Event Management Software

1. Introduction

1.1 Purpose

The purpose of this document is to outline the Software Requirements Specification (SRS) for the Event Management Software. The software will help manage events, track attendees, handle registrations, payments, and communication with participants. It will also provide analytics and reporting features.

1.2 Scope

The Event Management Software will serve as a platform for creating, managing, and hosting events. It will allow administrators to:

- Create event details (e.g., time, date, location, agenda)
- Manage attendee registration and ticketing
- Handle payments and invoicing
- Provide communication channels (e.g., email, SMS)
- Track event statistics and provide analytics

2. Overall Description

2.1 Product Perspective

The Event Management Software will integrate with existing payment gateways (e.g., PayPal, Stripe), communication services, and analytic tools (e.g., Google Analytics). It will be a web-based platform, accessible via browsers on desktops and mobile devices.

2.2 Product Features

The software will provide the following key features:

- Event creation and management
- Select venue and book venue for offline event
- Team Management (in app there is a feature to create team and assign task)
- Track status of task from assigning to completion.
- Authentication and Authorisation
- Personalised communication method between team mates
- Attendee registration and ticketing
- Payment processing
- Email and SMS notifications
- Reporting and analytics dashboard

- User management (Admin, Attendees)
- Search and filtering for events

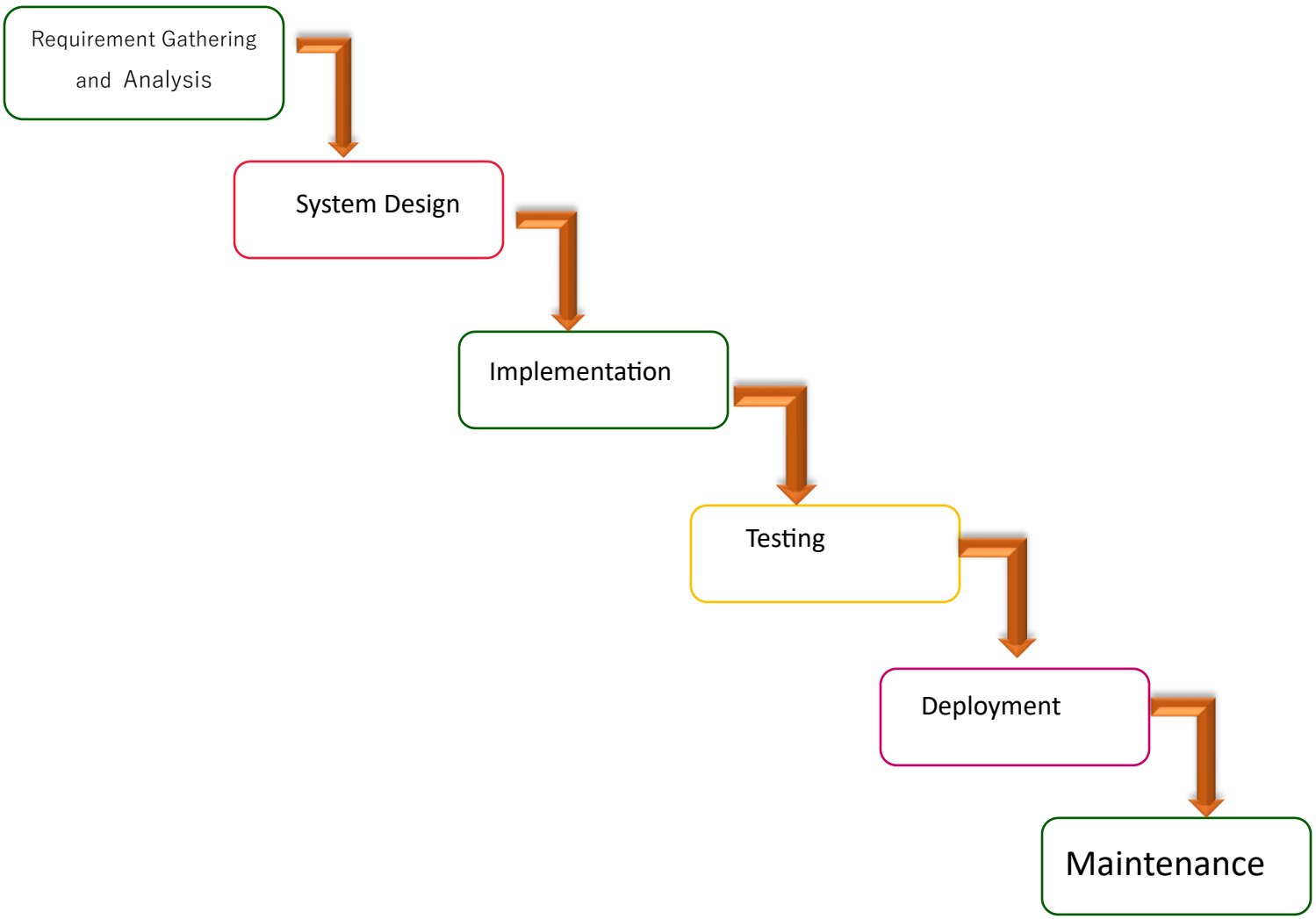
2.3 User Classes and Characteristics

- **Admin:** Manages events, users, and configurations. Requires advanced permissions.
- **Attendee:** Registers and manages their event participation. Requires basic permissions.
- **Event Organizer:** Manages specific events, including agenda, speakers, and attendees.
- **Team leader :** mange task
- **Team mates** : to perform specific task

2.5 Assumptions and Dependencies

- The software will rely on third-party services for payments and communication.
- Users will have access to the internet to interact with the platform.
- Event data is secure and stored following best practices for data privacy.

Representation of the different phases of the software using waterfall model



Requirement Gathering

- Event creation and management
- Select venue and book venue for offline event
- Team Management (in app there is a feature to create team and assign task)
- Track status of task from assigning to completion.
- Authentication and Authorisation
- Personalised communication method between team mates
- Attendee registration and ticketing
- Payment processing
- Email and SMS notifications
- Reporting and analytics dashboard
- User management (Admin, Attendees)
- Search and filtering for events

Requirement Analysis

