**Waterfall Model Documents – Part 2**

**Document 6**

**Answer.**

Use Case Diagram

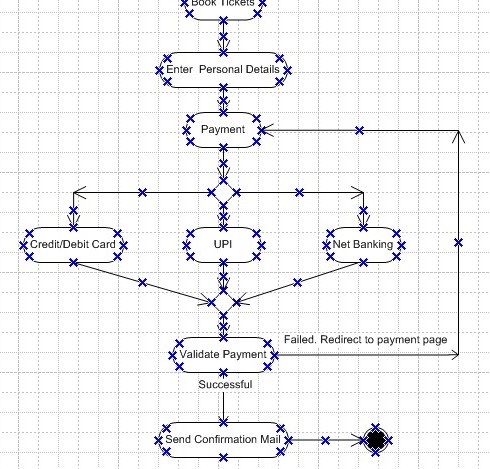
A diagram of a diagram of a business

Description automatically generated with medium confidence

Activity Diagram

A diagram of a data flow

Description automatically generated



**Use Case Specification Document**

**Use Case 1:** Event Ticket Booking System

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Id | PCE001 | | |
| Use Case Name | Event Ticket Booking System | | |
| Created by | Harsh | Last Updated | 03/01/2025 |
| Date Created | 01/01/2025 | Last Revision Date | 04/01/2025 |
| Actors | User, Admins and Payment Gateway | | |
| Description | This use case describes the process through which a user searches for event, view details, books ticket, makes payment and receives confirmation. | | |
| Basic Flow | 1. User logs into the system. 2. User searches for an event using filters (e.g., date, location). 3. User selects an event to view details. 4. User books tickets by specifying the number of attendees. 5. System displays the total price and redirects to the payment gateway. 6. User makes the payment. 7. Payment gateway confirms the transaction. 8. System generates a confirmation email and ticket for the user. | | |
| Alternate Flow | Alternate Flow 1: User Browses Without Login:   * User can browse events without logging in. * User is prompted to log in only during ticket booking.   Alternate Flow 2: Payment with Promo Code:   * User applies a valid promo code. * System recalculates the total price with the discount. | | |
| Exceptional Flow | Exceptional Flow 1: Payment Failure:   * System notifies the user of payment failure and provides options to retry or cancel the booking.   Exceptional Flow 2: Event Sold Out:   * System informs the user that tickets are no longer available.   Exceptional Flow 3: Session Timeout:   * If the user is inactive for a specified time, the session expires, and the user is redirected to the login page. | | |
| Pre-Conditions | * User must have a registered account. * Event details must be available in the system. * Payment gateway must be operational. | | |
| Post – Conditions | * User receives a confirmation email and ticket. * System updates the ticket inventory. | | |
| Assumptions | * Users have internet access to interact with the platform. * The payment gateway is reliable and secure. | | |
| Constraints | * The platform supports only modern browsers and devices. * Ticket booking is restricted to 10 tickets per transaction. | | |
| Dependencies | * Integration with the payment gateway for processing transactions. * Event data provided by the admin must be accurate. | | |
| Inputs and Outputs | Inputs:   * Event search criteria (date, type, location). * User ticket selection and payment details.   Outputs:   * Confirmation email and digital ticket. | | |
| Business Rules | * Promo codes must be validated before applying the discount. * Payments are refundable only up to 48 hours before the event. * Tickets are non-transferable after booking. | | |
| Miscellaneous Information | Notifications:   * Users receive email/SMS reminders about the event.   Accessibility:   * The platform supports screen readers and high-contrast modes for differently abled users. | | |

**Use Case 2**: Event Management by Admin

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Id | PCE002 | | |
| Use Case Name | Event Management by Admin | | |
| Created by | Harsh | Last Updated | 03/01/2025 |
| Date Created | 01/01/2025 | Last Revision Date | 04/01/2025 |
| Actors | Admin and System | | |
| Description | This use case describes how the admin creates, updates, and deletes event information on the platform | | |
| Basic Flow | 1. Admin logs into the system. 2. Admin navigates to the event management dashboard. 3. Admin selects the option to create a new event or edit/delete an existing one. 4. System validates the data and saves updates. | | |
| Alternate Flow | * Admin Adds Optional Details: Admin provides optional details like speaker bio, promotional videos, and sponsors. | | |
| Exceptional Flow | * Incomplete Data: System prompts the admin to complete required fields before proceeding. | | |
| Pre-Conditions | * Admin has valid credentials. * Event categories are pre-configured in the system | | |
| Post – Conditions | * Event details are updated in the system and visible to users. | | |
| Assumptions | * Admin has sufficient access rights. | | |
| Constraints | * Event descriptions are limited to 500 characters. | | |
| Dependencies | * Event data must be provided by the client | | |
| Inputs and Outputs | 1. Input: Event details. 2. Output: Updated event information visible on the platform. | | |
| Business Rules | * Events cannot overlap in time for the same venue. | | |
| Miscellaneous Information | * Admin receives confirmation upon successful update. | | |

