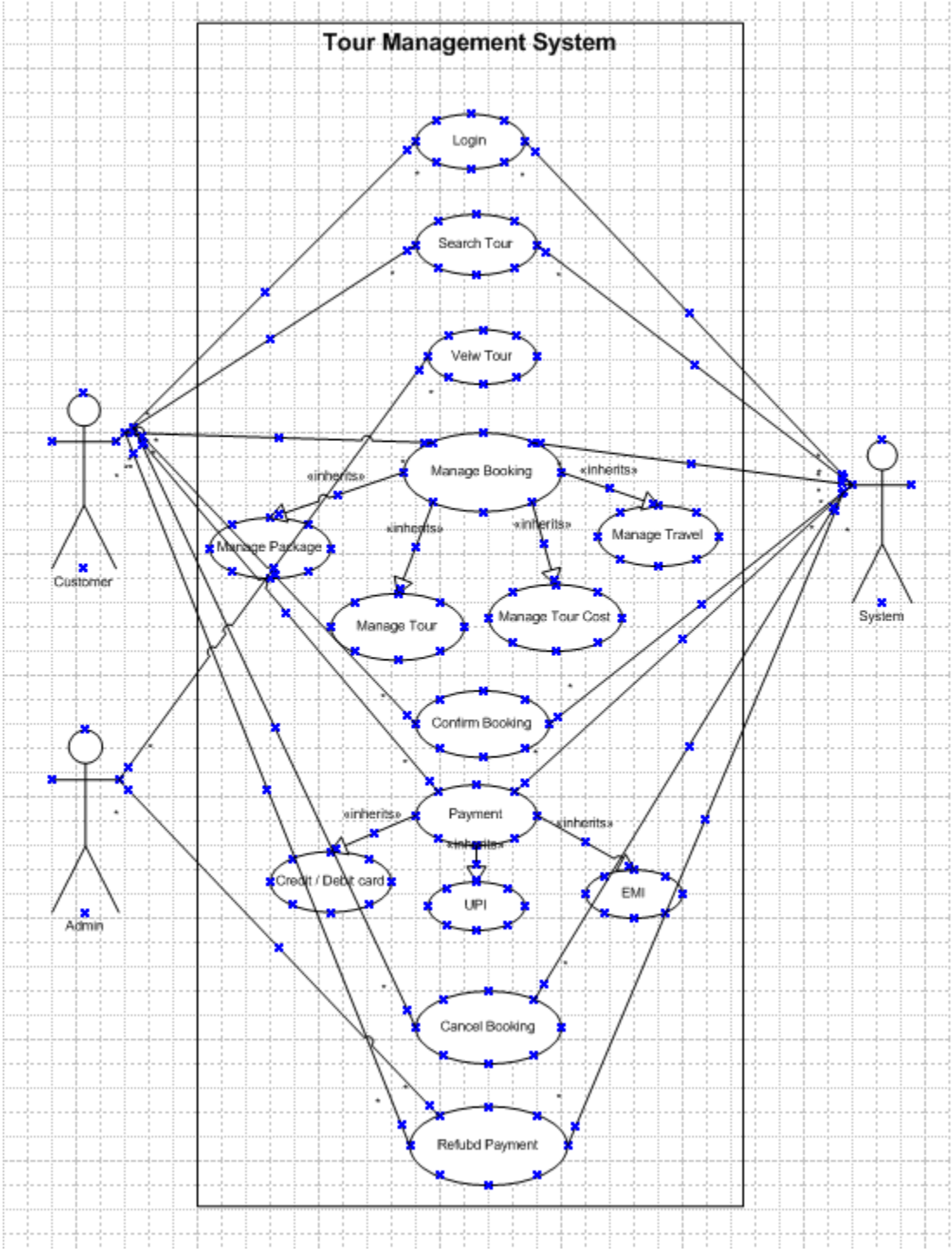
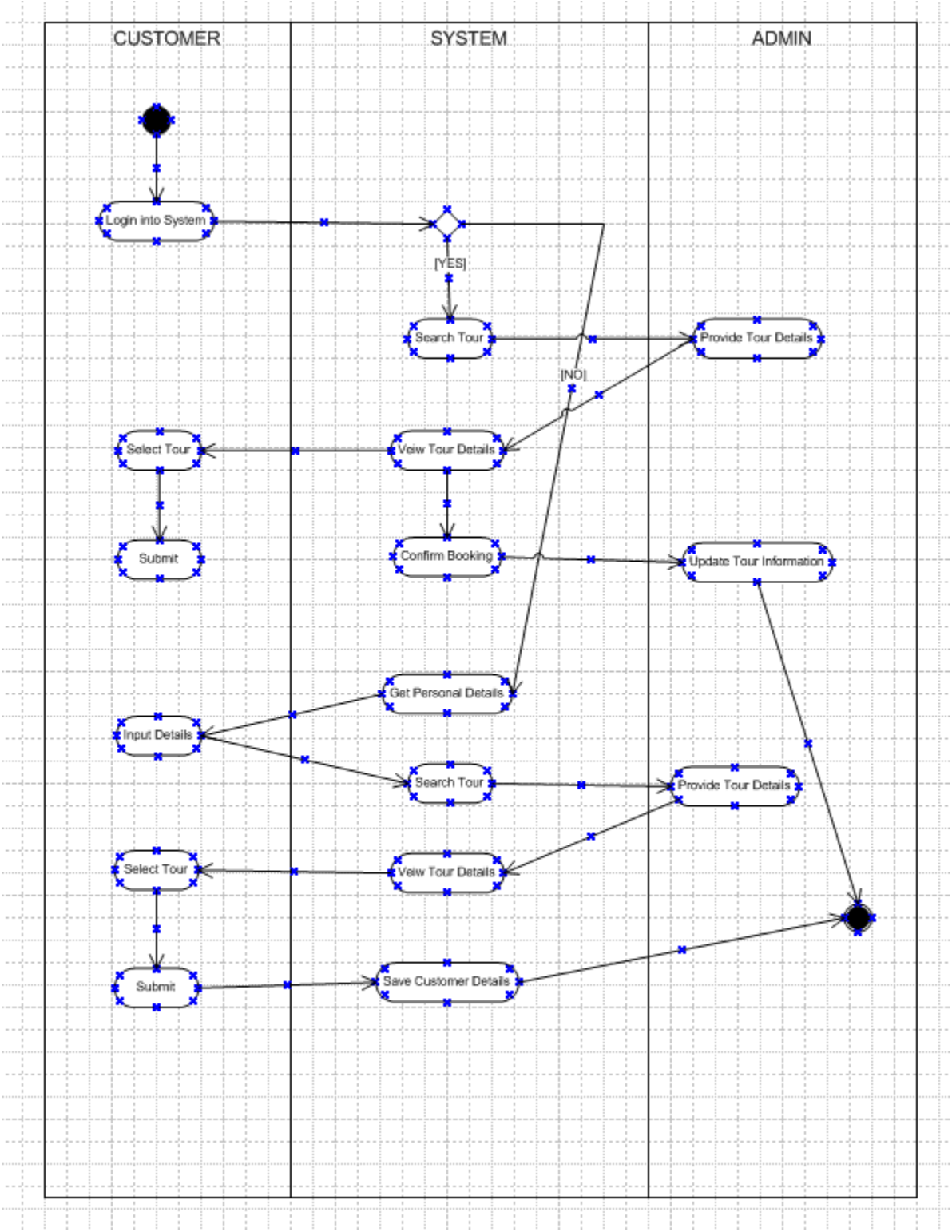
WATERFALL DELIVARABLES
TOUR MANAGEMENT SYSTEM (SMART TOUR)

