content: space-between; padding: 12px 0; border-bottom: 1px solid #e0e6ed; }
```

```
 .receipt-label { font-weight: 600; color: #2c3e50; }
   .receipt-value { color: #34495e; }
   .receipt-total { background: #f8f9fa; padding: 20px; margin-top: 20px; border-radius: 10px; text-align: center;
}
   .receipt-total-label { font-size: 18px; font-weight: 600; color: #2c3e50; margin-bottom: 8px; }
   .receipt-total-amount { font-size: 36px; font-weight: 800; color: #1e3c72; }
   .receipt-footer { margin-top: 30px; padding-top: 20px; border-top: 2px solid #e0e6ed; text-align: center;
color: #7f8c8d; font-size: 14px; }
   h3 { color: #1e3c72; margin-bottom: 16px; font-size: 20px; }
  </style>
</head>
<body>
  <div class="receipt-header">
   <div class="receipt-title">EVENT REGISTRATION RECEIPT</div>
   <div class="receipt-subtitle">Premium Event Registration System</div>
   <div style="margin-top: 12px; font-size: 14px; color: #7f8c8d;">
    Receipt #${registration.receipt_number} | Date: ${registration.registration_date}
   </div>
  </div>

  <div class="receipt-info">
   <h3>Customer Information</h3>
   <div class="receipt-row">
    <span class="receipt-label">Full Name:</span>
    <span class="receipt-value">${registration.full_name}</span>
   </div>
   <div class="receipt-row">
    <span class="receipt-label">Email:</span>
    <span class="receipt-value">${registration.email}</span>
   </div>
   <div class="receipt-row">
    <span class="receipt-label">Mobile:</span>
    <span class="receipt-value">${registration.mobile}</span>
   </div>
  </div>

  <div class="receipt-info">
   <h3>Event Details</h3>
   <div class="receipt-row">
    <span class="receipt-label">Event Type:</span>
    <span class="receipt-value">${registration.event_type}</span>
   </div>
   <div class="receipt-row">
    <span class="receipt-label">Event Date:</span>
    <span class="receipt-value">${registration.event_date}</span>
   </div>
   <div class="receipt-row">
    <span class="receipt-label">Venue:</span>
    <span class="receipt-value">${registration.venue}</span>
   </div>
```

```
 <div class="receipt-row">
     <span class="receipt-label">Duration:</span>
     <span class="receipt-value">${registration.duration}</span>
   </div>
   <div class="receipt-row">
     <span class="receipt-label">Number of Attendees:</span>
     <span class="receipt-value">${registration.attendees}</span>
   </div>
 </div>

 <div class="receipt-info">
   <h3>Payment Details</h3>
   <div class="receipt-row">
     <span class="receipt-label">Cost Per Person:</span>
     <span class="receipt-value">₹${registration.cost_per_person}</span>
   </div>
   <div class="receipt-row">
     <span class="receipt-label">Event Cost:</span>
     <span class="receipt-value">₹${registration.event_cost.toLocaleString('en-IN')}</span>
   </div>
   <div class="receipt-row">
     <span class="receipt-label">Venue Cost:</span>
     <span class="receipt-value">₹${registration.venue_cost.toLocaleString('en-IN')}</span>
   </div>
   <div class="receipt-row">
     <span class="receipt-label">Payment Mode:</span>
     <span class="receipt-value">${registration.payment_mode}</span>
   </div>
   ${registration.payment_mode && registration.payment_mode.toString().toLowerCase().includes('upi') ?
'<div class="receipt-row"><span class="receipt-label">UPI ID:</span><span class="receipt-
value">7013539146@ibl</span></div>' : ''}
 </div>

 <div class="receipt-total">
   <div class="receipt-total-label">Total Amount Paid</div>
   <div class="receipt-total-amount">₹${registration.total_amount.toLocaleString('en-IN')}</div>
 </div>