For an event management software, this analysis helps in understanding the system's functionality, the user's needs, and how the software will solve specific problems. It ensures that the software aligns with business goals and provides value to users.

1. Functional Requirements

These are the primary features and actions that the event management system must support.

1.1. User Account Management

- **User Registration and Authentication**:

    - The system must allow users to create accounts by providing basic details like name, email, password, and role (organizer, attendee, or admin).
    - The system must support secure login through email/password or third-party authentication methods (e.g., Google, Facebook).
    - The system must allow users to recover/reset their password if forgotten.
- **User Roles and Permissions**:
    - **Event Organizer**: Can create, edit, and manage events. Can view event statistics and attendee data.
    - **Attendee**: Can register for events, view event details, and receive notifications about events.
    - **Admin**: Has full access to all events, users, and can manage system-wide settings.

1.2. Event Creation and Management

- **Event Creation**:
    - Event organizers should be able to create events by entering the event name, description, date, time, venue, and other event-specific information.
    - The system should allow organizers to upload images, logos, or documents related to the event.
    - Organizers must be able to set event types (e.g., conference, workshop, seminar).
- **Event Updates and Management**:
    - Event organizers should be able to edit or update event details (venue, date, time) at any time.
    - The system should allow organizers to cancel or reschedule an event, with attendees being notified accordingly.
- **Event Listing**:
    - The system should display a list of all events, allowing attendees to browse, filter (by date, location, type), and search for events of interest.

1.3. Attendee Registration

- **Event Registration**:
    - Attendees should be able to register for an event via a simple sign-up form (name, email, and optional details).
    - The system should allow the user to select different types of tickets (if applicable), such as general admission, VIP, or discounted tickets.
- **Ticketing**:
    - The system must allow event organizers to set different ticket prices, quantity limits, and provide discounts or promotional codes.
    - Attendees should receive confirmation and an electronic ticket after successful registration.
- **Waitlist**:
    - If the event is fully booked, attendees should be able to join a waitlist and receive a notification if a spot becomes available.

1.4. Payment Integration

- The system must allow payment processing for event tickets. It should integrate with payment gateways like PayPal, Stripe, or credit card processors.
- The system should provide secure payment handling and ensure that users' financial data is protected.

