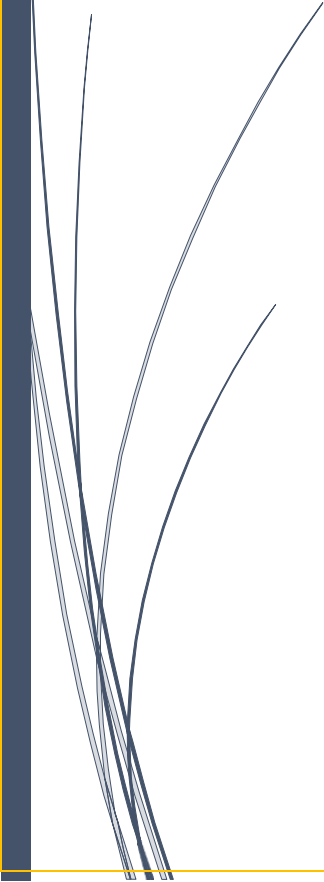# EVENT MANAGEMENT SYSTEM

Group 09

Yelena Siliankina

Maria Bhimisetty

Ria Patole

Hrishikesh Hete

Jinha Noh

Table of Contents

Event Management System

Meeting Minutes

Meeting of Minutes (MoM)						
Date	Day	Time	Length of meeting	Type of Meeting	Attendees	Topics Discussed
February 14, 2025	Friday	6:00 PM	1 hour	Teams	ALL	Introduction meeting, Project charter layout, and Project charter draft
February 16, 2025	Sunday	4:30 PM	1 hour	Teams	ALL	Review Project charter and revise the charter
February 26, 2025	Wednesday	6:30 PM	2 hours	Teams	ALL	Project charter and BPMN 1st Draft
March 05, 2025	Wednesday	7:00 PM	2 hours 20 min	Teams	ALL	Continuing Static Model (BPMN, Context Diagram, Use Case Diagram, and Use Case Description)
March 13, 2025	Thursday	4:00 PM	2hours 8 min	Teams	ALL	Prepare for the project submission of models
March 22, 2025	Saturday	5:00 PM	1 hour 26 min	Teams	ALL	Class Diagram
March 30, 2025	Sunday	6:00 PM	20 min	Teams	ALL	Class Presentation
April 2, 2025	Wednesday	6:30 PM	1 hour 35 min	Teams	ALL	ERD and DFD
April 9, 2025	Wednesday	5:30 PM	1 hour	Teams	ALL	State Diagram
April 10, 2025	Thursday	6:00 PM	1 hour 30 min	Teams	Yelena, Hrishikesh, Maria, and Ria	Sequence Diagram
April 16, 2025	Wednesday	5:30 PM	1 hour 30 min	Teams	ALL	Write Up and User Interface (UI)
April 23, 2025	Wednesday	7:30 PM	1 hour 15 min	Teams	ALL	Complete presentation slides, correct diagrams, and finish UI
April 28, 2025	Monday	3:20 PM	30 min	In Person	ALL	Practice presentation of diagrams
April 30, 2025	Wednesday	10:30 AM	2 hours	In Person	ALL	Finish video
May 7, 2025	Wednesday	10:00 AM	2 huors	Teams	ALL	Finish Project Report (Gantt Chart, Work Breakdown Structure, Resource Allocation, Financials)

Link: Meeting Note

Introduction

Problem statement and solutions:

The Event Management System (EMS) is designed to streamline event planning, coordination, and execution of client events. It will serve as a platform for internal event managers to collect event requirements, coordinate with service providers (vendors), generate invoices, confirm bookings, and follow up with post-event feedback. Overall, the system enables users to create, manage, and track events efficiently. The system will include features such as event scheduling, notifications, payment and feedback processing. The system aims to reduce menial coordination, increase operational efficiency, and provide seamless experience for clients, event managers, and vendors.

Project Writeup

Project Scope and Vision

The objectives of the event management system project include:

- Developing a user-friendly web-based application for event management.
- Implementing core functionalities including event creation, scheduling, and registration.
- Ensuring data security and system scalability.

Scope:

Creating an event management system with notifications, reporting, and database integration.
The deliverables for the project include a BPMN Diagram, Subsystem Diagrams, Context Diagrams, Use Case Diagrams, Use Case Description, Activity Diagram, Class Diagram, Entity Relationship Diagram (ERD), Data Flow Diagram (DFD), State chart Diagram, and Sequence Diagram.

In Scope:
- A web-based system for internal event managers and vendor coordination
- Client event request
- Vendor database (venues, catering, decor, photography)
- Client database
- Event planning workflow including vendor assignment and approval
- Invoice generation and client payment processing
- Event confirmation and post-event feedback collection

Tasks to tackle in the project include:

Development of the project charter, including the problem statement, scope, potential future scope, risks, risk mitigation strategies, technical feasibility assessment, feature design, diagram designs, and unique selling point evaluation.

Potential future phase scope:
Implement AI-driven recommendations for reaching budget goals and determining the most cost-friendly alternatives for clients.

Target Audience

The Event Management System (EMS) is designed to serve three primary groups:
- **Internal Event Managers**: Individuals responsible for coordinating all aspects of client events, including communication with vendors, managing logistics, and ensuring successful event execution.
- **Clients**: Individuals or organizations who submit event requests, provide necessary details, and complete payments for services rendered.
- **Vendors:** External service providers such as venue companies, catering services, decoration providers, and photography businesses who fulfill event-related needs based on the requirements shared through the EMS.

The system is built to simplify communication between these parties, reduce administrative overhead, and create a more efficient event planning experience. By providing a centralized platform, EMS helps event managers streamline their workflows, assists clients in tracking the progress of their events, and enables vendors to manage and confirm bookings more efficiently.

Features for End User

- **Event Request**
  - Users can request an event by providing details like event type (wedding, birthdays, graduation party etc.), number of attendees, budget, date and time.
  - For weddings, users can also request additional events like pre-wedding events, rehearsal dinners etc.
- **Venue, Photography, Decoration and Catering Booking**
  - Enables users to view and confirm the bookings placed for venue, photography, decoration and catering.
- **Payment Processing**
  - Enables users to validate and confirm payment requests for the events requested.
  - Users can submit their payment requests through secure payment options like credit/debit cards, PayPal etc.
- **Feedback**
  - Enables users to provide feedback on the services provided after the event is completed.
  - It allows vendors to improve their offerings and enables other users to view the quality of services provided.

Constraints and Risks

Risks:

1. Scope creep: gradual expansion of project goals, features, or deliverables beyond expected and agreed resources, time, and budget.
2. Feature Creep: Addition of low-priority features to a product or project, beyond the original planned scope.
3. Team Availability Issues: Key personnel are unexpectedly unavailable due to conflicting commitments and unexpected events or illnesses which can delay the project delivery, reduce the quality or force scope adjustments.
4. Technical Challenges: Unforeseen issues that arise during the development process due to system complexity and technology limitations.

Strategies to Mitigate Potential Risks:

1. To avoid scope creep, the scope is defined upfront. The deliverables, features, and system requirements will be documented in detailed project scope statement.
2. To avoid feature creep, there is a determined list of core features that are required for the system, which are agreed upon in the project planning prior to the start of the project.

3. To prevent team availability issues from disrupting the project, there is a resource management plan where team member responsibilities are outlined, as well as the dispersion of responsibilities if tasks are in-completed due to team member absences.
4. To prevent technology challenges, a technical feasibility assessment is performed in the early planning phases of the project. In the assessment, high-risk components such as payment systems are identified and available future solutions are discussed.

Constraints:
1. Time Constraints: The 4-month period provided for this project can limit the complexity and depth of features that can be implemented and tested.
2. Technology Stack Limitations: For the project, the use of only open-source or free technologies to build the system may limit flexibility in adding high-end components.
3. Team Size and Skill Limitations: The small team of 5 may result in a more simplified completion of the required tasks for the project.
4. Security and Compliance Constraints: The Event Management System will handle personal user information and payment processing, so basic security practices such as encryption and role-based access must be in place. However, high-level security integrations are limited by time and technical constraints.
5. Scope Constraints: Only internal managers and known vendors are supported in the initial system version. External vendor onboarding is outside of the scope. Mobile app development is also not included in the initial project scope.
6. AI and Automation Constraints: Full AI implementation for budget analysis and suggestions is limited to basic logic and placeholder features for the current system version.

Resource Management Plan

Team members, as listed on the cover page, will participate in each step of the project. The tasks in the project include project charter development, diagram development, team meeting management, UI development, and video creation.

If team member availability issues arise, the responsibility of the absent member will be dispersed amongst other team members. Once the absent team members can participate in the project again, they will assist in other team members' responsibilities to ensure fair dispersion of work.

Technical Feasibility Assessment

1. <u>System Architecture:</u>

   The system will follow a modular, service-oriented architecture, structured around user and workflows. This includes event managers, vendors (catering, venues, decorations, photography), and clients. It will be a web-based application allowing real-time updates.

2. <u>Integration requirements:</u>

   Payments Gateway, Email Notifications, and Vendor Communication.

3. <u>Core Feature Complexity:</u>

   Most features are workflow-based and not overly technical.
   a. Event creation and data collection are simple forms and validation
   b. Vendor selection and tracking requires a potential internal database of vendor profiles, with payment links.
   c. Invoice generation is automated through templates, with payment links.
   d. Client organization requires a potential internal database of client profiles with event linkage to profiles.
   e. Event confirmation logic is rules-based. When a required rule is completed, an event confirmation is sent.
   f. Feedback collection is a simple post-event form or survey link sent to clients once the event is followed through.

4. <u>Security and Compliance:</u>

   The platform will handle user data and payments; basic security measures such as input validations or role-based access control are essential. Payment Card Industry (PCI) compliance for payments is handled by a third party. User data is protected under the general best practices.

5. <u>Scalability:</u>

   Future needs like analytics or vendor rating systems can be modularly added without reworking the core.

6. <u>Team Capabilities:</u>

   The project can be handled with a small team of 5 members who will share responsibilities. Responsibilities include frontend developer, backend developer, UI designer, and a project technical lead.

7. <u>Project Timeline Fit:</u>

   Given the low complexity of most features, this is a 4-month project, from beginning of February through beginning of May. A phased approach of planning the project, developing the diagrams, creating the UI, completing the documentation, and completing the system will ensure rapid delivery of the project and manageable risk.

Implementation Section

Design Aspect

The following are the diagrams that are built as part of this project:

Static Modeling
- BPMN
- Context Diagram
- Use Case Diagram
- Use Case description
- Activity Diagram
- Class Diagram

Dynamic Modeling
- Sequence Diagram
- State Chart Diagram

Data Related (Modeling)
- Data Flow Diagram
- E-R Diagram

Static Modeling

BPMN Diagram

The BPMN (Business Process Model and Notation) diagram models a complete Event Management workflow, starting from a client's event request all the way to the feedback and closure of the event. It includes coordination between the customer, event manager, and the other managers (venue, catering, decoration, photography).
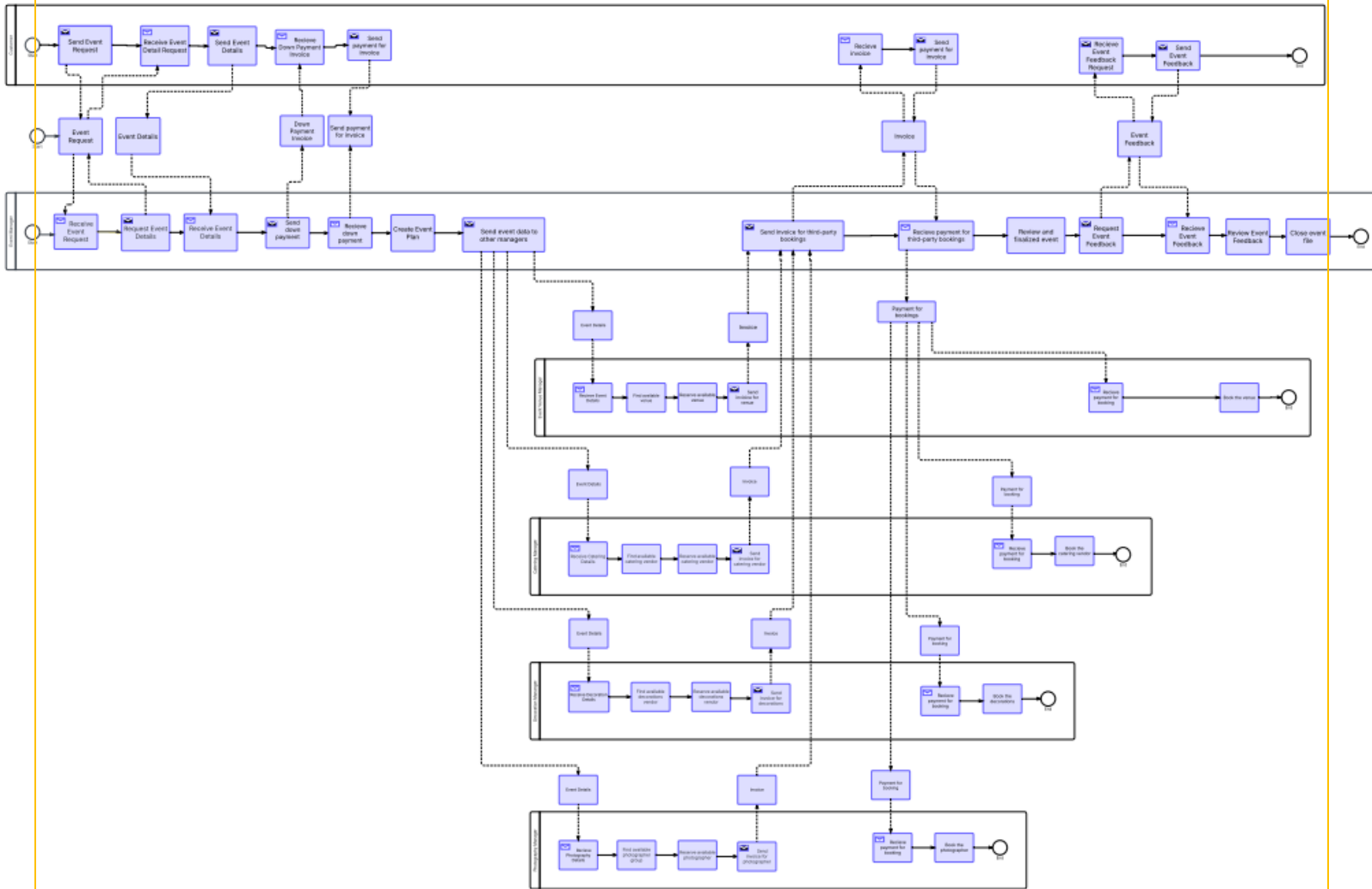
Swim lanes