**Use Case 3**: Feedback Submission

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Id | PCE003 | | |
| Use Case Name | Feedback Submission | | |
| Created by | Harsh | Last Updated | 03/01/2025 |
| Date Created | 01/01/2025 | Last Revision Date | 04/01/2025 |
| Actors | User and System | | |
| Description | This use case describes how users provide feedback after attending an event. | | |
| Basic Flow | 1. User logs in and selects an event they attended. 2. User fills out the feedback form and submits it. 3. System records the feedback and displays a success message. | | |
| Alternate Flow | * Anonymous Feedback: User submits feedback without logging in (only for general suggestions). | | |
| Exceptional Flow | * Feedback Submission Failure: System displays an error message and prompts the user to retry. | | |
| Pre-Conditions | * The event must be marked as attended by the user. | | |
| Post – Conditions | * Feedback is saved in the system and accessible for admin review. | | |
| Assumptions | * User is willing to provide honest feedback. | | |
| Constraints | * Feedback is limited to 300 words. | | |
| Dependencies | * Feedback questions must be pre-configured by the admin. | | |
| Inputs and Outputs | 1. Input: Feedback text, ratings. 2. Output: Stored feedback entry. | | |
| Business Rules | * Feedback cannot be edited once submitted. | | |
| Miscellaneous Information | * System sends acknowledgment to the user for their feedback. | | |

**Use Case 4**: User Profile Management

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Id | PCE004 | | |
| Use Case Name | User Profile Management | | |
| Created by | Harsh | Last Updated | 03/01/2025 |
| Date Created | 01/01/2025 | Last Revision Date | 04/01/2025 |
| Actors | User and System | | |
| Description | This use case describes how users update their profile information on the platform. | | |
| Basic Flow | 1. User logs in and navigates to their profile page. 2. User updates their personal details. 3. System validates the changes and saves the updates. | | |
| Alternate Flow | * Partial Update: User updates only specific fields. | | |
| Exceptional Flow | * Validation Error: System rejects invalid input (e.g., incorrect email format). | | |
| Pre-Conditions | * User is logged in. | | |
| Post – Conditions | * Updated profile details are saved in the system. | | |
| Assumptions | * Users will keep their profiles updated. | | |
| Constraints | * Certain fields (e.g., email) must be unique. | | |
| Dependencies | * System must allow profile updates. | | |
| Inputs and Outputs | 1. Input: Updated user details. 2. Output: Confirmation of saved changes. | | |
| Business Rules | * Users cannot change their registered email address directly. | | |
| Miscellaneous Information | * Profile updates are logged for audit purposes. | | |

**Use Case 5**: Notifications and Reminders

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Id | PCE005 | | |
| Use Case Name | Notifications and Reminders | | |
| Created by | Harsh | Last Updated | 03/01/2025 |
| Date Created | 01/01/2025 | Last Revision Date | 04/01/2025 |
| Actors | User and System | | |
| Description | This use case describes how users receive event notifications and reminders. | | |
| Basic Flow | 1. User opts in to receive notifications. 2. System sends reminders about upcoming events via email/SMS. | | |
| Alternate Flow | * User Opts Out: User disables notifications in their profile settings. | | |
| Exceptional Flow | * Notification Delivery Failure: System retries sending the notification after a specified interval. | | |
| Pre-Conditions | * User has provided contact information (email/SMS). | | |
| Post – Conditions | * Notifications are delivered successfully. | | |
| Assumptions | * Users will check their notifications regularly. | | |
| Constraints | * Notifications are limited to one reminder per event per day. | | |
| Dependencies | * Integration with third-party notification services. | | |
| Inputs and Outputs | 1. Input: Event details, user contact preferences. 2. Output: Delivered notifications. | | |
| Business Rules | * Notifications are sent only to registered users who have opted in. | | |
| Miscellaneous Information | * Admins can view notification delivery logs. | | |

**Document 7. Balsamiq Mockups**

**A screenshot of a computer

Description automatically generatedA computer screen shot of a computer

Description automatically generatedA computer screen with a hand pointing to a button

Description automatically generatedA computer screen shot of a computer screen

Description automatically generatedA computer screen with a hand pointing to a payment page

Description automatically generatedA computer screen shot of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generated**

**Document 8 – Tools – Visio, Axure and Balsamiq**

Answer. Using **Visio, Axure**, and **Balsamiq** for the **Prerna Centre for Excellence** project has significantly enhanced the clarity and quality of deliverables. **Visio** excelled in creating detailed process flow diagrams, activity diagrams, and stakeholder maps, offering a structured way to visualize workflows and dependencies. Visio was instrumental in visually mapping out the workflows and processes for the project. We used it to design process flow diagrams for the legacy system (AS-IS) and proposed system (TO-BE). And create activity diagrams to illustrate the user journey, such as ticket booking and payment processes.