1.5. Notifications

- **Event Notifications**:
  - Attendees should receive confirmation emails when they register for an event.
  - The system should send reminders to attendees via email and SMS (if applicable) before the event starts.
  - If there are any changes or cancellations to the event, attendees should be notified immediately.
- **Admin Notifications**:
  - Admins and organizers should receive alerts for registrations, payment completions, and any changes made to events.

1.6. Event Analytics and Reporting

- Event organizers and admins should have access to reports and statistics regarding the event, such as the number of registrants, ticket sales, revenue, and attendee demographics.
- The system should allow organizers to export event data in formats like CSV or Excel for further analysis.

1.7. Social Media Integration

- Attendees should be able to share event details on social media platforms (Facebook, Twitter, LinkedIn) directly from the event page.
- Event organizers should be able to promote events via social media using integrations with platforms like Facebook Ads or Google Ads.

---

2. Non-Functional Requirements

These are the system’s operational attributes, such as performance, security, scalability, and usability.

2.1. Performance Requirements

- The system should handle up to 10,000 concurrent users without performance degradation.
- The system must be responsive, with page load times under 3 seconds.
- Event registration and ticket payment should be completed in less than 5 seconds.

2.2. Security Requirements

- The system must use HTTPS for secure communication.
- User passwords must be stored in an encrypted format (e.g., bcrypt).
- Payment information should be processed in compliance with PCI DSS standards to ensure secure transactions.
- The system should have proper role-based access control, ensuring that users can only access data relevant to their role.

2.3. Usability Requirements

- The user interface (UI) must be intuitive, clean, and easy to navigate.
- The system should be mobile-responsive, allowing users to access and manage events from mobile devices.
- The system should support multiple languages (if targeting a global audience).

2.4. Scalability

- The system must be scalable to accommodate a growing user base and a high number of events.
- The system should be able to handle sudden traffic spikes, such as during ticket sales for a popular event.