This BPMN diagram uses 6 swim lanes to represent different participants (roles or systems) involved in the event management process:

- Customer
- Event Manager
- Venue Booking Manager
- Catering Manager
- Decoration Manager
- Photography Manager

Swim lane 1: Customer
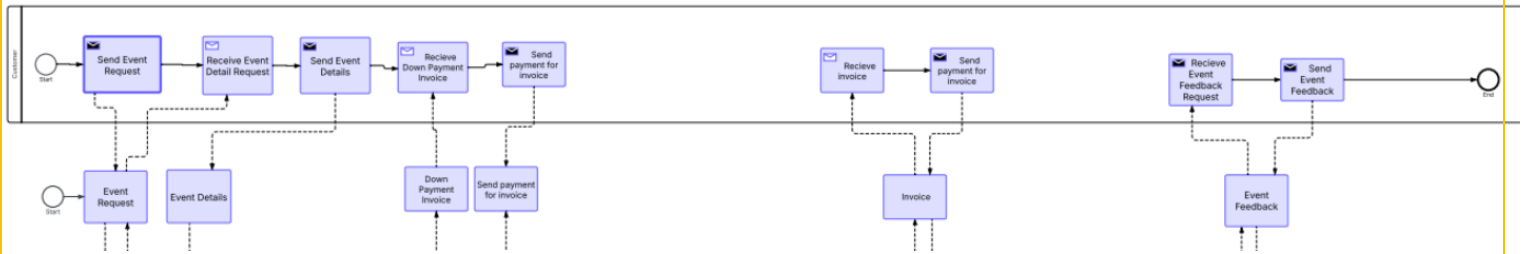
Swim lane 2: Event Manager
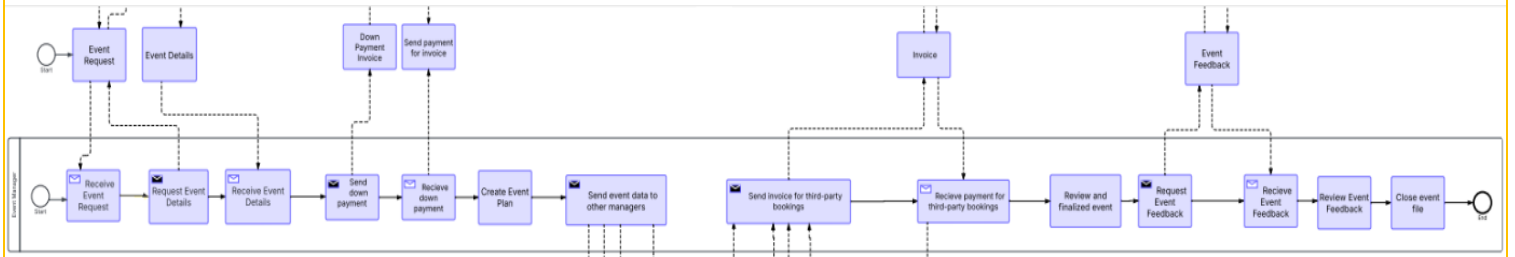
Swim lane 3: Venue Manager
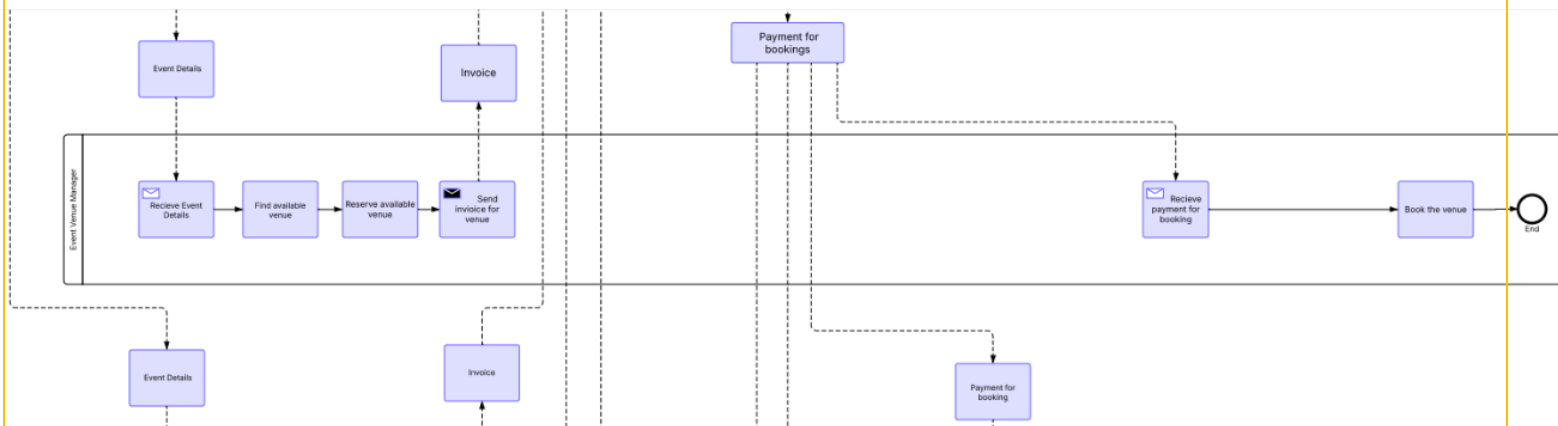
Swim lane 4: Catering Manager
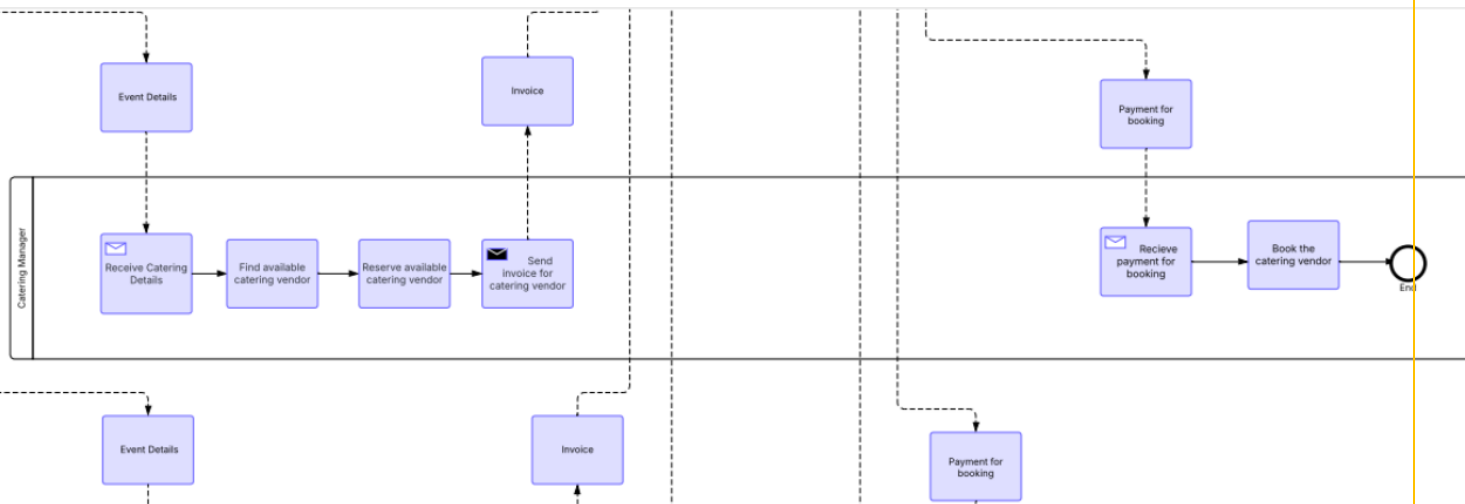
Swim lane 5: Decorations Manager
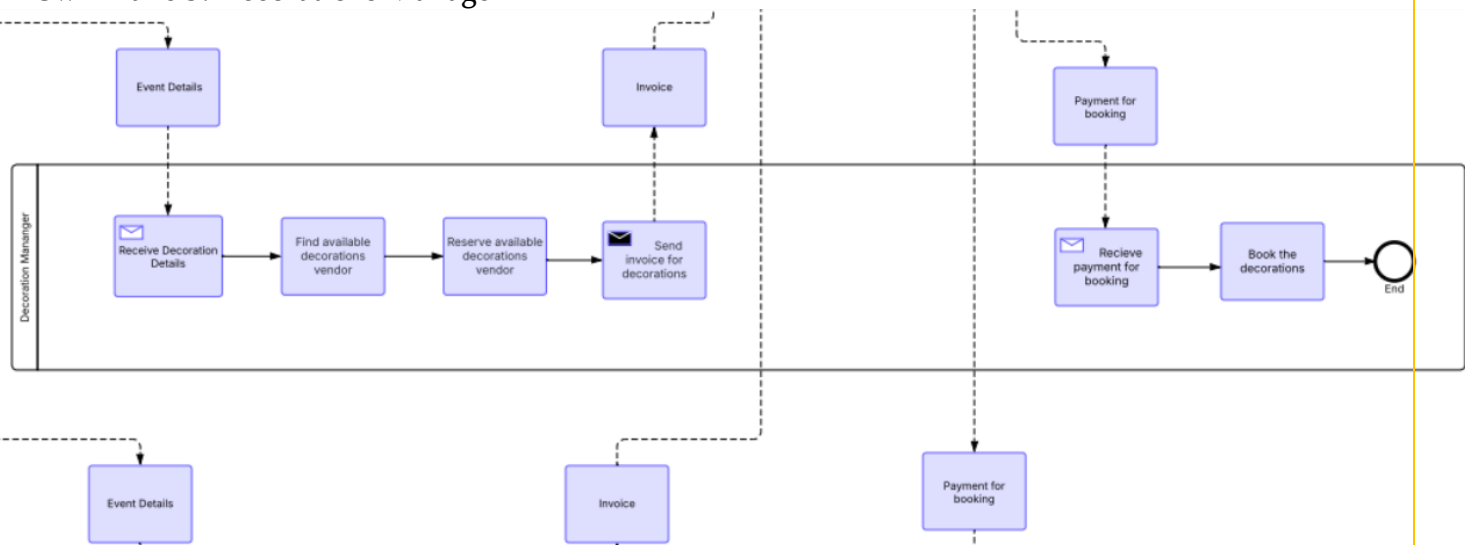
Swim lane 6: Photography Manager
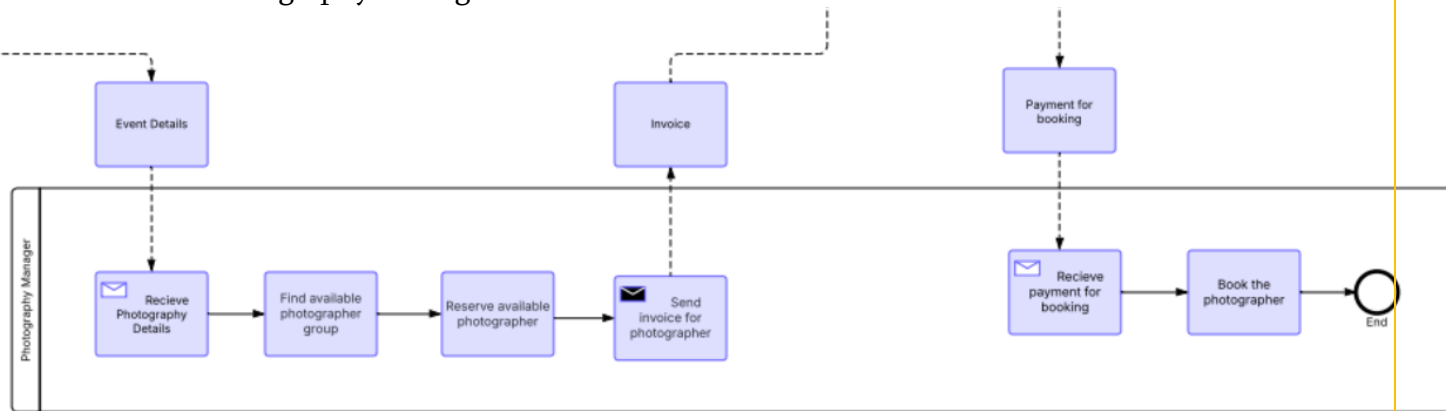
The different subsystems that are a part of the BPMN are as follows:

1. Requirement Gathering Subsystem
- The process starts with the client sending an event request.
- The event manager verifies whether the client is new or returning and creates or updates their profile accordingly.
- Event details are collected through a request-response loop and fed into the system.
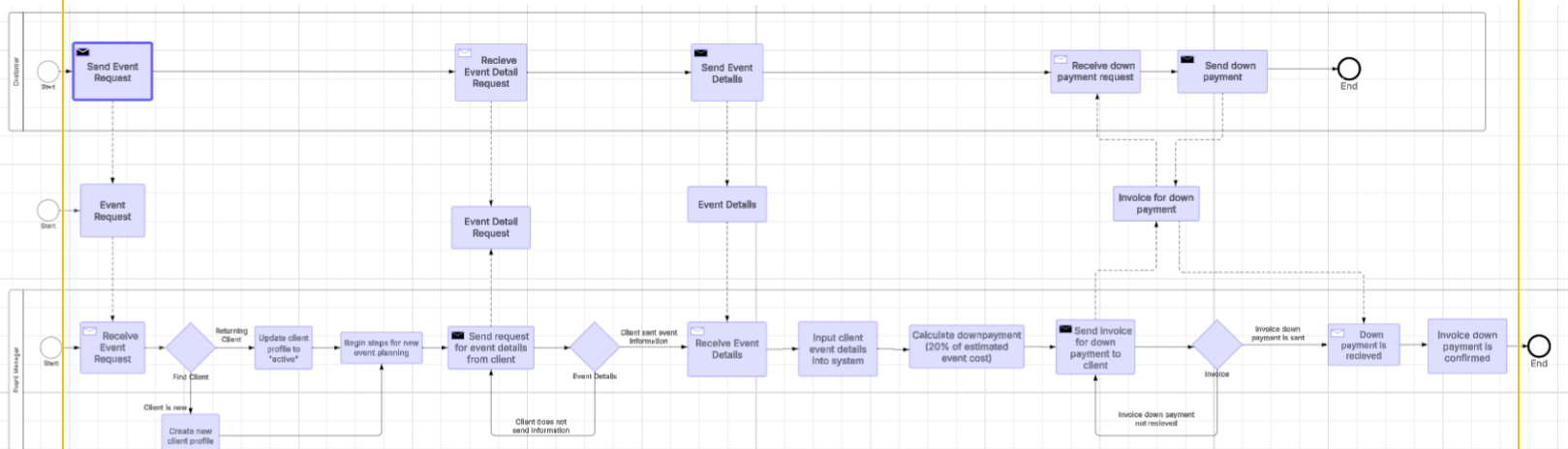
2. Event Planning Subsystem

- Once event details are received, the event manager reviews and builds the event plan.
- The plan is shared with internal teams or managers (e.g., vendor coordinators).
- Feedback from the client is sought up and used to finalize details before bookings.
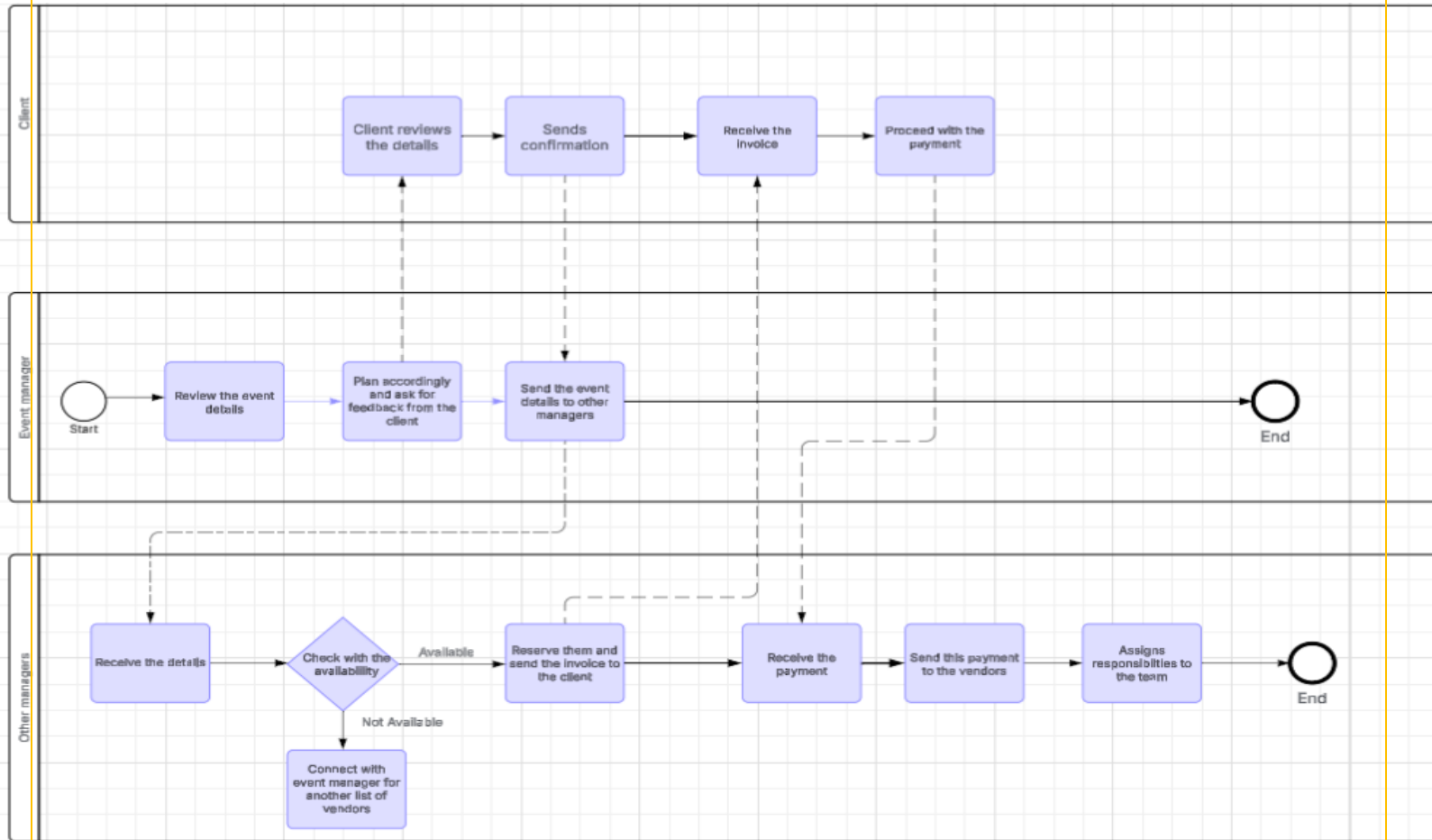
3. Booking Subsystem
- The event manager coordinates with different managers (venue, catering, etc.) to check availability and reserve resources.
- Each vendor receives payments and confirms their bookings individually.
- The event manager tracks all confirmations and proceeds only when all services are booked.
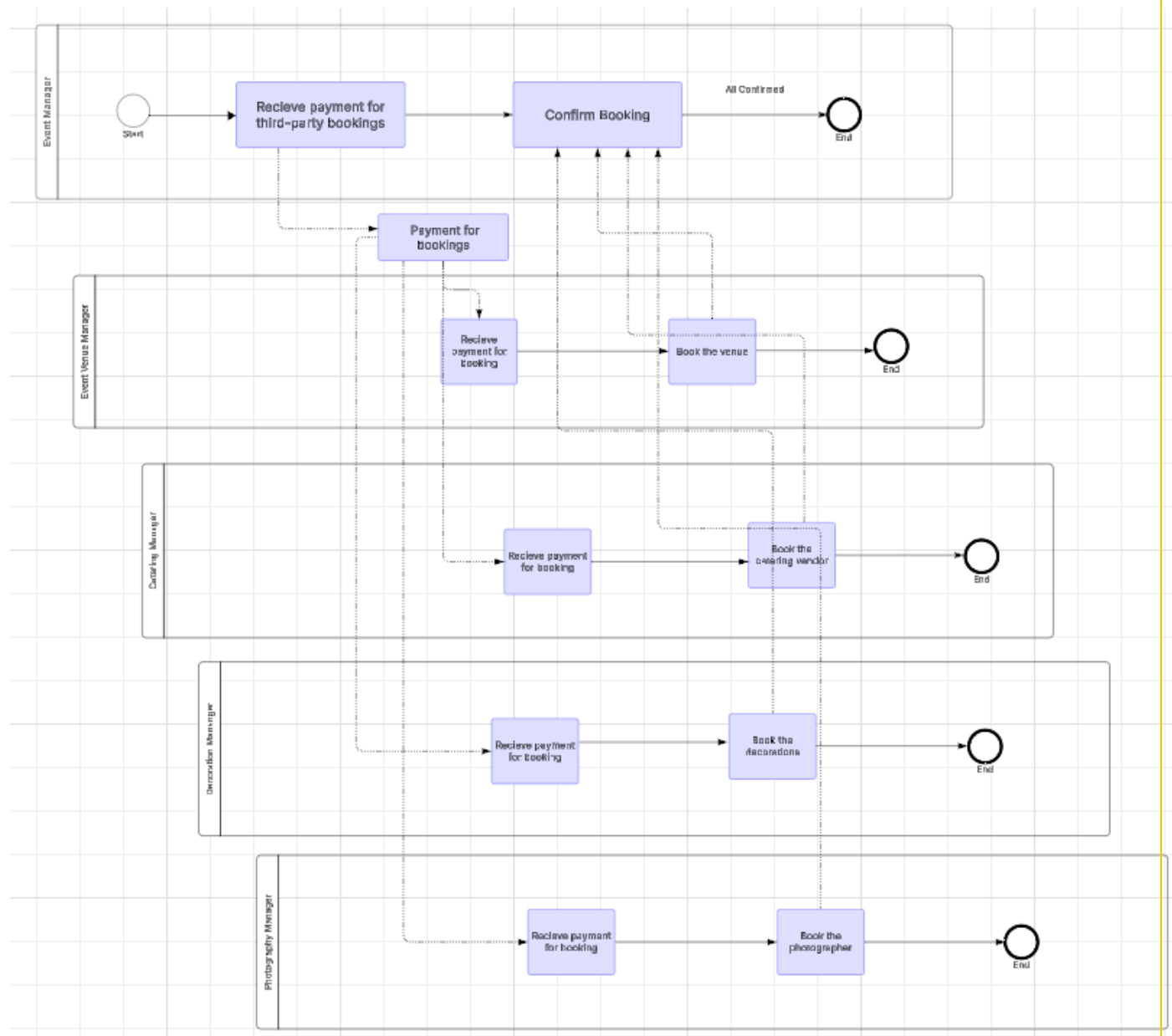
4. Payment & Invoice Subsystem

- After booking, invoices are generated and sent to the client.
- Clients may confirm or seek clarification before making the payment.
- Event manager receives confirmation of payment and updates system records.
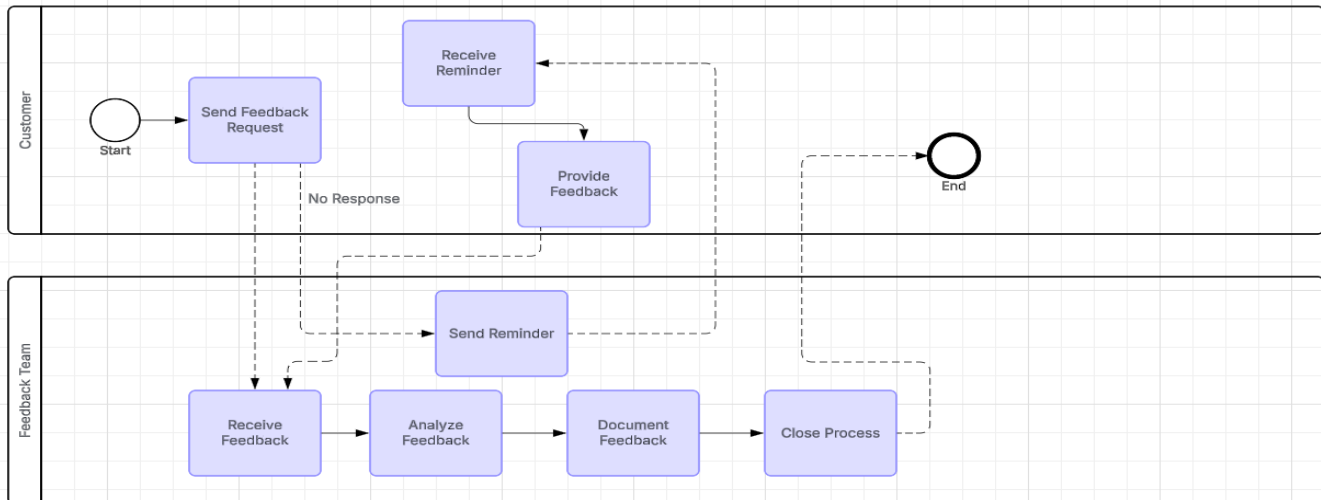
5. Feedback Subsystem

- A feedback request is sent post-event.
- If the client does not respond, automated reminders are triggered.
- Once feedback is received, it is analyzed and documented by the feedback team.
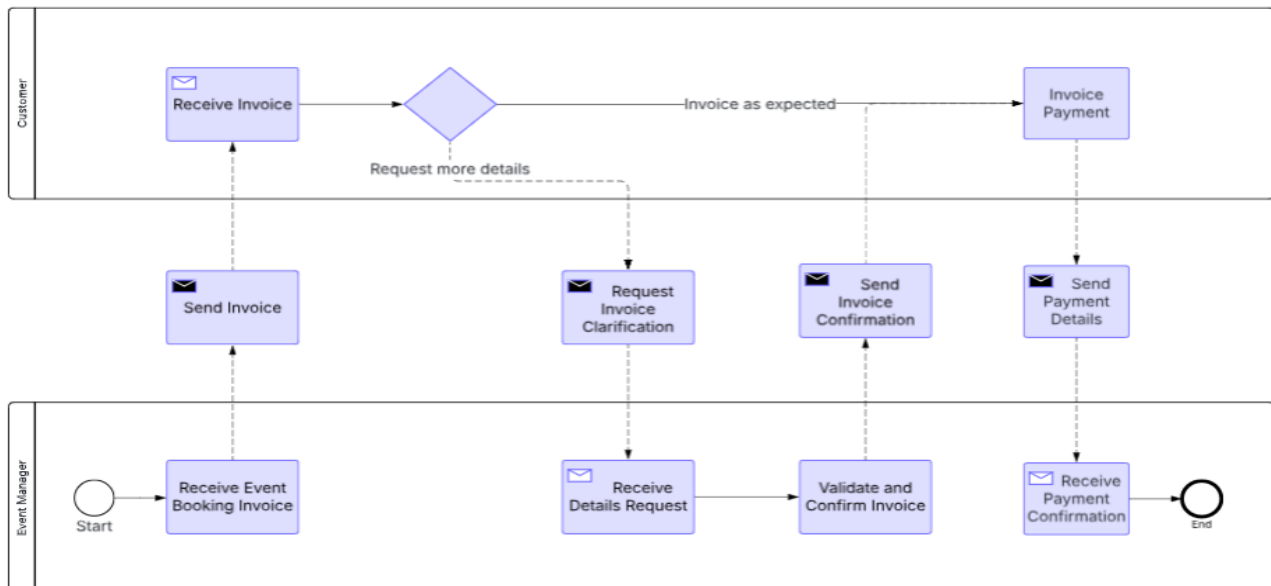
Context Diagram

The context diagram provides a high-level overview of how the Event Management System (EMS) interacts with external entities involved in the event planning process.