DOCUMENT 6 : Please prepare a use case diagram, activity diagram and a use case specification document.

USE CASE DIAGRAM



ACTIVITY DIAGRAM



USECASE SPECIFICATIONS

➢ USER LOGIN

1. Use Case Name - User Login

2. Use Case Description

This use case describes the process by which a user securely logs into the tour app using their credentials (email, phone number, or social media login). The system verifies the credentials and grants access to personalized services.

3. Actors

- Primary Actor: User (Customer)
- Secondary Actor: Authentication System, Database

4. Basic Flow (Normal Scenario)

- The user opens the app and navigates to the login screen.
- The system prompts the user to enter credentials (email/phone number & password) or select social login.
- The user inputs the details and submits the login request.
- The system validates the credentials against stored records.
- If valid, the system grants access and redirects the user to the home screen.
- The user can now access personalized features.

5. Alternate Flow

Social Login: Instead of entering credentials, the user selects a social media login (Google/Facebook/Apple). The system redirects to the respective authentication service, verifies credentials, and logs in the user.

6. Exceptional Flows (Error Handling)

- Invalid Credentials: If incorrect details are entered, the system displays an error message and prompts for re-entry.
- Account Locked: If multiple failed attempts occur, the account is temporarily locked, and the user is notified.
- Network Error: If the system fails to connect to the authentication server, an error message appears.

7. Pre-Conditions

- The user must be a registered member with valid credentials.
- Internet connectivity must be available for online authentication.

8. Post-Conditions

- The user is successfully logged in and redirected to the dashboard.
- Failed login attempts are logged for security monitoring.

9. Assumptions

- The user has an active account.
- The system supports multiple authentication methods.