 <div class="receipt-footer">
   <p style="margin-bottom: 12px; font-weight: 600; color: #2c3e50;">Contact Information</p>
   <p>Phone: 7013539146 | Email: sagarika7013@gmail.com</p>
   <p style="margin-top: 16px;">Thank you for choosing our event registration service!</p>
   <p style="margin-top: 8px; font-size: 12px;">This is a computer-generated receipt and does not require a
signature.</p>
 </div>
</body>
</html>`
       const blob = new Blob([receiptHTML], { type: 'text/html' });
       const url = URL.createObjectURL(blob);
       const a = document.createElement('a');
```

```
    a.href = url;
    const safeName = (registration.full_name || 'user').replace(/\s+/g, '_').replace(/[^\w_\-]/g, '');
    a.download = 'Receipt_' + registration.receipt_number + '_' + safeName + '.html';
    document.body.appendChild(a);
    a.click();
    a.remove();
    URL.revokeObjectURL(url);
  }

  // ---------------------------
  // Dashboard update
  // ---------------------------
  function updateDashboard() {
    const totalRegistrations = allRegistrations.length;
    const totalRevenue = allRegistrations.reduce((sum, reg) => sum + (reg.total_amount || 0), 0);
    const totalAttendees = allRegistrations.reduce((sum, reg) => sum + (reg.attendees || 0), 0);

    document.getElementById('totalRegistrations').textContent = totalRegistrations;
    document.getElementById('totalRevenue').textContent = '₹' + totalRevenue.toLocaleString('en-IN');
    document.getElementById('totalAttendees').textContent = totalAttendees;

    const registrationsList = document.getElementById('registrationsList');

    if (totalRegistrations === 0) {
      registrationsList.innerHTML = '<p style="text-align: center; color: #7f8c8d; padding: 40px;">No
registrations yet. Registrations will appear here automatically.</p>';
      return;
    }

    const sortedRegistrations = [...allRegistrations].sort((a, b) => b.timestamp - a.timestamp);

    registrationsList.innerHTML = sortedRegistrations.map(reg => `
      <div class="registration-card">
        <div class="registration-header">
          <div>
            <div class="registration-receipt-number">Receipt #${reg.receipt_number}</div>
            <div class="registration-date">${reg.registration_date}</div>
          </div>
          <button class="download-single-btn" data-backendid="${reg.__backendId || reg.receipt_number}">
            📥 Download
          </button>
        </div>
        <div class="registration-details">
          <div class="registration-detail">
            <div class="registration-detail-label">Name</div>
            <div class="registration-detail-value">${reg.full_name}</div>
          </div>
          <div class="registration-detail">
            <div class="registration-detail-label">Email</div>
            <div class="registration-detail-value">${reg.email}</div>
          </div>
```

```
  <div class="registration-detail">
        <div class="registration-detail-label">Mobile</div>
        <div class="registration-detail-value">${reg.mobile}</div>
      </div>
      <div class="registration-detail">
        <div class="registration-detail-label">Event Type</div>
        <div class="registration-detail-value">${reg.event_type}</div>
      </div>
      <div class="registration-detail">
        <div class="registration-detail-label">Event Date</div>
        <div class="registration-detail-value">${reg.event_date}</div>
      </div>
      <div class="registration-detail">
        <div class="registration-detail-label">Venue</div>
        <div class="registration-detail-value">${reg.venue}</div>
      </div>
      <div class="registration-detail">
        <div class="registration-detail-label">Duration</div>
        <div class="registration-detail-value">${reg.duration}</div>
      </div>
      <div class="registration-detail">
        <div class="registration-detail-label">Attendees</div>
        <div class="registration-detail-value">${reg.attendees}</div>
      </div>
      <div class="registration-detail">
        <div class="registration-detail-label">Payment Mode</div>
        <div class="registration-detail-value">${reg.payment_mode}</div>
      </div>
    </div>
    <div class="registration-footer">
      <div class="registration-total">Total: ₹${reg.total_amount.toLocaleString('en-IN')}</div>
    </div>
  </div>
 `).join('');