<u>In this system:</u>

- Clients initiate interactions by submitting event requests and providing essential event details. They also receive updates, invoices, and confirmations through the system.
- Venue Companies receive booking requests from the EMS and send back confirmations regarding availability and reservation status.
- Catering Companies manage food and beverage arrangements upon receiving requests from the EMS and return confirmation notices.
- Decoration Companies are contacted for venue decor needs, with booking requests and confirmations exchanged through the system.
- Photography Companies handle requests related to event photography services and provide booking confirmations to the EMS.

Through these connections, the EMS acts as an intermediary, ensuring seamless coordination among clients and service providers. The system reduces the need for direct communication between clients and vendors, allowing event managers to oversee the entire process efficiently.
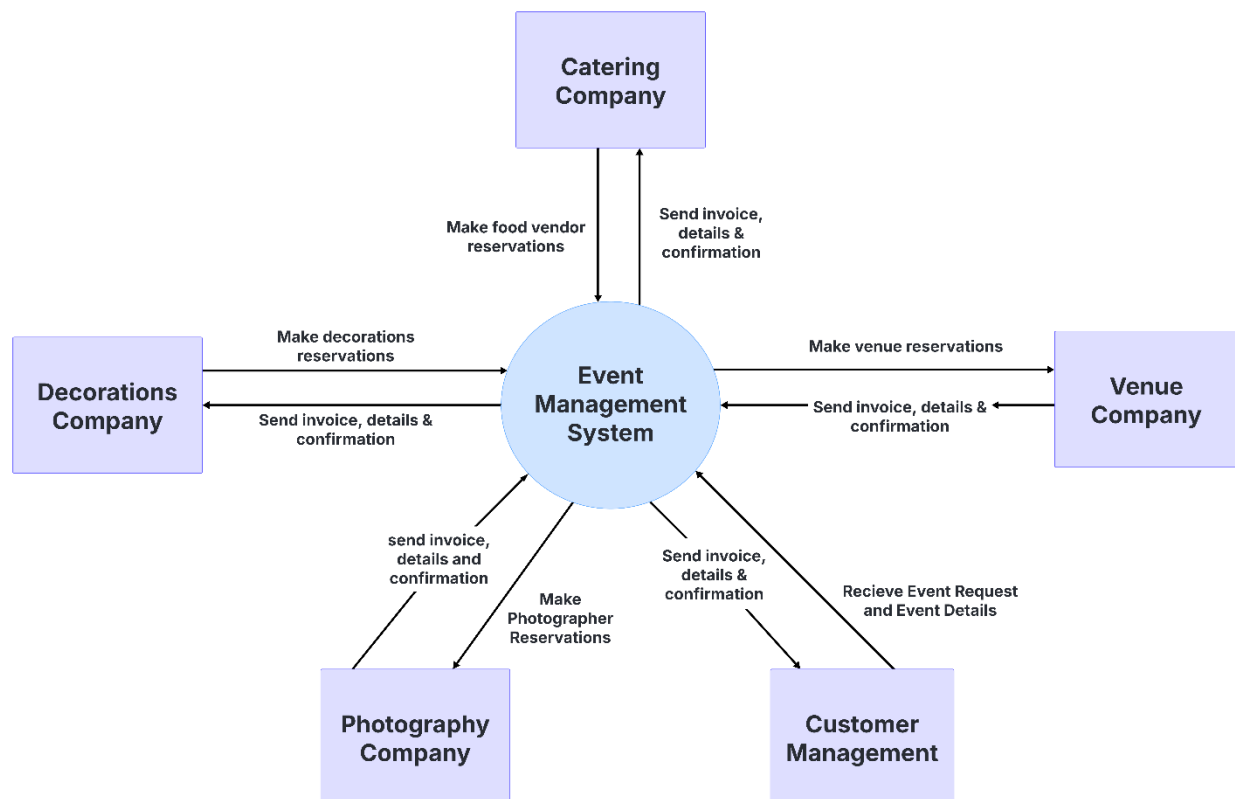
Use Case Diagram

The Use Case Diagram visualizes how the system interacts with various external actors. The diagram helps us understand user needs and structure our features based on function requirements. The use case diagram represents the who, what, and how of the Event Management System.

- **Who** - Primary actors who interact with the system.
    - **Customer**, **Event Manager, and Vendors** (venue company, catering company, photography company, and decorations company).
- **What** - Uses cases, actions, that can be completed in our system.

  Core use cases represented in the diagram include:
    - Send event request, reserve event venue, reserve event catering, rent event decorations, and book photographer
- **How** - The grouping or logical connections of the interactions occurring in the system, each action linked to a relevant actor.
    - For example, the customer is associated with the send event request use case.

The Use Case Description provides more intricate details to capture the functionality of each interaction in the system. This description document provides clarity for the Event Management System development team.

The first mentioned use case, Send Event Request, establishes the baseline for customer interaction and allows the modeling of customer-to-system communication, given the primary actor in this use case to be the customer. Each use-case description includes the primary actor, the trigger, a brief description, the normal flow, sub flows, and occasionally other details.

Use-Case Name: **Send Event Request**	ID: **1**	Importance Level: **High**
Primary Actor: **Customer**	Use Case Type: **Detail**	

Stakeholders and Requests:
Event Management - wants to schedule event for customer
Customer - wants to book event

Brief Description: **This use case describes how we receive a event request and details from the customer**

Trigger: **Customer sends email requests for event**
Type: **External**

Relationship:
    Association: **Customer**
    Include: **Event Request, Event detail, Invoice payment, and event feedback.**

Normal Flow of Events:
1. The customer contacts the event management company and sends request for an event
2. The event manager inquires for details about the event
3. The customer sends event details
4. The event manager inputs details
5. The event manager sends a down payment invoice
6. The customer receives a down payment invoice request
7. The customer sends payment for the invoice request

Subflows:
S-1: If the client is new, create a profile for the client. If the client is returning, update the client profile to "active" status.
S-2: If client does not send event detail within a reasonable amount of time, another event detail request will be sent and the event planning for the client will be made "inactive" until further contact with the client.
S-3: Based on event details received from the customer, the event down payment is calculated (20% of estimated event cost is the necessary down payment).
S-4: If the customer does not fulfill the down payment invoice within a reasonable time, the invoice request will be sent again and the event planning for the client will be made "inactive" until further contact with the client.

Use-Case Name: **Reserve Event Venue**	ID: **2**	Importance Level: **High**
Primary Actor: **Venue Company**	Use Case Type: **Detail**	

Stakeholders and Requests: **Venue Manager - wants to book a venue** **Venue Company - wants to receive venue bookings**
Brief Description: **This use case describes how we book a venue for the event**
Trigger: **Event manager sends an email with event details** Type: **External**

Relationship:
    Association: **Venue Company**
    Include: **Send Invoice and Reserve Venue for event date/time**

Normal Flow of Events:
   1. **The venue manager contacts the venue company to book a venue for date/time**
   2. **The venue manager sends the booking details to the company**
   3. **The company sends booking confirmation and invoice to the venue manager**
   4. **The venue manager send the payment to the company**
   5. **The company send the payment confirmation to the venue manager**

Subflows:
   S-1: **Search for venues**
   S-2: **Confirm event venue availability**
   S-3: **Reserve event space for date/time**
       1. **Send invoice for confirmed appointment**

Use-Case Name: **Reserve Event Catering**	ID: **3**	Importance Level: **High**
Primary Actor: **Catering Company**	Use Case Type: **Detail**	

Stakeholders and Requests: **Catering Manager - wants to book catering** **Catering Company - wants to receive catering booking**
Brief Description: **This use case describes the process of reserving catering service for an event.**
Trigger: **Event Manager sends event with event details** Type: **External**

Relationship:
    Association: **Catering Company**
    Include: **Send Invoice and Reserve Catering for event date/time**

Normal Flow of Events:
   1. **The event manager contacts the catering company to book catering for date/time**
   2. **The catering manager sends the booking details to the company**
   3. **The company sends booking confirmation and invoice to the event manager**

Subflows:
   S-1: **Check catering availability**
   S-2: **Collect event details**
   S-3: **Payment processing and confirmation**
       1. **The event manager receives the invoice and confirms**

Use-Case Name: **Rent Event Decorations**	ID: **4**	Importance Level: **High**
Primary Actor: **Decorations Company**	Use Case Type: **Detail**	

Stakeholders and Requests:
Decoration Manager - wants to rent event decorations
Event Manager - wants to finalize decoration setup

Brief Description: **This use case describes the process of renting decorations for an event.**

Trigger: **Event manager sends a request for decoration rental**
Type: **External**

Relationship:
    Association: **Decorations Company**
    Include: **Send Invoice and Reserve Decorations for event date/time**

Normal Flow of Events:
1. **The event manager contacts the decorations company to reserve decorations for date/time**
2. **The venue manager sends the booking details to the company**
3. **The company sends booking confirmation and invoice to the event manager**

Subflows:
    S-1: **Suggest alternatives if decorations are unavailable**
    S-2: **Apply discount code before payment**
    S-3: **Modify request before payment**
        1. **Send the invoice for confirmed decorations**

Use-Case Name: **Book Photographer**	ID: **5**	Importance Level: **High**
Primary Actor: **Photography Company**	Use Case Type: **Detail**	

Stakeholders and Requests:
Photography Manager - wants to book a photographer
Event Manager - wants photography services for the event

Brief Description: **This use case describes the process of booking a photographer based on event details provided by the event manager.**

Trigger: **Event manager sends event details**
Type: **External**

Relationship:
    Association: **Photography Company**
    Include: **Send Invoice and Reserve Photographer for event date/time**
    Extend:
    Generalization:

Normal Flow of Events:
1. **The event manager contacts the photography company to book a photographer for date/time**
2. **The event manager sends the booking details to the company**
3. **The company sends booking confirmation and invoice to the event manager**

Subflows:
    S-1: **Receive Photography Details**
    S-2: **Confirm Photographer Availability**
    S-3: **Send Booking Confirmation**
        1. **Send invoice for confirmed appointment**

The activity diagrams are designed based on the subsystems that are a part of the BPMN process.

❖ **Requirements Activity Diagram:**

This activity diagram outlines the flow of creating a new event based on requirements gathered from the client, from initial event request to down payment invoice confirmation:

1. The process starts with receiving event requests from the client
2. The event manager looks for the client profile for the client requesting the event
   - If client profile is unavailable, the manager creates a new client profile
   - If client profile is available, the manager attached the new event request to the client's profile
3. The event manager requests event details from the client
4. The manager checks for event details:
   - If unavailable, they request event details again
   - If available, the manager inputs the details into the created event under the client profile
5. The manager calculates the down payment (within the system) for the event request based on event details and requirements given by the client
6. The manager sends the down payment request
7. The manager checks for the receival of the down payment:
   - If unavailable, the event request is rejected
   - If available, confirm that the invoice is completed
8. The process then ends.

Event request, event details, and event down payment

The Logical Flow of Requirements Gathering

Start

Recieve client request for event

Find Client

Create Client — Client is new

Returning Client

Update client profile to "active"

Begin steps for new event planning

Request event details from client

No event details recieved

Event Details

Client sent event information

Input client event details in system

Calculate the down payment

Send invoice for down payment to client

Invoice down payment recieved

Invoice — Invoice not completed

Invoice is completed

Event Request Rejected

Invoice is confirmed

End

❖ **Planning Activity Diagram:**

This activity diagram outlines the flow of planning an event from initial payment to final task assignment:

1. The process starts with receiving the invoice payment.
2. The team identifies event objectives and shares the details with managers.
3. Managers check vendor availability:
   - If unavailable, they review alternatives.
   - If available, vendors are reserved, and an invoice is sent to the client.
4. The system checks if payment is received:
   - If not, the event is put on hold.
   - If yes, the manager proceeds with assignments.
5. The process then ends.