10. Constraints

- Passwords must meet security standards (e.g., minimum 8 characters, special character required).

- The system should not allow more than 5 failed login attempts within a short period.

11. Dependencies

- Integration with authentication APIs (e.g., Google OAuth, Facebook Login).
- A secure database for storing user credentials.

12. Inputs and Outputs

- Inputs: Email/phone number, password, or social login credentials.
- Outputs: Success message, error message, or locked account notification.

13. Business Rules

- Users must verify their email/phone before first login.
- Two-factor authentication (2FA) may be required for enhanced security.
- Expired passwords prompt a reset request.

14. Miscellaneous Information

- Future updates may include biometric login (fingerprint/face ID) for enhanced security.
- A "Remember Me" option allows users to stay logged in for a specified duration.

➢ SEARCH TOUR

1. Use Case Name - Search Tour

2. Use Case Description

This use case describes how a user searches for available tours based on preferences like destination, date, price range, and activities. The system retrieves and displays relevant results from the tour database.

3. Actors

- Primary Actor: User (Traveler)
- Secondary Actors: System, Tour Database, External APIs (if applicable)

4. Basic Flow

- The user opens the app and navigates to the search tour section.
- The user enters search criteria (e.g., destination, travel dates, budget, category).
- The system fetches relevant tour data from the database.
- Matching tours are displayed to the user with filters and sorting options.
- The user selects a tour to view more details.

5. Alternate Flow

- If no criteria are entered, the system displays all available tours.
- If no exact matches are found, the system suggests similar tours.
- If filters are applied, the system updates the displayed results dynamically.

6. Exceptional Flows

- If no tours match the search criteria, a "No results found" message is displayed with suggestions.
- If the tour database is down, the system displays an error message and asks the user to try later.
- If the user enters invalid inputs, an error message prompts them to correct the information.

7. Pre-Conditions

- The system must be connected to a tour database.
- The user must have internet access to retrieve tour results.

8. Post-Conditions

- The user successfully views available tours or receives alternative suggestions.
- Search history may be saved for future reference (if enabled).

9. Assumptions

- The user has basic knowledge of using a mobile or web-based application.
- The database contains up-to-date tour information.

10. Constraints

- System response time should be within 3-5 seconds.
- Search queries should not exceed a predefined threshold (e.g., 100 results).
- The app must support searches in multiple languages (if applicable).

11. Dependencies

- The system depends on the tour database and external APIs for fetching tour data.
- Search results rely on the accuracy and availability of stored data.

12. Inputs and Outputs

- Inputs: Destination, date, budget, activities, filters, and sorting preferences.
- Outputs: List of matching tours, alternative suggestions, or an error message.

13. Business Rules

- Search results should be ranked based on relevance, popularity, and user ratings.
- Special discounts and promotions should be highlighted in the search results.
- Users should be able to save searches for later viewing.

14. Miscellaneous Information

- The app may include AI-based recommendations based on past user behavior.
- Search history may be used to improve future recommendations.

➢ VEIW TOUR

1. Use Case Name:- View Tour

2. Use Case Description:

This use case describes how a user views available tours, including tour details such as itinerary, pricing, availability, and booking options.

3. Actors:

- Primary Actor: Traveler (User)
- Secondary Actor: System Administrator (for updating tour details), Tour Operator

4. Basic Flow:

- User logs into the tour app.
- User navigates to the “View Tours” section.
- The system displays a list of available tours with basic details (name, location, price, date).
- User selects a specific tour to view more details.

- The system retrieves and displays the full tour details (itinerary, images, pricing, availability, etc.).
- User can choose to proceed with booking, save the tour, or exit.

5. Alternate Flow:

- If the user filters the tour list: The system applies filters (e.g., price range, location) and displays results accordingly.
- If the user searches for a specific tour: The system retrieves relevant tours matching the search query.

6. Exceptional Flows:

- If no tours are available: The system displays a message, “No tours available at the moment.”
- If the system fails to retrieve tour details: The system displays an error message and prompts the user to try again later.

7. Pre-Conditions:

- The user must have access to the app.
- The system must have an updated database of available tours.

8. Post-Conditions:

- The user successfully views tour details or receives an appropriate error message.
- The system logs user interactions for analytics.

9. Assumptions:

- Users have a stable internet connection.
- The tour database is regularly updated.

10. Constraints:

- System response time must be quick to avoid user frustration.
- Tour details must be presented in a user-friendly format.

11. Dependencies:

- The system depends on a real-time database of tours.
- API integrations may be needed for live availability updates.

12. Inputs and Outputs:

- Inputs: User search/filter criteria (e.g., location, price range, date).
- Outputs: A list of matching tours with relevant details.

13. Business Rules:

- Only verified and available tours should be displayed.
- Prices should be dynamically updated based on availability and demand.