 // attach click handlers to download buttons
 document.querySelectorAll('.download-single-btn').forEach(btn => {
  btn.removeEventListener('click', handleDownloadSingleBtn);
  btn.addEventListener('click', handleDownloadSingleBtn);
 });
}

function handleDownloadSingleBtn(e) {
 const id = e.currentTarget.getAttribute('data-backendid');
 // find by backend id or receipt number
 const registration = allRegistrations.find(r => r.__backendId === id || r.receipt_number === id);
 if (!registration) {
  showInlineMessage('Registration not found.');
  return;
 }
```

```
 generateAndDownloadReceipt(registration);
    }

    // ---------------------------
    // Email sending (EmailJS) — placeholders
    // ---------------------------
    async function sendEmails(registrationData) {
      // Update: you must replace SERVICE_ID, TEMPLATE_CUSTOMER, TEMPLATE_ORGANIZER with your EmailJS values.
      const SERVICE_ID = 'YOUR_EMAILJS_SERVICE_ID';
      const TEMPLATE_CUSTOMER = 'YOUR_EMAILJS_TEMPLATE_CUSTOMER';
      const TEMPLATE_ORGANIZER = 'YOUR_EMAILJS_TEMPLATE_ORGANIZER';

      // short HTML for email body
      const receiptBasicHTML = `
        <h2>Event Registration Receipt</h2>
        <p><strong>Receipt #:</strong> ${registrationData.receipt_number}</p>
        <p><strong>Date:</strong> ${registrationData.registration_date}</p>
        <h3>Customer</h3>
        <p>${registrationData.full_name} — ${registrationData.email} — ${registrationData.mobile}</p>
        <h3>Event</h3>
        <p>${registrationData.event_type} on ${registrationData.event_date}</p>
        <p>Venue: ${registrationData.venue} — Duration: ${registrationData.duration}</p>
        <p>Attendees: ${registrationData.attendees}</p>
        <h3>Payment</h3>
        <p>Cost per person: ₹${registrationData.cost_per_person}</p>
        <p>Event cost: ₹${registrationData.event_cost.toLocaleString('en-IN')}</p>
        <p>Venue cost: ₹${registrationData.venue_cost.toLocaleString('en-IN')}</p>
        <p><strong>Total: ₹${registrationData.total_amount.toLocaleString('en-IN')}</strong></p>
        <p>Payment mode: ${registrationData.payment_mode}</p>
      `;

      const customerParams = {
        to_email: registrationData.email,
        to_name: registrationData.full_name,
        receipt_html: receiptBasicHTML,
        receipt_number: registrationData.receipt_number,
        registration_date: registrationData.registration_date,
        total_amount: '₹' + registrationData.total_amount.toLocaleString('en-IN'),
      };

      const organizerParams = {
        admin_email: 'sagarika7013@gmail.com',
        customer_name: registrationData.full_name,
        customer_email: registrationData.email,
        customer_mobile: registrationData.mobile,
        receipt_html: receiptBasicHTML,
        receipt_number: registrationData.receipt_number,
        total_amount: '₹' + registrationData.total_amount.toLocaleString('en-IN'),
      };
```

```
// If EmailJS not configured, reject early
      if (!window.emailjs || SERVICE_ID.startsWith('YOUR_')) {
        return Promise.reject('EmailJS not configured. Replace placeholders with your EmailJS service/template IDs and call emailjs.init(USER_KEY).');
      }

      const p1 = emailjs.send(SERVICE_ID, TEMPLATE_CUSTOMER, customerParams);
      const p2 = emailjs.send(SERVICE_ID, TEMPLATE_ORGANIZER, organizerParams);
      return Promise.all([p1, p2]);
    }

    // ---------------------------
    // Data SDK integration (if available)
    // ---------------------------
    const dataHandler = {
      onDataChanged(data) {
        // expects array of registration objects
        if (Array.isArray(data)) {
          allRegistrations = data;
          updateDashboard();
        }
      }
    };

    (async function initDataSdk() {
      if (window.dataSdk) {
        try {
          const result = await window.dataSdk.init(dataHandler);
          if (!result.isOk) {
            console.warn('Data SDK init returned not OK');
          } else {
            // Load initial data if present
            if (result.initialData && Array.isArray(result.initialData)) {
              allRegistrations = result.initialData;
              updateDashboard();
            }
          }
        } catch (err) {
          console.warn('Data SDK init error', err);
        }
      }
    })();