Creating event plan and sending details to other manangers

The Logical Flow of Event Planning

Start → Receive invoice payment → Identify event objectives and goals → Send the information to all the managers → Check with the availability of vendors

- Not available → Review with the manager for another vendor
- Available → Reserve them and generate invoice → Send the invoice to the client → Payment received from the client?
  - No → Event is on hold
  - Yes → Manager assigns responsibilites to it's team → End

24

❖ Payments Activity Diagram:

1. The payments activity starts when the system receives the booking invoice.
2. The invoice details are sent to the customer.
3. The customer validates the invoice.
4. If the customer requires more information, the request is sent back to the system or else the customer proceeds with the payment.
5. After the payment is made, the confirmation is sent back to the system, and it will be verified by the event manager.

Booking invoices, invoice sent to customer, and invoice payment recieved

The Logical Flow of Payments

Start

Receive booking invoice

Send invoice to customer

No

Customer requests for more information

Customer validates invoice

Yes

Customer does the payment

Send payment confirmation to event manager

End

25
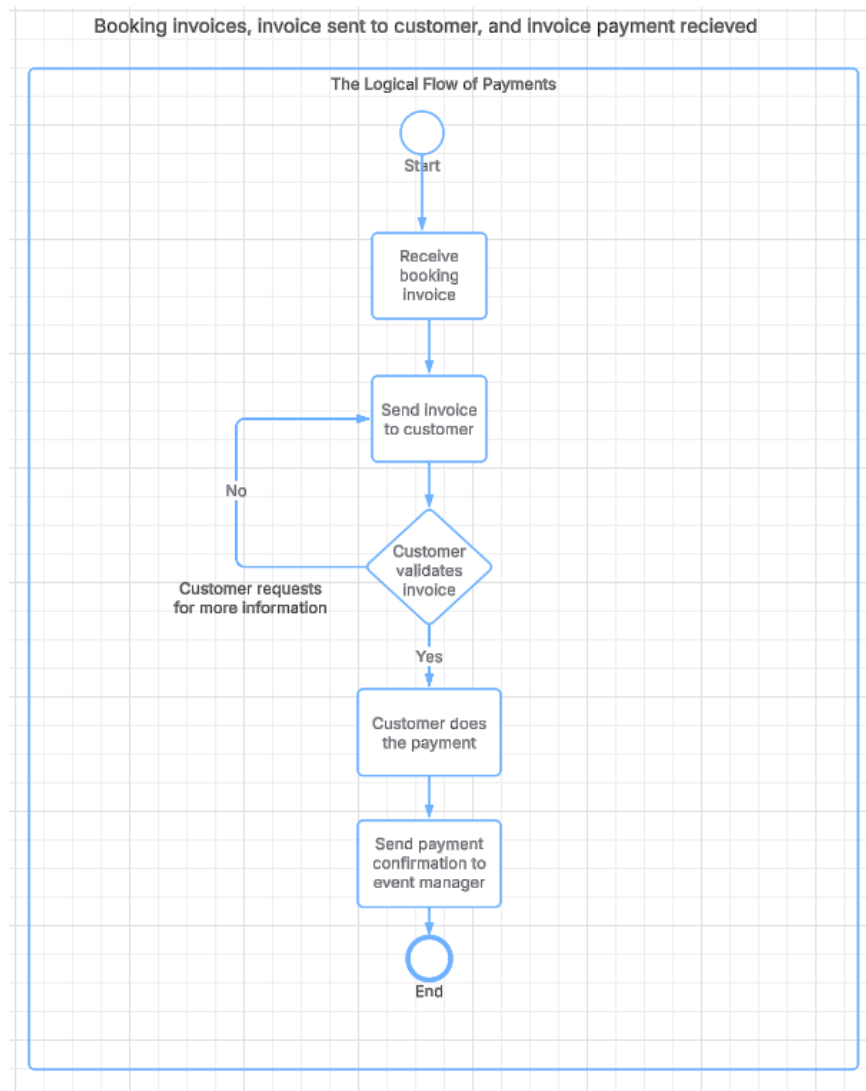
❖ **Booking Activity Diagram:**

This activity diagram outlines the booking process after payment is received:

1. The process starts when the system receives the client's payment.
2. The event manager initiates the booking of the venue, catering vendor, decorator, and photographer.
3. Each booking request is sent to the respective vendors through the system.
4. If payment has not been received initially, the system sends a follow-up payment reminder to the client.
5. Once all vendors confirm their bookings, the event manager finalizes and confirms the overall booking.
6. The process then ends.

Finalizing the Booking after Payment Processing
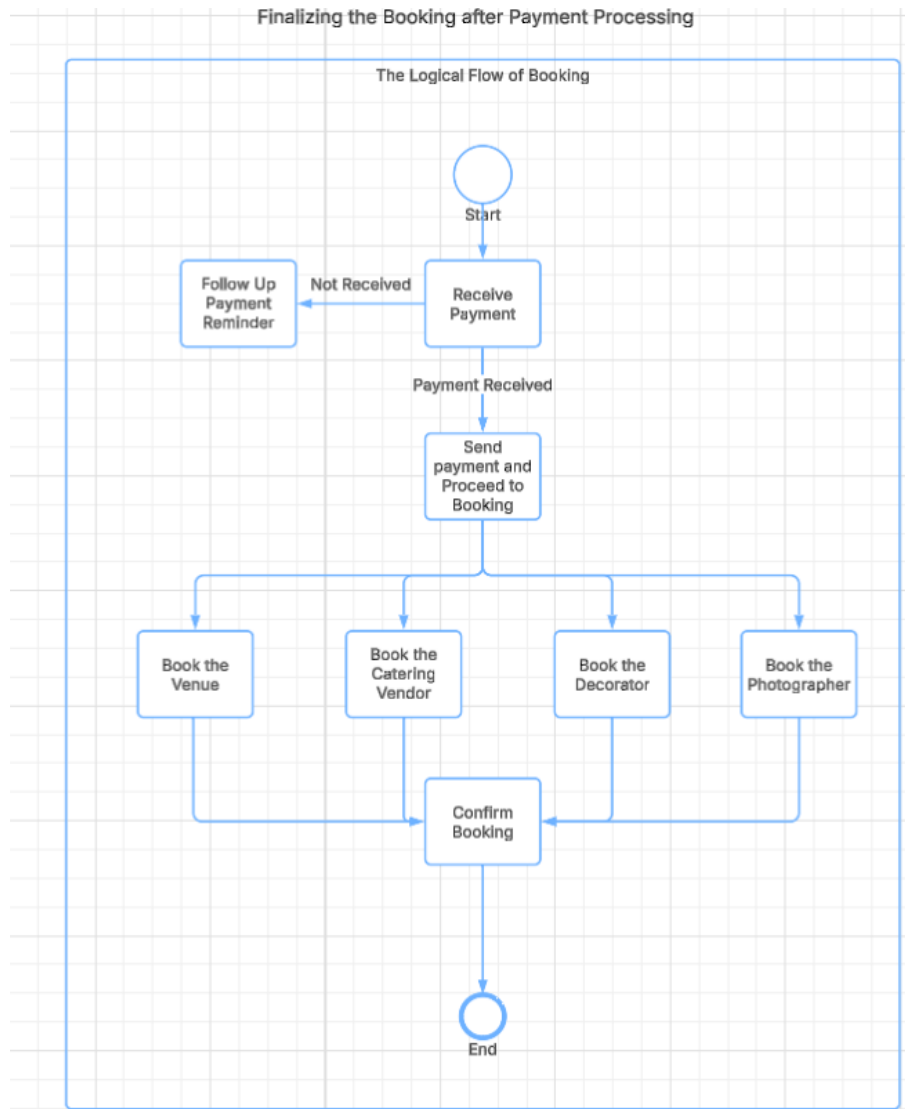
❖ **Feedback Activity Diagram:**

1. The process starts by sending feedback requests to the customer.
2. If there is no response, a reminder is sent to prompt the customer.
3. Once the customer responds, the system receives the customer feedback.
4. The event manager analyzes the feedback received.
5. After analysis, feedback is categorized as either:
   - Positive Feedback:
     o It is documented properly.
     o Then, the process moves to the closure stage.
   - Negative Feedback
     o Improvement actions are taken based on the feedback.
6. After implementing improvements, the process moves to the closure stage.
7. Once documentation or improvement is completed, the process closes. The process finally ends, completing the feedback cycle.

Finalizing the Feedback Process after Collection and Analysis

The Logical Flow of Feedback
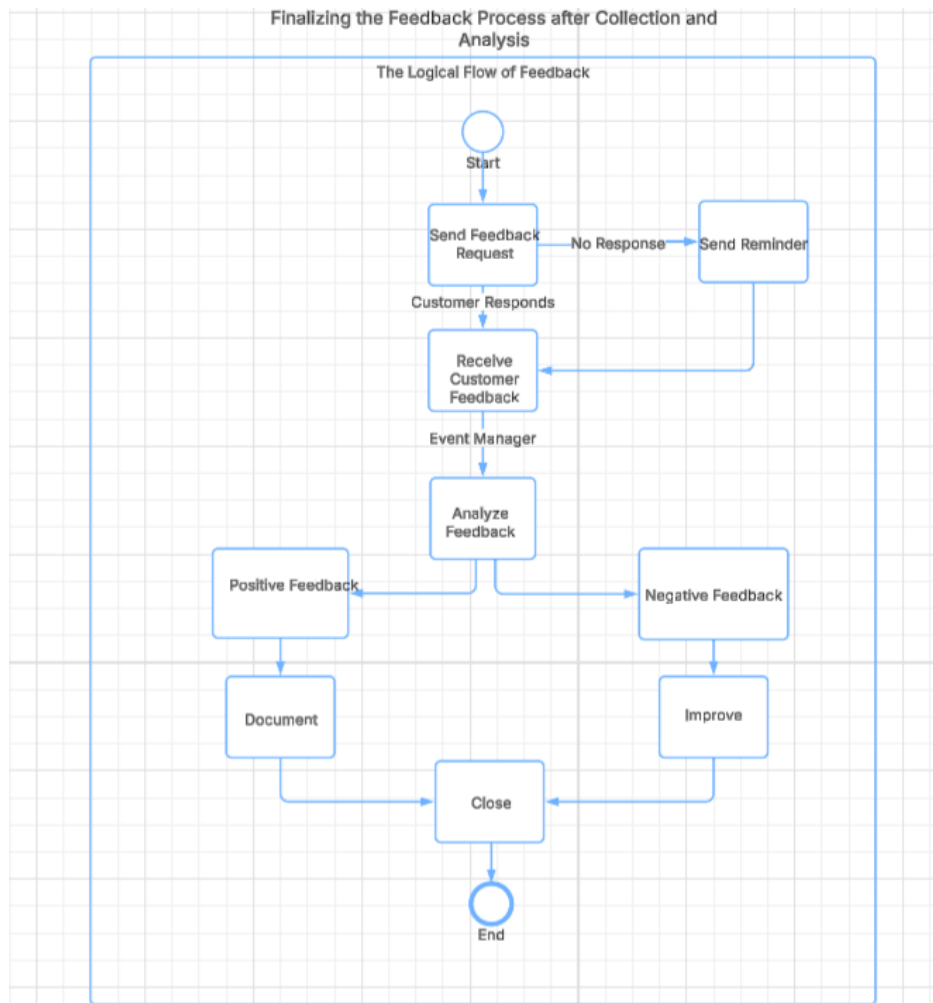
Class Diagram

The Class Diagram represents the object-oriented structure of the Event Management System, showing the various entities (classes), their attributes, behaviors (methods), and relationships among them. The system facilitates event booking and coordination between customers, employees, and service providers.

- "Customer" can book multiple events.
    - Each event has attributes (name, date, location, and status).
- "Invoice" is generated per event, and Payment records capture the transaction details.
- "Feedback" allows customers to rate and review events.
- "Employee" is linked to events via EmployeeEvent, supporting many-to-many assignments.
- "SpecialEvent" handles additional logistics like Venue, Catering, Photography, and Decoration, each represented as separate service provider classes.