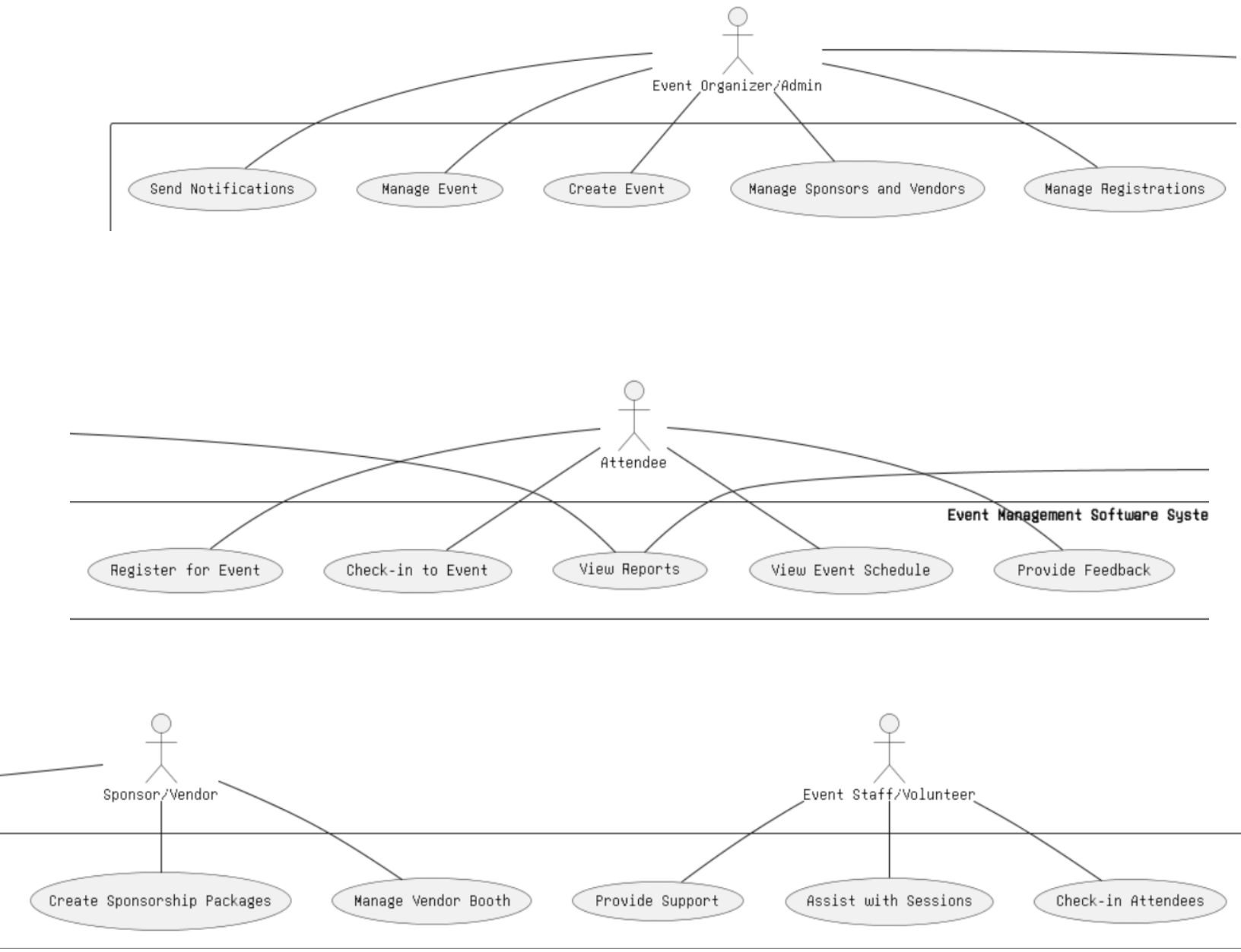
2.5. Availability and Reliability

- The system must have 99.9% uptime, ensuring it is available to users without significant downtime.
- The system should be able to recover gracefully from failures, with data integrity maintained.

2.6. Compliance and Legal Requirements

- The software must comply with data protection regulations such as the General Data Protection Regulation (GDPR) for handling personal data of users.
- The system must adhere to legal requirements regarding ticket sales and refunds based on regional laws.

Use case diagram



Use Cases:

For Event Organizer/Admin:

- **Create Event**: Set up events with all necessary details (date, location, description, schedule).
- **Manage Event**: Edit or update event details, sessions, or attendees.
- **View Reports**: Generate reports on ticket sales, revenue, and attendee statistics.
- **Manage Registrations**: View attendee registrations, approve/deny registrations, or send updates.

- **Manage Sponsors and Vendors**: Create sponsorship packages, manage vendor and sponsor details.
- **Send Notifications**: Send out emails or push notifications for reminders, updates, or emergency alerts.

For Attendee:

- **Register for Event**: Sign up for events and select tickets.
- **View Event Schedule**: View session details, speakers, and times.
- **Check-in to Event**: Use QR codes or check-in manually for event participation.
- **Participate in Sessions**: Attend virtual or in-person sessions.
- **View Exhibitors/Vendors**: Browse sponsor/vendor booths during the event.
- **Provide Feedback**: Fill out surveys after the event.

For Speaker/Presenter:

- **Create/Upload Presentations**: Upload presentation materials or content for their sessions.
- **View Session Details**: Check the schedule and session logistics.
- **Interact with Attendees**: Engage with attendees through Q&A, chat, or other tools during the event.