    // ---------------------------
    // Event handlers & submission
    // ---------------------------
    // interactive updates
    [fullNameInput, emailInput, mobileInput].forEach(el => el.addEventListener('input', updateSummary));
    [eventTypeSelect, eventDateInput, venueSelect, durationSelect, attendeesInput].forEach(el => el.addEventListener('change', updateSummary));
    attendeesInput.addEventListener('input', updateSummary);
```

```
upiRadio.addEventListener('change', function() {
  if (upiRadio.checked) upiSection.classList.add('show'); else upiSection.classList.remove('show');
  updateSummary();
});
codRadio.addEventListener('change', function() {
  if (upiRadio.checked) upiSection.classList.add('show'); else upiSection.classList.remove('show');
  updateSummary();
});

// regenerate QR when relevant fields change (ensures instant update)
function regenerateQrIfNeeded() {
  const costs = calculateCosts();
  if (upiRadio.checked && costs.total > 0) {
    const ustr = buildUpiString(costs.total);
    renderQRCode(ustr);
  } else {
    qrHolder.innerHTML = '';
  }
}
eventTypeSelect.addEventListener('change', regenerateQrIfNeeded);
venueSelect.addEventListener('change', regenerateQrIfNeeded);
durationSelect.addEventListener('change', regenerateQrIfNeeded);
attendeesInput.addEventListener('input', regenerateQrIfNeeded);

// Pay Now button (opens UPI link)
document.getElementById('payNowBtn').addEventListener('click', function() {
  const costs = calculateCosts();
  if (costs.total <= 0) {
    showInlineMessage('Please enter attendees and select options to calculate amount.');
    return;
  }
  const ustr = buildUpiString(costs.total);
  upiClicked = true;
  sessionStorage.setItem('upiClickedAt', Date.now().toString());
  sessionStorage.setItem('upiAmount', costs.total.toString());
  window.open(ustr, '_blank');
  showInlineMessage('Opening UPI app to pay ₹' + costs.total.toLocaleString('en-IN'));
});

// Form submit
form.addEventListener('submit', function(e) {
  e.preventDefault();
  if (!validateForm()) {
    return;
  }
  submitBtn.disabled = true;
  const submitBtnText = document.getElementById('submit_button_text');
  if (submitBtnText) submitBtnText.textContent = 'Processing...';
```

```
// gather data
      const costs = calculateCosts();
      const costPerPerson = costs.costPerPerson;
      const attendees = costs.attendees;
      const eventCost = costs.eventCost;
      const venueCost = costs.venueCost;
      const total = costs.total;

      const paymentModeRaw = document.querySelector('input[name="paymentMode"]:checked').value;
      let paymentModeFinal = paymentModeRaw;
      if (paymentModeRaw === 'UPI') {
        paymentModeFinal = upiClicked ? 'UPI (payment initiated)' : 'UPI (pending)';
      }

      const receiptNumber = 'EVT' + Date.now();
      const registrationDate = new Date().toLocaleString('en-IN', { hour12: false });

      const registrationData = {
        receipt_number: receiptNumber,
        registration_date: registrationDate,
        full_name: fullNameInput.value.trim(),
        email: emailInput.value.trim(),
        mobile: mobileInput.value.trim(),
        event_type: eventTypeSelect.value,
        event_date: eventDateInput.value ? new Date(eventDateInput.value).toLocaleDateString('en-IN') : '',
        venue: venueSelect.value,
        duration: durationSelect.value,
        attendees: attendees,
        cost_per_person: costPerPerson,
        event_cost: eventCost,
        venue_cost: venueCost,
        payment_mode: paymentModeFinal,
        total_amount: total,
        timestamp: Date.now()
      };

      // Save to Data SDK if available
      (async function persistAndNotify() {
        try {
          if (window.dataSdk && window.dataSdk.create) {
            // show saving text
            if (submitBtnText) submitBtnText.textContent = 'Saving registration...';
            const result = await window.dataSdk.create(registrationData);
            if (result && result.record && result.record.__backendId) {
              registrationData.__backendId = result.record.__backendId;
            }
          } else {
            // fallback: store locally
          }
```