This class diagram enables clear modularity and reusability. It maps real-world event management processes into software components, ensuring seamless coordination between users, services, and internal staff. The design supports extensibility by allowing future integration of additional services or event features without disrupting the core structure.
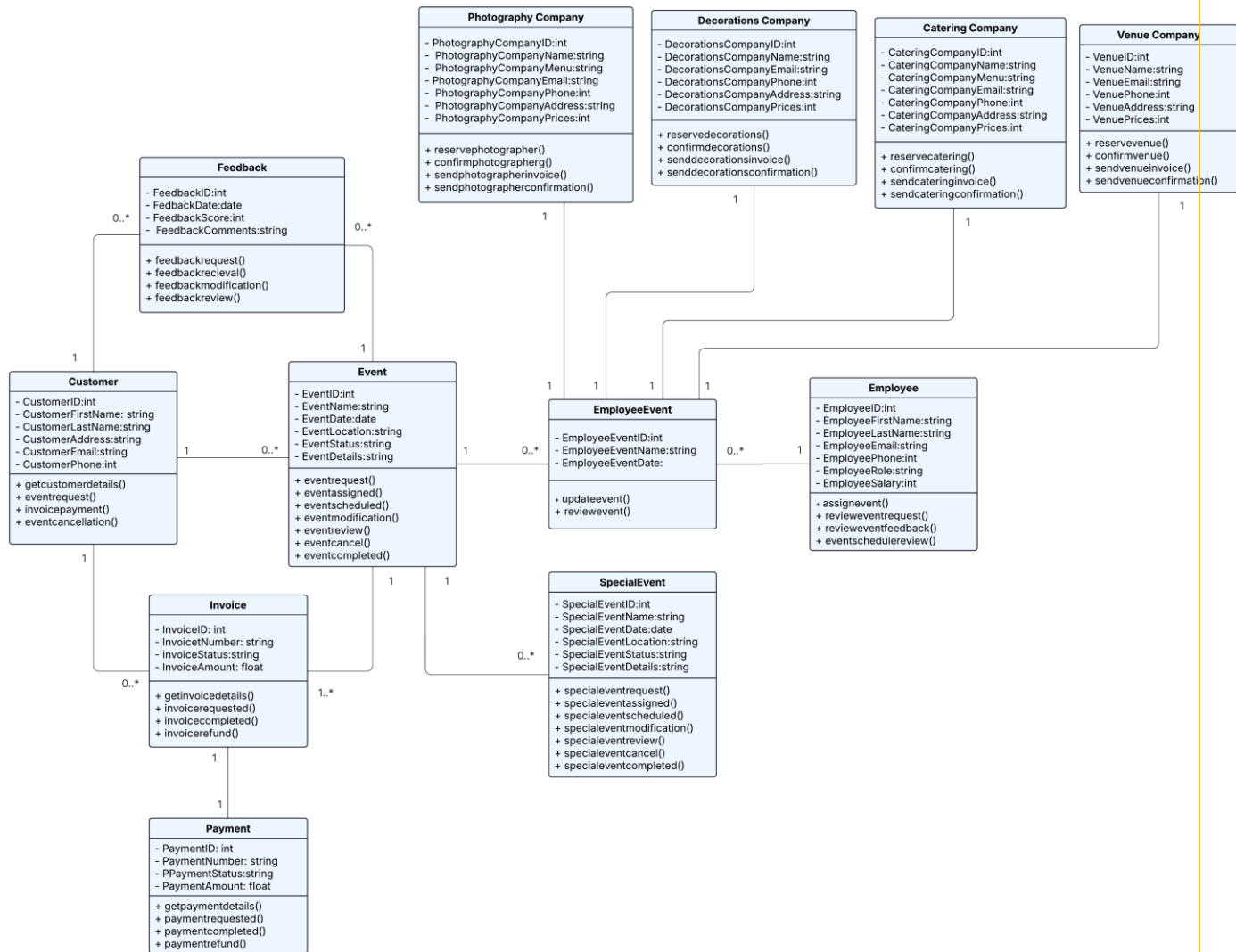
Data Modeling

Entity- Relationship Diagram (ERD)

The Entity-Relationship Diagram (ERD) for the Event Management System captures the key entities involved in organizing events and their relationships. It models how customers interact with the system, how services are assigned to events, and how internal employees manage operations.

1. "Customer" captures client details such as first name, last name, address, email, and phone number. A customer can make multiple bookings, provide feedback, and make payments through associated invoices.

2. "Event" represents the core booking unit. It contains attributes like event name, date, location, status, and details. Each event is linked to one customer (indirectly via invoices) and may receive multiple feedback.

3. "Invoice" stores billing data per event, including invoice number, status, and amount. Each invoice is tied to a single customer and event, while a Payment (identified by payment number and amount) is made for each invoice.

4. "Feedback" allows customers to rate and comment on their experience. It includes feedback score, data, and comments, and is tied to both the customer and the event.

5. "SpecialEvent" handles extra arrangements for an event such as themes or premium features. It includes name, date, status, and location and is associated with only one event.

6. "Employee" stores staff details such as name, role, phone, and salary. Employees are assigned to events through EmployeeEvent, which allows for many-to-many relationships and includes the assignment date.

7. "Venue", "Photography", "Decorations", and "Catering" are service providers. Each entity includes standard attributes like name, email, phone, address, and pricing. One of each can be assigned per event through one-to-one relationships.

Venue
VenueID
VenueName
VenueEmail
VenuePhone
VenueAddress
VenuePrices

Photography
PhotographyID
PhotographyName
PhotographyEmail
PhotographyPhone
PhotographyAddress
PhotographyPrices

Feedback
FeedbackID
FeedbackData
FeedbackScore
FeedbackComments

One event can only have one venue assigned
One venue can only be assigned to one event at a time

One customer can give zero or more feedbacks
A feedback can be received by only one customer

provides

receives

One customer can create zero or more events.
Each event is created by one customer

One event receives zero or more feedbacks.
A feedback when received is associated to one event

An employee event can have only one photographer
A photographer is associated to one event

holds

captures

creates

Customer
CustomerID
CustomerFirstName
CustomerLastName
CustomerAddress
CustomerEmail
CustomerPhone

SpecialEvent
SpecialEventID
SpecialEventName
SpecialEventDate
SpecialEventLocation
SpecialEventStatus
SpecialEventDetails

One event can have one or zero special events associated with it
One special event is associated with one event

Event
EventID
EventName
EventDate
EventLocation
EventStatus
EventDetails

EmployeeEvent
EmployeeEventID
EmployeeEventName
EmployeeEventDate

Employee
EmployeeID
EmployeeFirstName
EmployeeLastName
EmployeeEmail
EmployeePhone
EmployeeRole
EmployeeSalary

specializes

links

manages

An event is linked to zero or more employee events.
An employee events is linked to one event

An employee manages zero or more employee events.
An employee event is managed by one employee

pays

One customer pays zero or more invoices
There is only one customer per invoice

One event generates one or many invoices
An invoice can be linked to a single event

decorates

serves

Invoice
InvoiceID
InvoiceNumber
InvoiceStatus
InvoiceAmount

creates

One invoice can have one payment
One payment can have one invoice

collect

One event can only have one decorations company
One decorations company can only be assigned to decorate one event at a time

Decorations
DecorationsD
DecorationsName
DecorationsEmail
DecorationsPhone
DecorationsAddress
DecorationsPrices

One event can only have one catering company assigned
One catering company can only be assigned to serve one event at a time

Catering
CateringID
CateringName
CateringEmail
CateringPhone
CateringAddress
CateringPrices

Payment
PaymentID
PaymentNumber
PaymentStatus
PaymentAmount

Data Flow Diagram (DFD)

The Data Flow Diagram (DFD) for the Event Management System represents the main processes, data stores, and interactions with external entities. It shows how information moves through the system from event creation to payment completion.

The process begins when a customer creates an event. The event details are stored in the "Event" data store for future use. After creating an event, the customer can make reservations for various services such as Venue, Catering, Photography, and Decorations. Each service has its own reservation process and interacts with its respective external service providers.

Once all required services are reserved, the system generates an invoice based on the selected services and their associated costs. The invoice is then sent to the customer. After receiving the invoice, the customer proceeds with the payment, which is recorded and linked back to the invoice and event information in the system.

The DFD effectively maps the complete flow of information from event creation, through multiple service reservations, to invoice generation and payment ensuring that all important interactions and data movements are captured clearly.
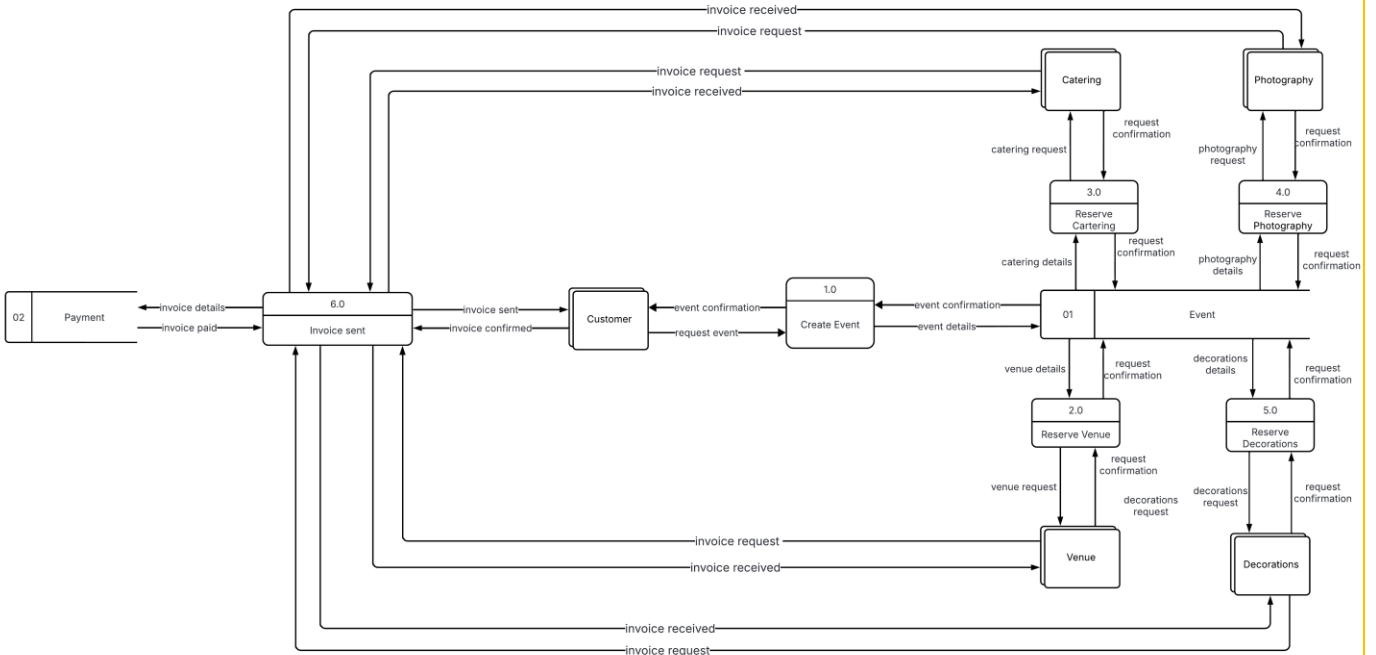
Dynamic Modeling

Sequence Diagram

The sequence diagram is modelled based on the use case descriptions designed for the project.

1. Send Event Request:
   - Participants:
     - Actor: Customer, System: Event Manager
   - Flow:
     - The customer sends an event request to the system along with all the necessary details for the event to be created in the system.
     - After event creation, the system generates an invoice and sends it to the customer for payment.

Send Event Request

```
Customer                                              :System

   │                                                      │
   │              Request (EventID)                       │
   │─────────────────────────────────────────────────────▶│
   │                                                      │
   │              Request (EventDetails)                  │
   │◀ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ │
   │                                                      │
   │              Send (EventDetails)                     │
   │─────────────────────────────────────────────────────▶│
   │                                                      │
   │       Send Invoice (InvoiceID, InvoiceAmount)        │
   │◀ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ │
   │                                                      │
   │  Complete Invoice (InvoiceID, InvoiceAmount, InvoiceStatus)  │
   │─────────────────────────────────────────────────────▶│
   │                                                      │
   │       Send Invoice (InvoiceID, InvoiceAmount)        │
   │◀ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ │
   │                                                      │
   │  Complete Invoice (InvoiceID, InvoiceAmount, InvoiceStatus)  │
   │─────────────────────────────────────────────────────▶│
   │                                                      │
```
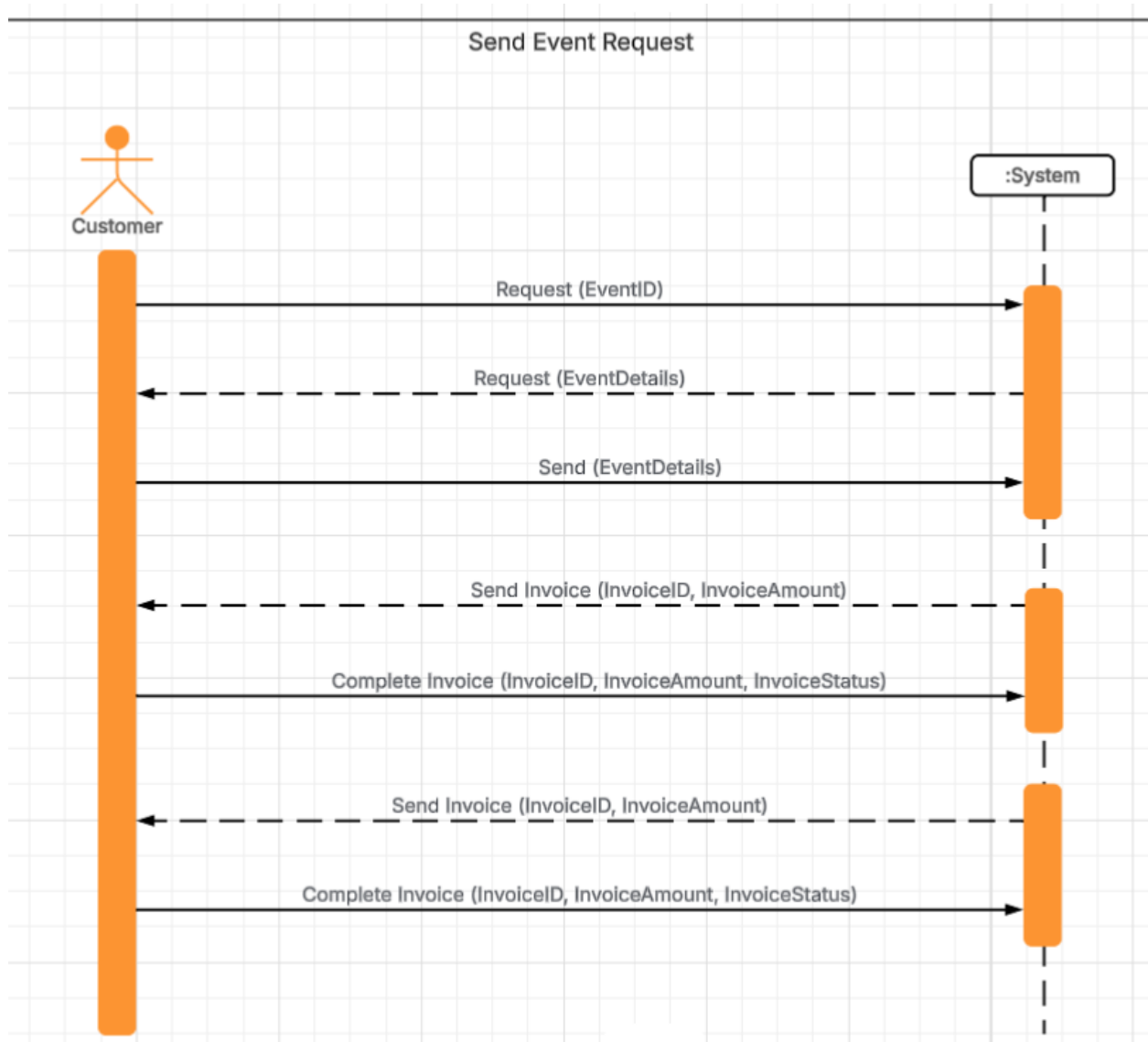
2. Reserve Event Venue:
   - Participants:
     - Actor: Venue Company, System: Venue Manager
   - Flow:
     - The system sends the event and venue request details to the venue company.
     - After booking, the venue company sends the venue confirmation details and invoice details back to the system.
     - The system completes the payment and receives confirmation back from the company.