**Axure**, with its advanced prototyping capabilities, was instrumental in developing interactive wireframes and mockups. By simulating user interactions and incorporating dynamic elements, Axure was used to develop interactive prototypes that bridged the gap between conceptual designs and actual system functionality.

**Balsamiq** provided a simpler, yet highly effective platform for rapid wireframing during the early stages of design. Balsamiq was used during the early design phase to quickly brainstorm and iterate on ideas for the user interface. It was used to design low-fidelity wireframes for core features such as login, event search, and booking pages. Rapid experimentation with layouts and features to ensure usability and simplicity. Provide an easy-to-understand visual representation of the application for stakeholders who may not be technical.

**Document 9 – BA Experience**

**Answer.**

1. Requirement Gathering

During the Requirement Gathering phase, I facilitated workshops and conducted interview to gather requirements. I also used the MoSCoW technique to prioritize and categorize requirements as Must-Have, Should-Have, Could-Have, and Won’t-Have. Given the client’s unavailability at times, I proactively identified alternate points of contact to obtain the necessary information promptly.

* Validated requirements using the FURPS framework (Functionality, Usability, Reliability, Performance, and Supportability).
* Identified and removed duplicate and conflicting requirements to maintain clarity.
* Created prototypes to gather more specific requirements, ensuring stakeholder alignment.

2. Requirement Analysis

In the Requirement Analysis phase, I focused on visually representing requirements for clarity and team understanding:

* Developed UML diagrams, such as use case diagrams and activity diagrams, to illustrate the process flow and requirements.
* Facilitated discussions with the team to explain the diagrams and incorporated feedback to improve their accuracy and relevance.
* Prepared detailed Business Requirements Specification (BRS) and Software Requirements Specification (SRS) documents to formalize the gathered requirements.

3. Design

During the Design phase, I played a key role in connecting the requirements with actionable designs:

* Used use case diagrams to create detailed test cases that covered both positive and negative scenarios.
* Regularly communicated with the client to validate the design and solution documents.
* Ensured no test case was missed, as gaps could have significant implications on project development.
* Prepared test data and updated the Requirements Traceability Matrix (RTM) to ensure all requirements were addressed in the design.

4. Development

In the Development phase, my focus was on bridging communication between stakeholders and the technical team:

* Conducted regular meetings to ensure clarity on requirements.
* Resolved team disagreements by having one-on-one discussions, promoting a collaborative and positive environment.
* Provided clarifications to the tech team by referring to UML diagrams and other documentation.
* Conducted regular meetings with the technical team and client to address progress and concerns, ensuring everyone stayed aligned.
* For those who missed key meetings, I recorded sessions and followed up individually to address their questions.

5. Testing

During the Testing phase, I ensured a high level of accuracy and preparedness:

* Assisted QA in preparing test cases derived from use cases and ensured they were comprehensive.
* Performed high-level testing to identify any gaps or issues early.
* Collaborated with the client to request and finalize test data.
* Updated the RTM to validate that all requirements were thoroughly tested.
* Secured the client’s signoff on testing outcomes and prepared them for User Acceptance Testing (UAT) by organizing reviews and walkthroughs.

6. Deployment

In the Deployment phase, I worked to ensure a smooth handover and client readiness:

* Shared the updated RTM as part of the project closure document to demonstrate requirements coverage.
* Coordinated the preparation and sharing of end-user manuals to ensure easy adoption of the system.
* Planned and organized training sessions for all relevant stakeholders to familiarize them with the system.
* Ensured that all attendees participated in the training sessions by tracking and addressing absences.
  1. In the context of the **Prerna Centre for Excellence** project, managing changing requirements during the SDLC (Waterfall methodology) required a structured and proactive approach. Changes introduced at various phases had the potential to disrupt timelines and project outcomes, so a robust **Change Management Process** was essential.
* Changes during Requirement Gathering or Analysis were easier to accommodate by revising documentation.
* Changes during the Design, Development, or Testing phases involved more effort, often leading to rework. These were minimized by ensuring clear requirements early in the process.
* Ensured the client was informed about any potential delays or additional costs resulting from changes by regularly communicating with the stakeholders about the status and implications of requested changes.