14. Miscellaneous Information:

- User feedback on tours may be collected to enhance recommendations.
- A caching mechanism may be used to speed up tour retrieval.

➢ CONFIRM BOOKING

1. Use Case Name - Confirm Booking

2. Use Case Description

This use case describes the process of confirming a tour booking in the app after the user selects a tour, provides required details, and makes a payment. The system verifies availability, processes payment, and generates a booking confirmation.

3. Actors

Primary Actor: User (Traveler)

Secondary Actors:

- Payment Gateway
- Hotel/Tour Operator System
- Admin (for manual verification if needed)

4. Basic Flow (Main Success Scenario)

- The user selects a tour package and proceeds to checkout.
- The system displays the booking summary, including price, itinerary, and policies.
- The user enters personal details and selects a payment method.
- The system verifies real-time availability for the selected tour.
- The user confirms the booking and initiates payment.
- The payment gateway processes the transaction and returns a success response.
- The system generates a booking confirmation and sends it via email/SMS.
- The user can view the confirmed booking under "My Trips" in the app**.**

5. Alternate Flow (Variations in Process)

- 5A: If the user applies a promo code, the system validates and applies the discount before payment.
- 5B: If the user selects "Pay Later" (if available), the booking is reserved temporarily with a deadline for payment.

6. Exceptional Flows (Error Handling)

- 6A: If payment fails, the system displays an error message and allows the user to retry or choose another payment method.
- 6B: If the tour is fully booked during the confirmation process, the system notifies the user and suggests alternative dates or tours.
- 6C: If the internet connection is lost during the process, the system saves the progress and allows the user to resume booking later.

7. Pre-Conditions

- The user must be logged into the app.
- The selected tour must be available for booking.
- The user must have a valid payment method.

8. Post-Conditions

- The booking is successfully recorded in the system.
- The user receives confirmation details via email/SMS.
- The system updates tour availability to prevent overbooking.

9. Assumptions

- The payment gateway is operational.
- The tour operator provides real-time availability updates.
- The user has a stable internet connection.

10. Constraints

- Bookings must be completed within a fixed time limit (e.g., 10 minutes) before the session expires.
- Some tours may require manual confirmation before finalizing.

11. Dependencies

- Reliable third-party payment processing.
- Accurate tour inventory updates from vendors.
- Secure data transmission for user details and payment information.

12. Inputs and Outputs

- Inputs: User details, selected tour, payment details, promo codes (if applicable).
- Outputs: Booking confirmation, payment receipt, email/SMS notification.

13. Business Rules

- A booking is not confirmed until payment is successfully processed.
- Refund and cancellation policies apply as per tour operator rules.
- Promo codes must be validated before checkout.

14. Miscellaneous Information

- Future enhancements could include AI-powered recommendations for alternative bookings in case of unavailability.
- Integration with travel insurance services for added security.

➢ CANCEL BOOKING

1. Use Case Name - Cancel Booking

2. Use Case Description

This use case describes the process by which a user cancels an existing booking for a tour, hotel, or transportation. It ensures the booking is successfully canceled, with appropriate notifications, refund processing (if applicable), and updates to the availability database.

3. Actors

Primary Actor: User (Customer)

Secondary Actors:

- Customer Support Representative (for assisted cancellations)
- Payment Gateway System (for refunds)
- Booking System (for availability updates)

4. Basic Flow (Main Success Scenario)

- The user logs into the app and navigates to the "My Bookings" section.
- The user selects the booking they wish to cancel.
- The app displays the cancellation policy and refund details.
- The user confirms the cancellation request.
- The system processes the request and updates the booking status to "Canceled."

- If applicable, the system initiates a refund via the payment gateway.
- The user receives a confirmation notification via email and/or SMS.

5. Alternate Flow (Variations in the Main Process)

- 5A: If the user is eligible for a partial refund, the system calculates and displays the refundable amount before confirmation.
- 5B: If cancellation is requested after the refund window has passed, the system informs the user that no refund is available but allows cancellation.

6. Exceptional Flows (Error Scenarios)

- 6A: The booking cannot be canceled (e.g., non-refundable booking). The system notifies the user and suggests contacting customer support.
- 6B: Payment gateway failure during refund processing. The system retries or escalates the issue to support.
- 6C: Internet connectivity issues prevent cancellation. The system prompts the user to retry later.

7. Pre-Conditions (Requirements Before Execution)

- The user must have an active booking in the system.
- The booking must be eligible for cancellation (as per cancellation policy).
- The user must be logged into the app.

8. Post-Conditions (Results After Execution)

- The booking status is updated as "Canceled."
- If applicable, the refund is processed.
- Notifications are sent to the user and relevant service providers.