Reserve Event Venue
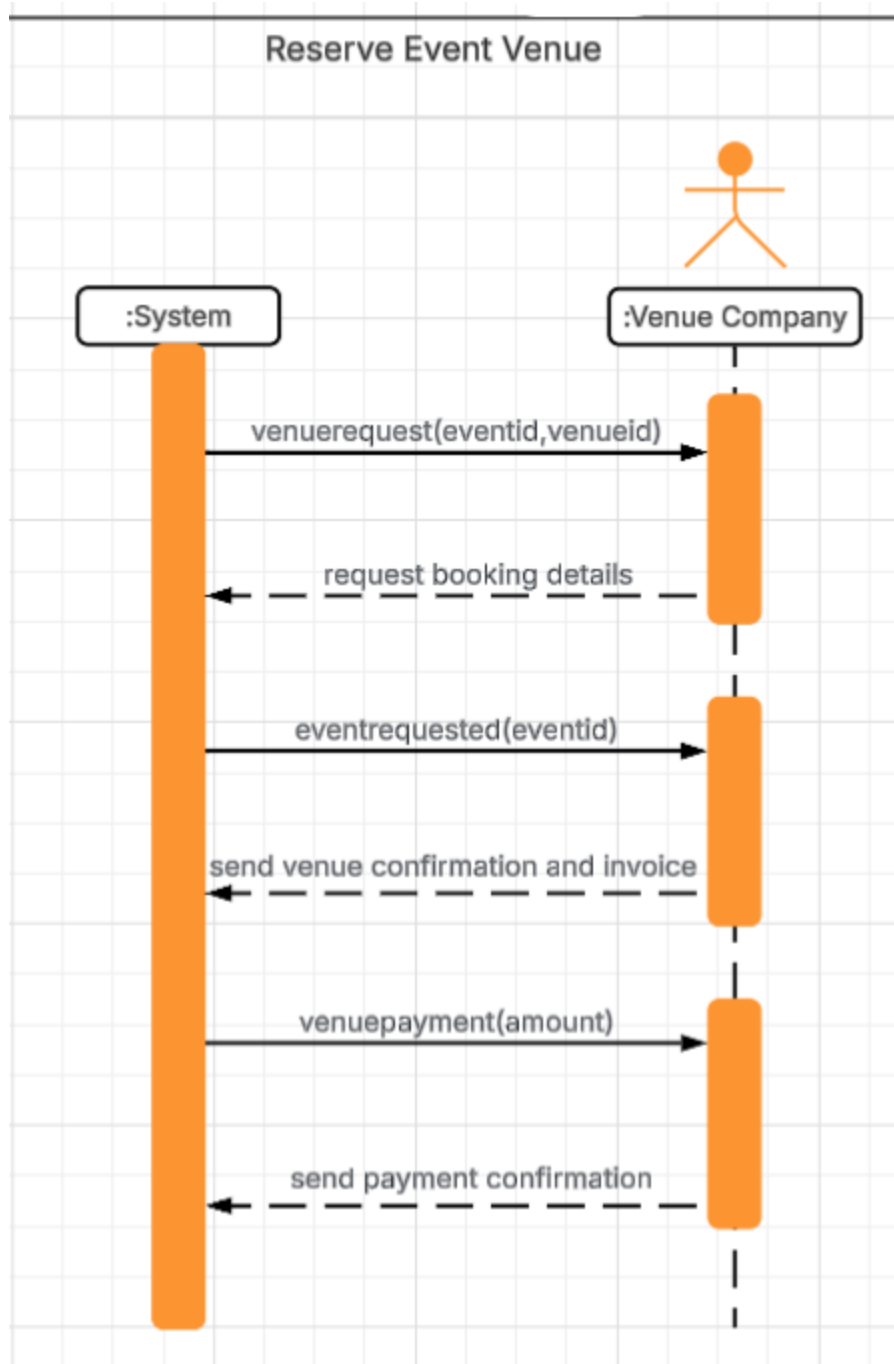
The sequence for Reserve Event Catering, Reserve Event Decorations and Reserve Event Photographer will follow the similar sequence as the Reserve Event Venue.

The state chart diagrams are designed based on the state changes that occur in the subsystems that are a part of the BPMN process.

1. Requirements Gathering State Diagram:
The states in the requirements gathering subsystems are as follows:
- Event Requested: The initial state when the client sends a event request
- Client Profile Created: The state after the client is created in the system
- Client Profile Updated: The state after the client is updated in the system
- Details Requested: The state after the details are requested from the client
- Details Received: The state after the client sends details to the system
- Details Recorded: The state after the client details are recorded into the system
- Invoice Sent: The state after the invoice is sent to the client
- Invoice Rejected: The state after the invoice is not fulfilled by the client
- Event Request Rejected: The state after the event invoice is canceled
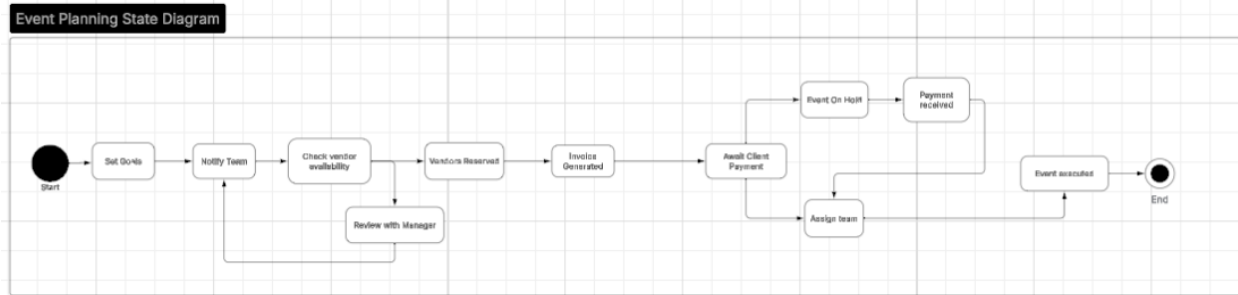- Payment Confirmed: The state after the invoice is completed by the client
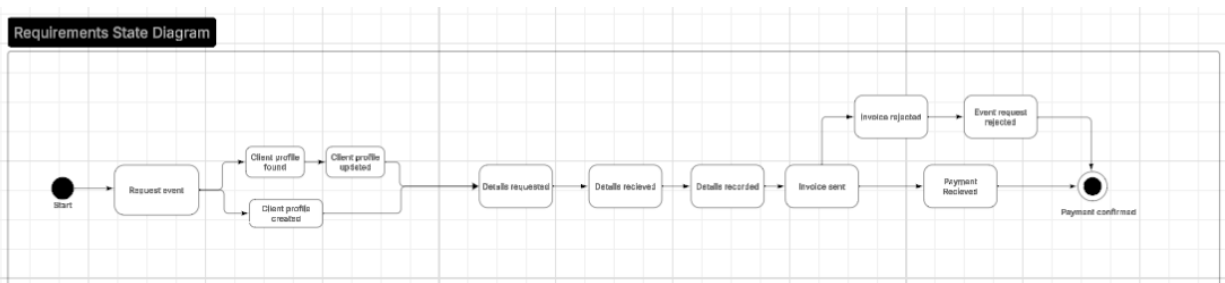
2. Planning State Diagram:
The states in the planning subsystems are as follows:
- Start: The process begins when an event request is initiated.
- Set Goals: Event objectives and requirements are defined.
- Notify Team: The internal team is informed about the upcoming event.
- Check Vendor Availability: The team verifies the availability of vendors.
- If vendors are unavailable, the process loops to Review with Manager and then back to Notify Team.
- Vendors Reserved: Once vendors are confirmed, bookings are made.
- Invoice Generated: The system generates an invoice for the client.
- Await Client Payment: The system waits for the client's payment.
- If payment is not received, the event moves to Event On Hold.
- If payment is received, the process continues to the next step.
- Assign Team: Once payment is confirmed, responsibilities are assigned.

- Event Executed: The event is successfully conducted

Event Planning State Diagram

3. Payments State Diagram:

The states in the payment subsystem are as follows:
- Invoice Generated: The initial state when the event manager generates the invoice.
- Invoice Received: The state after the customer receives the invoice from the system.
- Invoice Reviewed: The state after the customer reviews the invoice and validates the details.
- Invoice Rejected: The state when the customer rejects the invoice and requests more information.
- Invoice Approved: The state when the customer approves the invoices and proceeds for payment.
- Invoice Paid: The state where the customer completes the payment.
- Payment Confirmed: The final state after the confirmation is sent to the event manager/by the customer


Payments State Diagram
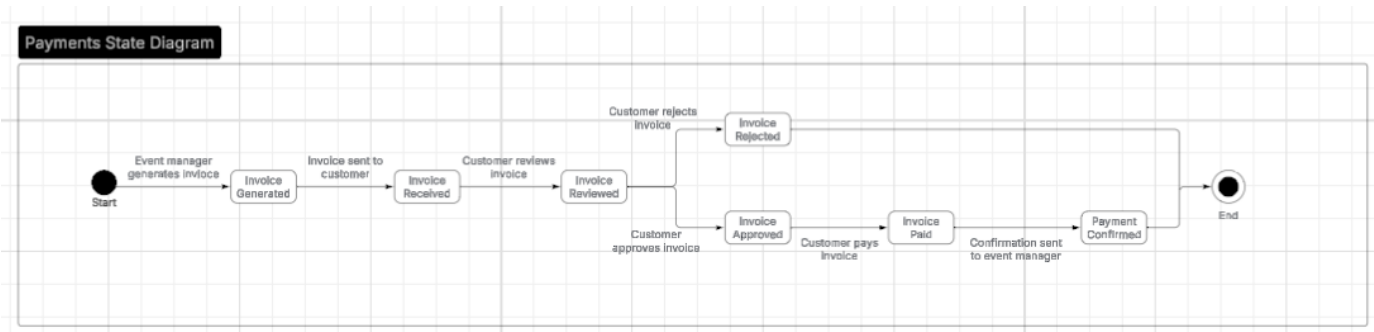
4. Booking State Diagram:

The states in the booking subsystem are as follows:
- Awaiting Payment: The initial state after an event request is created.
- Payment Not Received: If payment is not completed, a follow-up reminder is triggered.
- Payment Received: The state where the client completes the payment and the system proceeds.

- Booking Vendors: The state where the event manager books venue, catering, decoration, and photographer.
- Booking Confirmed: Each vendor confirms their individual booking.
- All Vendors Booking Confirmation Received: All vendor bookings are successfully confirmed.
- All Booking Completed: The final state where all bookings are completed and the event is finalized.

Booking State Diagram
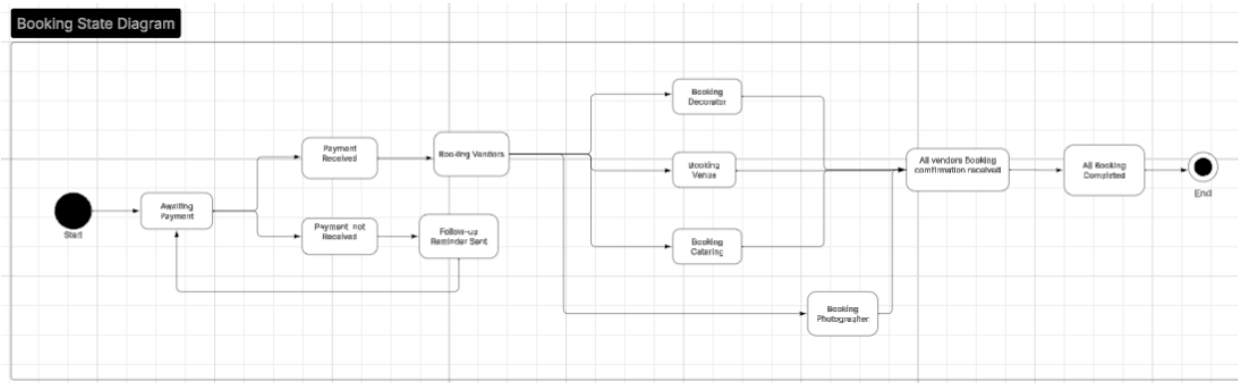
5. Feedback State Diagram:

The states in the feedback subsystem are as follows:
- The initial state when feedback is first provided by the customer.
- The state when feedback is saved as a draft and not yet finalized or submitted.
- The state when the drafted feedback is completed and submitted for internal review.
- The state where feedback has been reviewed by an authorized reviewer to check for validity and completeness.
- The state where reviewed feedback is rejected and requires further revision or clarification.
- The state where the reviewed feedback meets all criteria and receives formal approval.
- The final state is when feedback is officially published or shared with the intended recipients.