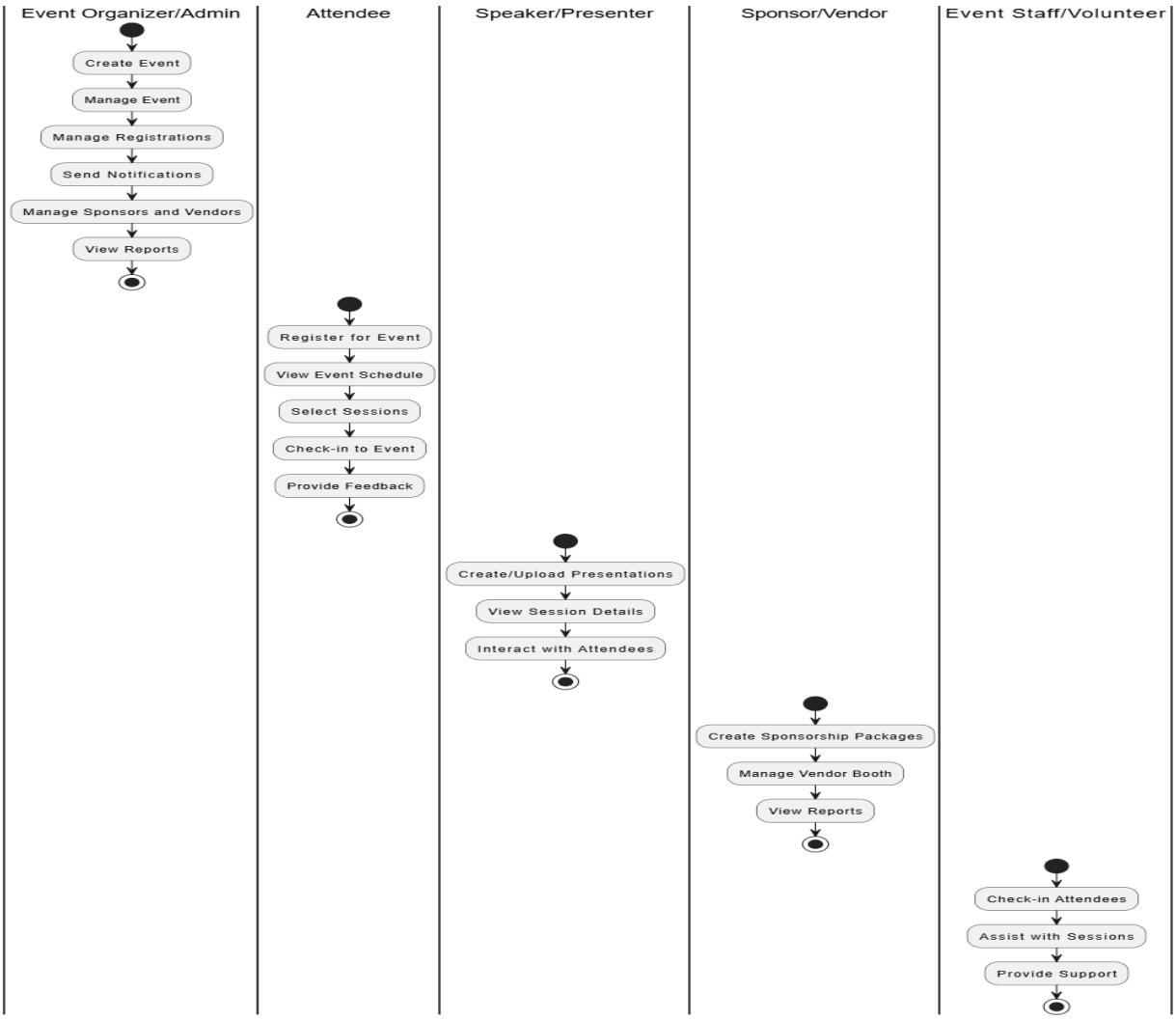
For Sponsor/Vendor:

- **Create Sponsorship Packages**: Choose and manage sponsorship options (e.g., booth placement, logo visibility).
- **Manage Vendor Booth**: Set up and manage virtual or physical booths, and communicate with attendees.
- **View Reports**: Access analytics on booth traffic, leads, and interactions.

For Event Staff/Volunteer:

- **Check-in Attendees**: Verify attendees during the check-in process.
- **Assist with Sessions**: Help organize and manage logistics for sessions or booths.
- **Provide Support**: Offer on-site or remote support for event issues or inquiries.