9. Assumptions

- Users have read and understood the cancellation policies before booking.
- The system has real-time integration with the payment gateway.
- The user has a stable internet connection.

10. Constraints

- Cancellation policies vary based on service providers (hotels, airlines, etc.).
- Refunds may take a few days to process, depending on the payment gateway.
- Some bookings may be non-refundable**.**

11. Dependencies

- The booking system must update the availability database.
- Payment gateway must support automated refunds.
- Notification services (SMS, email) must be functional.

12. Inputs and Outputs

Inputs:

- Booking ID
- User confirmation of cancellation
- Refund eligibility status

Outputs:

- Booking cancellation confirmation
- Updated booking status
- Refund transaction ID (if applicable)

- Cancellation confirmation message (email/SMS)

13. Business Rules

- Cancellations must comply with the service provider's refund policy.
- Refunds are processed only if cancellation is within the allowed time frame.
- The system should record all cancellation requests for future audits.

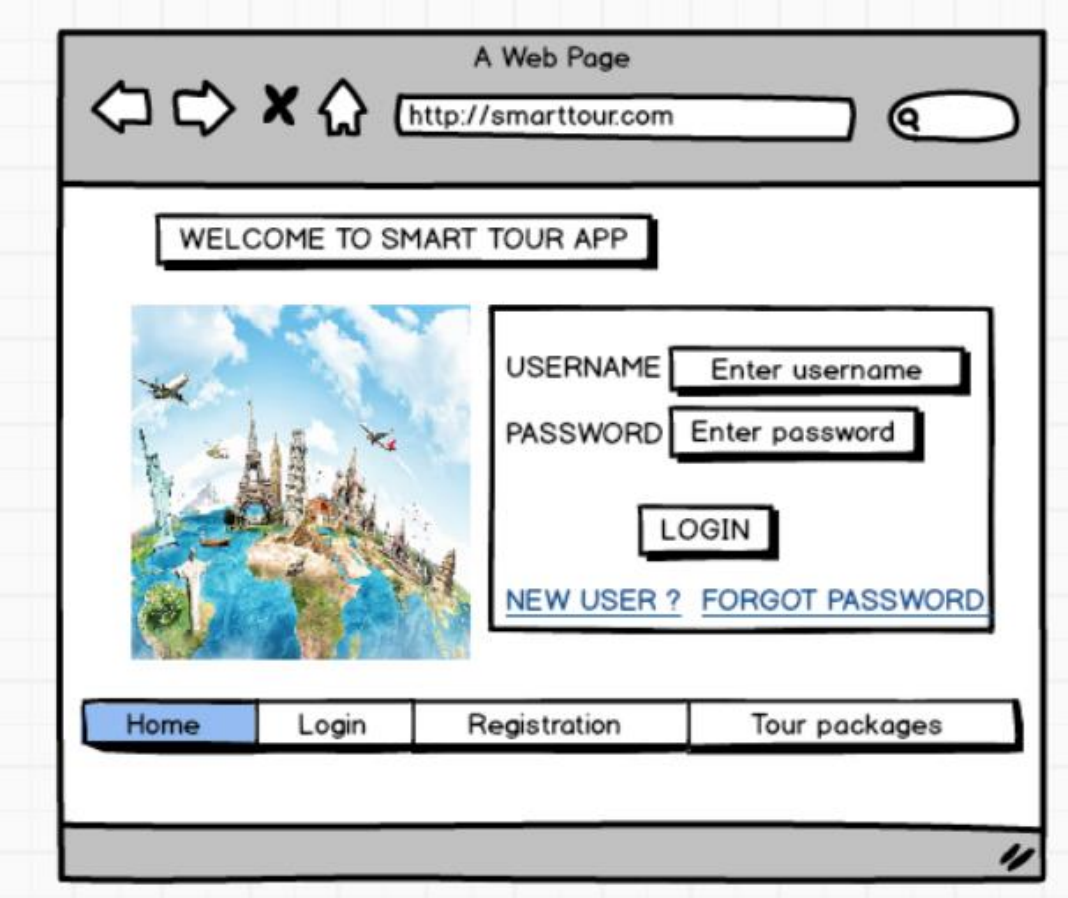
14. Miscellaneous Information

- The system should provide an option for users to give feedback on the reason for cancellation.
- If a booking is canceled due to system errors or double bookings, the company may offer compensation.