Feedback State Diagram

37
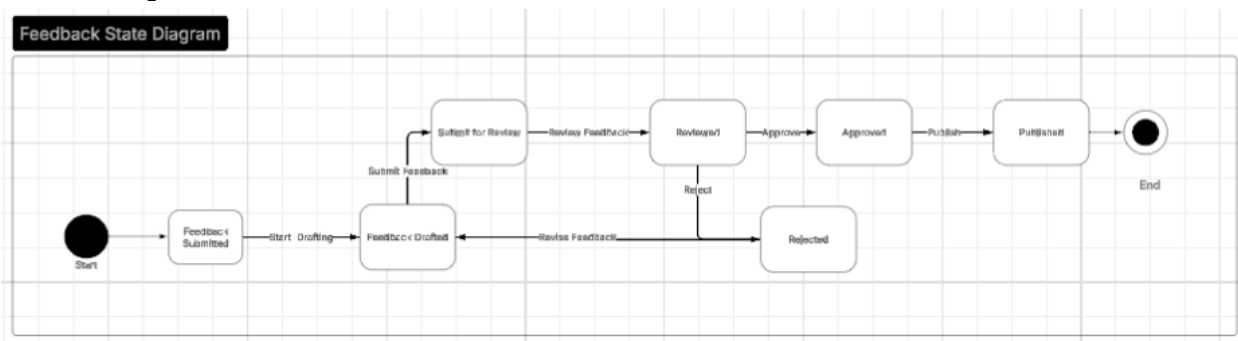
UI Design

The Event Management System UI, designed using UXPilot.ai and Figma, focuses on simplicity, clarity, and responsiveness. The dashboard offers a quick overview of events, attendees, and revenue, enabling fast actions. Users can easily create and manage events through a clean step-by-step form.

The UI design for *Planora*, an event management system, was crafted to deliver a seamless, user-friendly experience for clients planning personal and corporate events. Built in Figma, the design focuses on intuitive navigation, clean visual hierarchy, and responsive layout principles.

The landing page features a modern, minimalistic design that highlights core services such as venue booking, vendor coordination, and scheduling. A clear call-to-action ("Book Now") is placed above the fold to encourage immediate engagement. The color palette balances professionalism and vibrance to appeal to a wide demographic, while consistent typography enhances readability.

The dashboard interface offers clients a consolidated view of ongoing and past events, integrated calendar scheduling, and real-time task updates. The sidebar menu supports easy access to major modules like Bookings, Tasks, Budgeting, and Communication. Each module is designed with user context in mind—leveraging components such as dropdown selectors, status tags, and modal popups to simplify complex interactions.

Accessibility considerations include sufficient color contrast, scalable text, and logical tab order. The design also supports mobile responsiveness to cater to on-the-go users, making Planora adaptable across devices.

Overall, Planora's UI emphasizes clarity, functionality, and aesthetic balance, ensuring both novice and experienced users can navigate and manage events efficiently.

Figma Link
Prototype
https://www.figma.com/design/g1gJwZGoaVKGzTNmZV86Fo/Planora?node-id=0-1&t=oFwS7Ljr3J97YHlN-1

Responsive Link
https://www.figma.com/proto/g1gJwZGoaVKGzTNmZV86Fo/Planora?page-id=0%3A1&node-id=25-2169&p=f&viewport=187%2C61%2C0.1&t=pK9k6SXZBcLaHtmd-1&scaling=min-zoom&content-scaling=fixed&starting-point-node-id=4%3A203
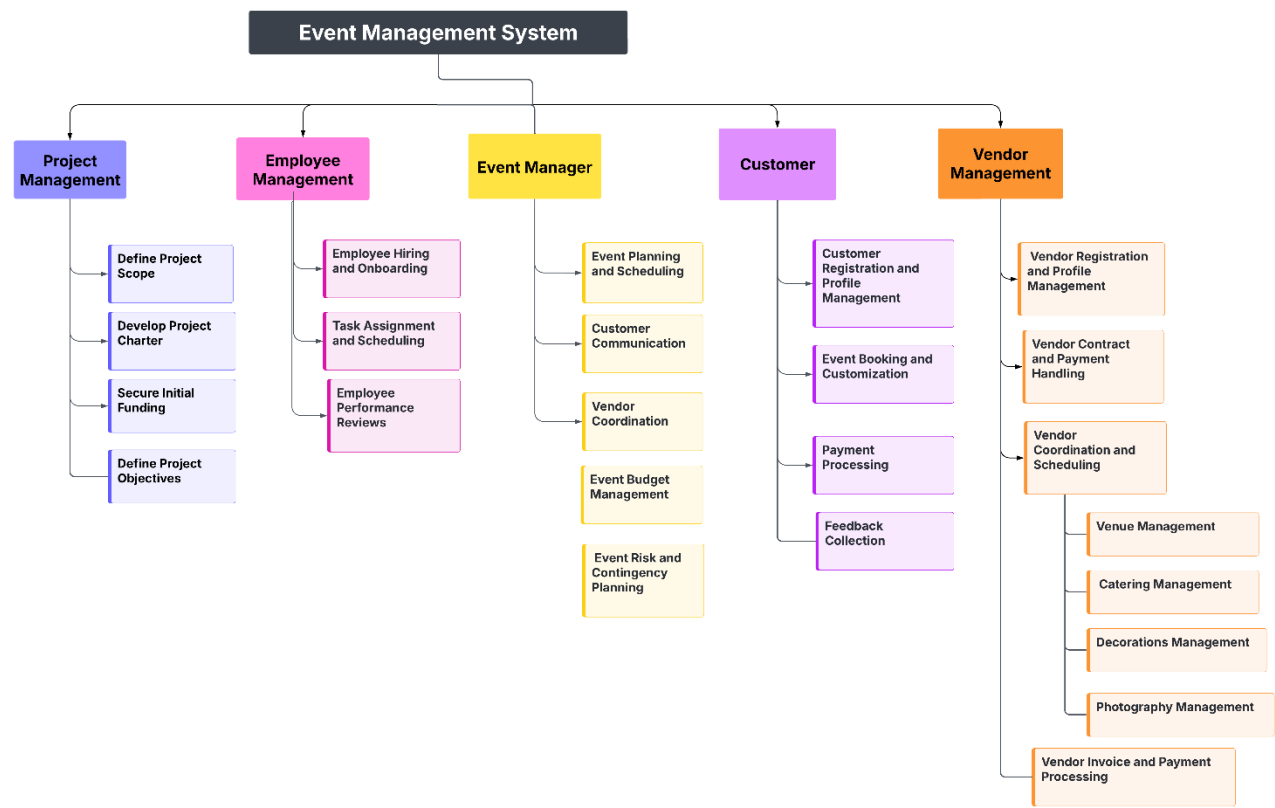
Project Management

Gantt Chart

The Gantt chart is designed based on the tasks performed to design the event management system.

Stage	Task Name	February Week 1	Week 2	Week 3	Week 4	March Week 1	Week 2	Week 3	Week 4	April Week 1	Week 2	Week 3	Week 4	May Week 1	Week 2	Week 3	Week 4
Requirements Gathering & Planning	Project Planning & Charter																
	Financial Planning		▓	▓													
	Resource Allocation		▓	▓													
	Gantt Chart		▓	▓													
	WBS			▓													
Analysis & Design	BPMN				▓	▓	▓	▓									
	Context Diagram					▓	▓	▓									
	Use Case Diagram						▓	▓									
	Use Case Description						▓	▓									
	Activity Diagram								▓	▓							
	Class Diagram								▓	▓							
	ERD									▓							
	DFD									▓							
	Sequence Diagram										▓						
	Statechart Diagram										▓						
Development & Testing	User Interface (UI)										▓	▓					
Implementation	Project Write Up									▓	▓	▓	▓	▓			
	Video									▓	▓	▓	▓	▓			
	Final Report												▓	▓	▓		

Work Breakdown Structure

Resource Allocation

The success of the Event Management System (EMS) project was strongly dependent on the careful allocation of human resources, technology, and time. A balanced and flexible approach to resource management ensured that project milestones were met effectively despite potential risks

Human Resources

The project team was structured with defined roles to minimize overlap and maintain accountability throughout the development cycle:

- **Project Manager**: Responsible for coordinating activities, facilitating communication between stakeholders, and ensuring project deadlines were achieved.
- **Backend Developer**: Developed and maintained the system's database and server-side logic using PostgreSQL and Node.js.
- **Frontend Developer**: Designed and implemented the user interface and front-end logic using React.js.
- **UX/UI Designer**: Created wireframes and mockups to guide the user experience and maintain a cohesive design standard.

- **QA Tester**: Conducted functional and non-functional testing, reported issues, and worked closely with developers to resolve bugs and ensure system stability.

In the event of unforeseen availability issues, contingency plans were in place to redistribute tasks among the remaining team members. This flexibility helped maintain progress without compromising quality.

Tools and Technologies

The team selected technologies based on reliability, scalability, and compatibility with project requirements:

- **React.js** for front-end development
- **Node.js** and **PostgreSQL** for backend services and data storage
- **Firebase** to enable real-time notifications within the system
- Collaborative and version control tools to support efficient team workflow

The use of open-source and widely supported technologies helped meet project goals while maintaining cost-effectiveness.

Time and Task Allocation

The EMS project followed a structured timeline with key phases aligned to the project milestones:

Phase	Timeline	Key Deliverables
Planning	February	Project charter, scope definition
Design	March	Wireframes, UI prototypes
Development	April	Backend and frontend development
Testing & Finalization	May	System testing, debugging, documentation

Throughout the project, the team conducted regular meetings to review progress, identify potential blockers early, and adjust as needed to stay aligned with the project plan.

Budget Estimation

The project budget was estimated based on realistic assumptions for a small software development team using market average rates and typical cloud and software service costs. The main cost drivers and assumptions are as follows:

- **Human Resources**: Includes five team members (Project Manager, Backend Developer, Frontend Developer, UX/UI Designer, and QA Tester) for four months. The estimation is based on average U.S. market monthly rates per role.
- **Cloud Hosting**: Covers anticipated expenses for cloud infrastructure such as servers, storage, databases, and bandwidth over the project period.
- **Tools & Licenses**: Accounts for collaboration and design tool subscriptions such as private repositories, project management tools, and design software.
- **Contingency**: A standard 10% contingency reserve is included to accommodate unforeseen costs or delays.

Category	Description	Estimated Cost (USD)
Human Resources	1 Project Manager, 2 Developers, 1 Designer, 1 QA Tester for 4 months	$120,000
Cloud Hosting	Servers, storage, and bandwidth	$4,000
Tools & Licenses	Version control, collaboration platforms	$1,500
Contingency	10% of total estimated project cost	$12,550
Total Estimated Cost		$138,050

Project Costing

The project costing is based on the Cost-plus method.

These are the actual, estimated expenses needed to execute the project.
Overhead accounts for indirect costs like office space, utilities, admin staff, etc.
- **Assumed Rate:** 15% of total direct costs
- **Calculation:** 15% × $138,050 = $20,708

This is the markup added for profit (a common part of cost-plus contracts).
- **Assumed Rate**: 20% of (direct + overhead)
- **Base for Profit**: $138,050 + $20,708 = $158,758
- **Calculation**: 20% × $158,758 = **$31,752**

Total Project Cost"
- **Direct**: $138,050
- **Overhead**: $20,708
- **Profit**: $31,752
- **Final Total**: **$190,510**

Category	Description	Amount (USD)
Direct Costs	Human Resources (1 PM, 2 Devs, 1 Designer, 1 QA for 4 months)	$120,000
Direct Costs	Cloud Hosting (Servers, storage, bandwidth)	$4,000
Direct Costs	Tools & Licenses (Version control, collaboration tools)	$1,500
Direct Costs	Contingency (10% of above subtotal)	$12,550
	Total Direct Costs	**$138,050**
Overhead	Overhead (15% of direct costs)	$20,708
Profit	Profit (20% of direct + overhead)	$31,752
	Total Project Cost	**$190,510**

Data Security

Data security is crucial to protect sensitive data like user information, event details and payment processing. The following are the security measures implemented:

1. **Authentication and Authorization:** Users will be required to create an account that will be password protected to validate entry into the system.
2. **Data Encryption:** Data entered the system will be encrypted for secure communication during storage and transfer.
3. **Access Control:** Access will be restricted where required, to ensure only the personnel with the required access can view the event and user details.
4. **Secure Payment Processing:** Payment processing gateway will be protected to ensure secure transactions.
5. **Audit and Compliance:** Regular audits will be performed to ensure the system is secure with no vulnerabilities and in accordance with the privacy rules.

Implementation of AI

Unique Selling Point (AI Budget feature)

The unique selling point of the project is its AI-powered smart budgeting, cost optimization and real-time tracking of expenditures.

The platform provides cost analysis to help organizers maximize ROI on event spending based on the budget specified by the user.

It also allows organizers to focus on creativity and event execution without worrying about going over budget by providing smart budgeting tools that suggest cost-effective solutions, and real time tracking of expenditures.

Project Video Link

https://youtu.be/b8sbYFkJatU

Conclusion

In conclusion, the event management system enables users to request events and be a part of the planning process. It also offers a smooth payment and feedback process. With features like smart budgeting tools, cost-saving analysis and real-time tracking, it helps organizers to customize the planning, stay on budget and ensure they deliver the best value to the client.