Flow Oriented


Event Organizer/Admin
Create Event
Manage Event
Manage Registrations
Send Notifications
Manage Sponsors and Vendors
View Reports
Attendee
Register for Event
View Event Schedule
Select Sessions
Check-in to Event
Provide Feedback
Speaker/Presenter
Create/Upload Presentations
View Session Details
Interact with Attendees
Sponsor/Vendor
Create Sponsorship Packages
Manage Vendor Booth
View Reports
Event Staff/Volunteer
Check-in Attendees
Assist with Sessions
Provide Support

Project Management

a. Computing Functional Point

Function Type	Count	Complexity	Functional Point = SUM {(count * complexity)}
External Input (EI) (event registration, payment information, user feedback, vendor info)	4	3	4x3=12
External Output (EO) (registration confirmation, event report, invoice generation)	3	4	3x4=12
External Inquiries (EI) (event info query, availability check, attendee list query)	3	3	3x3=9
Internal Logical File (ILF) (attendee database, event data, payment record, vendor data)	4	7	4x7=28
External Internal File (EIF) (payment gateway, social media feedback, ticketing platform interface)	4	5	4x5=20
			12+12+9+28+20= 81

Functional point= 85

b. Now we will be calculated estimated effort:

EF= functional point x productivity (effort factor)

We take productivity 2.5 as universally so here,

= 81 x 2.5

=202.5

Therefore, estimated effort is 202.5

c. Schedule, Risk Table, Timeline chart

Schedule Table :-

task	Start date	End date	duration	Responsible team(s)
Project planning & research	April 1	April 30	1 Month	Project Manager, Business Analysts
Requirement gathering & analysis	April 15	May 15	1 Month	Business Analysts, Dev Team, Clients
UI/UX Design & Prototyping	May 1	June 15	1.5 Months	Design Team
Backend Development	June 1	September 30	4 Months	Dev Team
Front Development	June 15	October 15	4 Months	Dev Team, Design Team
Payment Gateway Integration	August 1	September 15	1.5 Months	Dev Team
Internal Testing &Bug Fixing	September 1	October 15	1.5 Months	QA Team
Beta Release Unit Testing	October 15	November 15	1 Month	QA Team, Selected Users
Final Development & Launch	November 15	November 30	2 Weeks	Dev Team, Marketing Team
Post Maintenance & Updates	Ongoing	Ongoing	Continuous	Dev Team, Support Team

💡 **Key Differences from the scheduled Table:**

✅ **Date-Specific:** Includes actual start and end dates for each task.
✅ **Task-Oriented:** Focuses on what needs to be done rather than just phases.
✅ **Responsible Teams Listed:** Clearly assigns tasks to specific teams.
✅ **Different Durations:** Not all tasks are the same length—some overlap **while** others are sequential.

Risk Table :-

Risks	category	probability	impact	RMMM
Inaccurate estimation of event capacity	PS	50%	3	Optimize estimation algorithms, use real event data
Unexpected surge in users during events	PS	40%	3	Implement scalable cloud infrastructure
Low adoption by event organizers	BU	50%	2	Conduct training sessions, offer incentives
User resistance to new ticketing system	BU	30%	2	Provide tutorials and support for onboarding
Tight deadlines for event feature releases	BU	60%	3	Use agile methodologies and prioritize features
Last-minute event cancellations	CU	50%	2	Implement refund policies and automated notifications
Frequent change requests from clients	CU	70%	3	Use flexible modular development approach
Security breaches in payment processing	TE	40%	3	Implement encryption and compliance measures (PCI-DSS)
System crashes due to high traffic	TE	50%	3	Load testing, auto-scaling, and failover strategies
Lack of training on new event management tools	DE	30%	2	Conduct training workshops for event organizers
Inexperienced developers working on the system	ST	40%	2	Provide mentorship and documentation
High staff turnover leading to project delays	ST	50%	3	Offer competitive salaries and career growth

Category

PS - (Project Scope)

BU - (Business)

CU - (Customer)

TE - (Technology)

DE - (Development Environment)