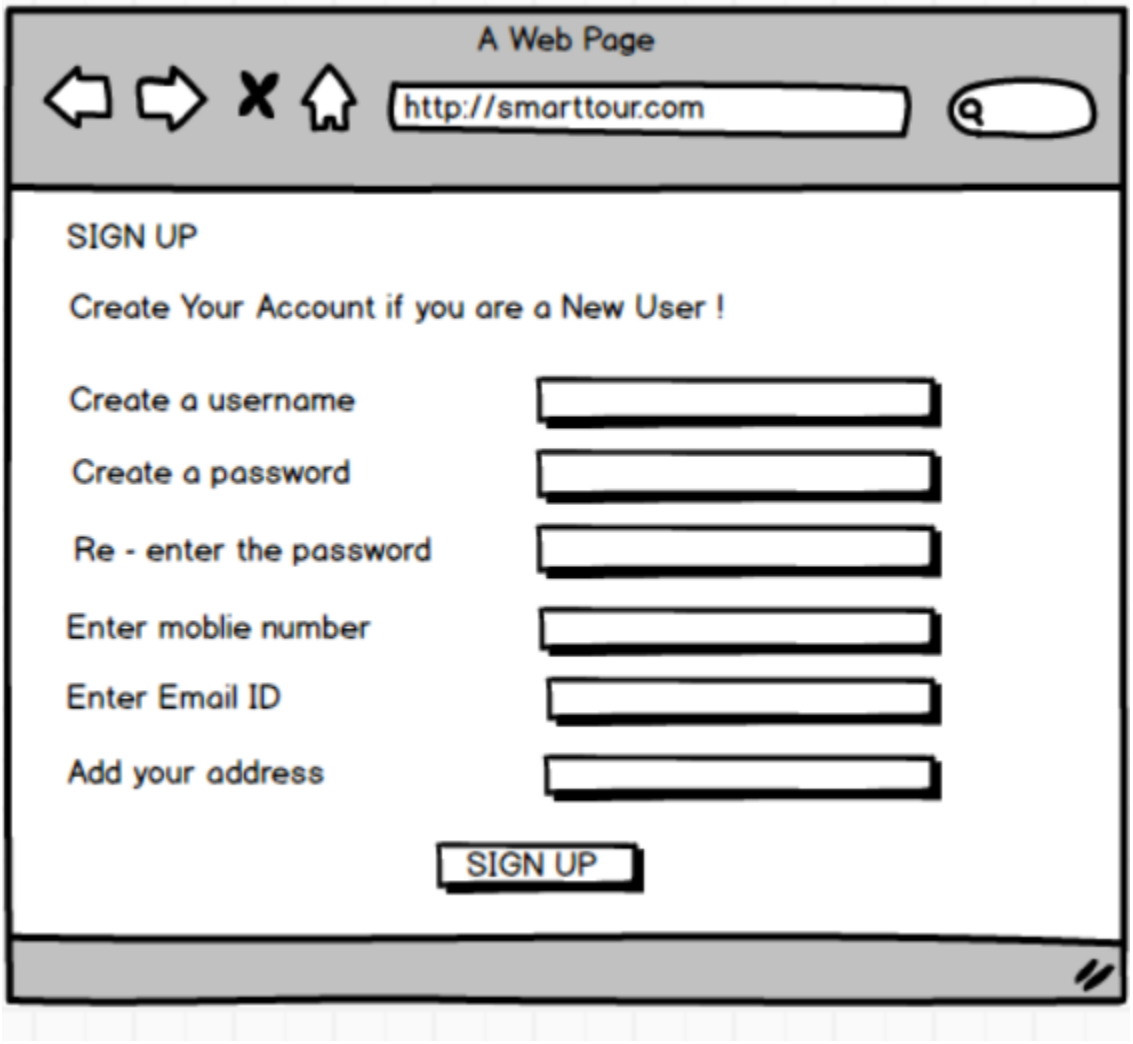
DOCUMENT 7 – SCREENS & PAGES

1. HOME PAGE

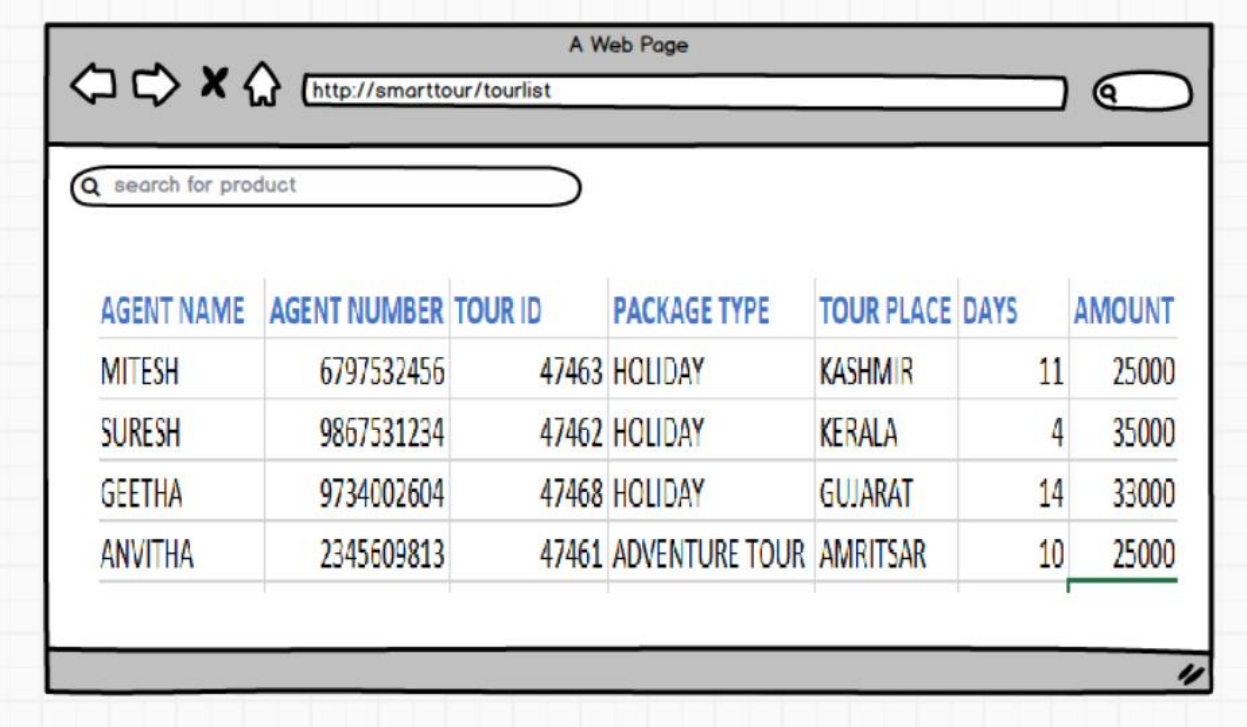
2 . LOGIN PAGE


A Web Page
http://smarttour.com
WELCOME TO SMART TOUR APP
USERNAME
Enter username
PASSWORD
Enter password
LOGIN
NEW USER ?
FORGOT PASSWORD
Home
Login
Registration
Tour packages