```
// Add to in-memory list and update dashboard
        allRegistrations.push(registrationData);
        updateDashboard();

        // Organizer auto-download (keep behavior)
        setTimeout(() => {
          generateAndDownloadReceipt(registrationData);
          showInlineMessage('Receipt downloaded for organizer.');
        }, 400);

        // Try sending emails (may fail if EmailJS not configured)
        try {
          await sendEmails(registrationData);
          showInlineMessage('Confirmation emails sent to customer and organizer.');
        } catch (err) {
          console.warn('Email send failed or not configured:', err);
          showInlineMessage('Note: Email not sent — configure EmailJS to enable emails.');
        }

        // Populate hidden receipt UI for user
        document.getElementById('receipt_number').textContent = registrationData.receipt_number;
        document.getElementById('receipt_date').textContent = registrationData.registration_date;
        document.getElementById('receipt_fullname').textContent = registrationData.full_name;
        document.getElementById('receipt_email').textContent = registrationData.email;
        document.getElementById('receipt_mobile').textContent = registrationData.mobile;
        document.getElementById('receipt_event').textContent = registrationData.event_type;
        document.getElementById('receipt_eventdate').textContent = registrationData.event_date;
        document.getElementById('receipt_venue').textContent = registrationData.venue;
        document.getElementById('receipt_duration').textContent = registrationData.duration;
        document.getElementById('receipt_attendees').textContent = registrationData.attendees;
        document.getElementById('receipt_costperperson').textContent = '₹' + registrationData.cost_per_person;
        document.getElementById('receipt_eventcost').textContent = '₹' +
registrationData.event_cost.toLocaleString('en-IN');
        document.getElementById('receipt_venuecost').textContent = '₹' +
registrationData.venue_cost.toLocaleString('en-IN');
        document.getElementById('receipt_payment').textContent = registrationData.payment_mode;
        document.getElementById('receipt_total').textContent = '₹' +
registrationData.total_amount.toLocaleString('en-IN');

        const upi_row = document.getElementById('upi_row');
        if (registrationData.payment_mode && registrationData.payment_mode.toLowerCase().includes('upi'))
{
          if (upi_row) upi_row.style.display = 'flex';
        } else {
          if (upi_row) upi_row.style.display = 'none';
        }
        // show modal to user (they can download)
        successModal.classList.add('show');
```

```
// Reset states
  submitBtn.disabled = false;
          if (submitBtnText) submitBtnText.textContent = 'Complete Registration & Payment';
          upiClicked = false;
          sessionStorage.removeItem('upiClickedAt');
          sessionStorage.removeItem('upiAmount');
        } catch (err) {
          console.error('Save/notify error', err);
          showInlineMessage('Error saving registration. Try again.');
          submitBtn.disabled = false;
          if (submitBtnText) submitBtnText.textContent = 'Complete Registration & Payment';
        }
      })();
    });

    // Download receipt (user)
    downloadReceiptBtn.addEventListener('click', function() {
      const rn = document.getElementById('receipt_number').textContent;
      const registration = allRegistrations.find(r => r.receipt_number === rn);
      if (registration) {
        generateAndDownloadReceipt(registration);
        showInlineMessage('Receipt downloaded.');
      } else {
        showInlineMessage('No receipt available.');
      }
    });

    // modal close
    closeModalBtn.addEventListener('click', function() {
      successModal.classList.remove('show');
      form.reset();
      updateSummary();
      upiSection.classList.remove('show');
    });
    successModal.addEventListener('click', function(e) {
      if (e.target === successModal) {
        successModal.classList.remove('show');
        form.reset();
        updateSummary();
        upiSection.classList.remove('show');
      }
    });

    // initialize
    updateSummary();
    updateDashboard();
```

```
// Expose for debugging (optional)
      window._ERS = {
        rebuildQr: regenerateQrIfNeeded,
        getAll: () => allRegistrations
      };
    })();
  </script>
 </body>
</html>
```