ST - (Staffing)

impact value

1= catastrophic

2= critical

3= marginal

4= negligible

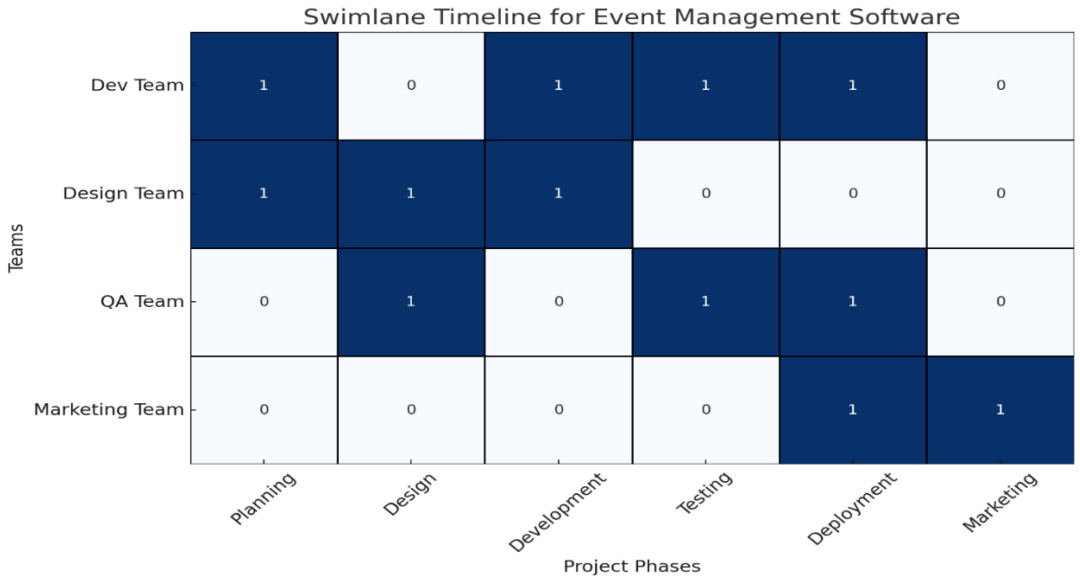
RMMM= Risk Mitigation, Monitoring, and Management

Timeline chart:-

Phase	Phase 👷	Design Team 🎨	QA Team 🔍	Marketing 📢
Planning (Month 1)	🟢 Research	🟢 Wireframes	-	-
Development (Months 3-6)	🟢 Backend & Frontend	🟢 UI Enhancements	-	-
Testing (Month 7)	-	-	🟢 Testing	-
Deployment (Month 8)	🟢 Final Fixes	-	🟢 Security Tests	🟢 Promotions

How to Read the Chart:

1. **Rows → Teams**
    - **Dev Team (Developers 👷)**: Handles coding, integrations, and system architecture.
    - **Design Team (UI/UX Designers 🎨)**: Works on user interface and experience.
    - **QA Team (Testers 🔍)**: Ensures software quality through testing.
    - **Marketing Team (Marketers 📢)**: Manages promotion and branding.
2. **Columns → Project Phases**
    - **Planning** (Strategy, Research, Scope Definition)
    - **Design** (UI/UX Prototyping, Wireframes)
    - **Development** (Backend, Frontend, Feature Implementation)
    - **Testing** (Bug Fixing, Security, Performance Checks)
    - **Deployment** (Final Release, Go-Live Preparation)
    - **Marketing** (Promotional Campaigns, User Engagement)



Explanation of the Chart:

- Each row represents a **team** (Development, Design, QA, Marketing).
- Each column represents a **project phase** (Planning, Design, Development, Testing, Deployment, Marketing).
- **Blue cells** indicate when a team is actively working on a phase.
- Some teams work in multiple phases at the same time (e.g., Development is active during Design, Development, Testing, and Deployment).

This chart helps in **visualizing team responsibilities and workload** throughout the project timeline.

Difference Between a Schedule Table and a Timeline Table

Aspect	Schedule Table	Timeline Table
Purpose	Specifies exact durations and task assignments.	Shows a sequential order of phases visually.
Format	A **tabular format** listing tasks, time, and team assignments.	A **linear, vertical, or graphical representation** of phases over time.
Focus	Focuses on **who does what and for how long.**	Focuses on **when phases occur and their dependencies.**
Granularity	More detailed—includes specific teams and their involvement in each phase.	More high-level—shows how the project progresses over time.
Best For	**Day-to-day planning**, resource allocation, and tracking team contributions.	**Project overview**, presentations, and understanding the big picture.
Example Representation	Table with **Phases, Duration, Teams, and Responsibilities.**	Gantt chart, Swimlane Diagram, or Linear Timeline.