3.REGISTRATION

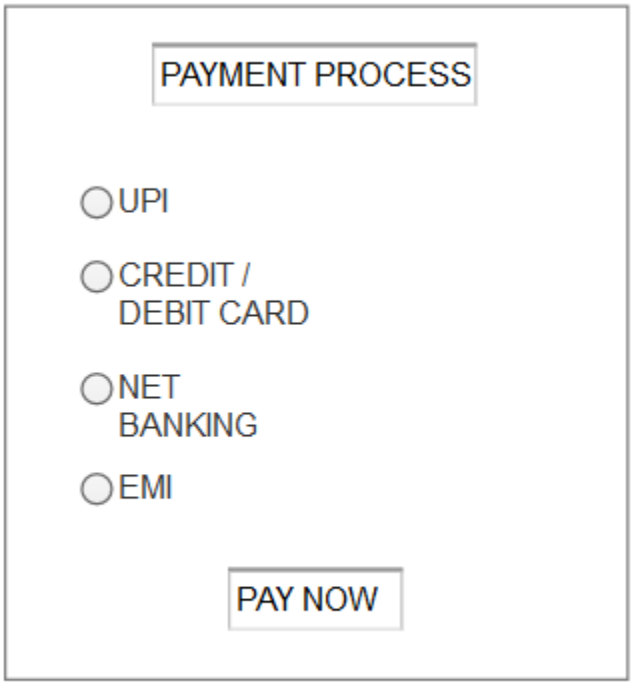


4. TOUR PACKAGES

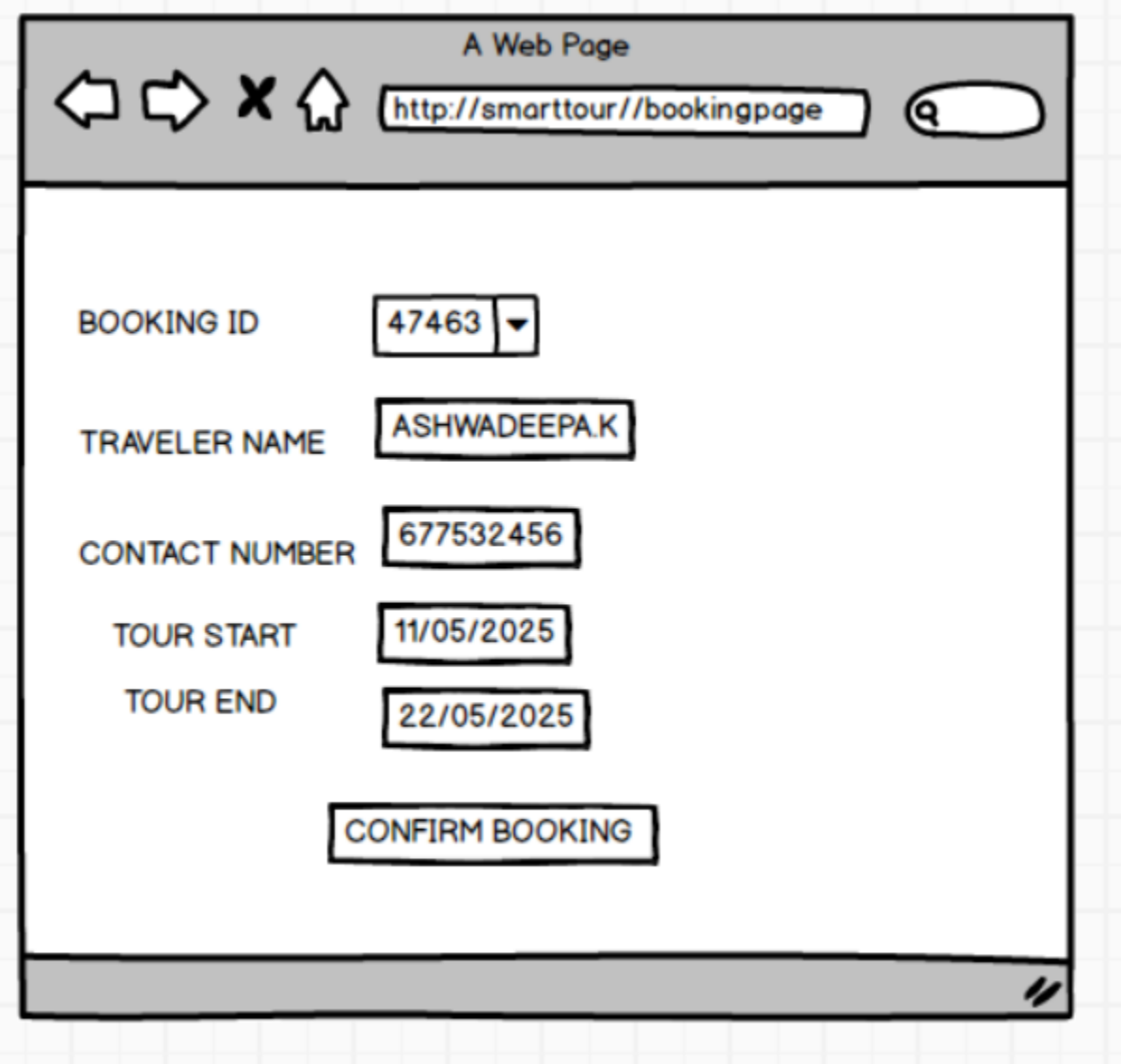


AGENT NAME	AGENT NUMBER	TOUR ID	PACKAGE TYPE	TOUR PLACE	DAYS	AMOUNT
MITESH	6797532456	47463	HOLIDAY	KASHMIR	11	25000
SURESH	9867531234	47462	HOLIDAY	KERALA	4	35000
GEETHA	9734002604	47468	HOLIDAY	GUJARAT	14	33000
ANVITHA	2345609813	47461	ADVENTURE TOUR	AMRITSAR	10	25000

5.PAYMENT PAGE



6. CONFIRM BOOKING



DOCUMENT 8 – TOOLS VISIO AND AXURE

Using MS Visio:
MS Visio was instrumental in creating flowcharts, use case diagrams, and system architecture for the Tour App's Cancel Booking feature. The tool's drag-and-drop interface made it easy to map out processes, such as booking cancellation flows, refund processing, and exception handling. The ability to use predefined stencils and templates helped streamline the development of entity-relationship diagrams (ERD) for database design. Collaboration features in Visio Online allowed team members to review and refine diagrams efficiently.

Using Axure:
Axure was used for prototyping and UI/UX design, enabling the creation of interactive wireframes and mockups for the booking cancellation interface. With dynamic panels and conditional logic, we simulated real-time responses like refund eligibility checks and error handling (e.g., non-refundable bookings). The ability to create clickable prototypes helped in gathering early feedback from stakeholders before moving to development. Axure's integration with Figma and Sketch also ensured seamless collaboration with designers and developers.

DOCUMENT 9 – BA EXPERIENCE

1. Requirement Gathering

- Conducted stakeholder interviews with travel agencies, customers, and service providers to understand pain points.
- Collected data on current booking and cancellation challenges through surveys and market research.
- Created a Business Requirement Document (BRD) outlining core functionalities like booking, cancellation, and refund processing.
- Defined scope, goals, and success criteria with stakeholders to align business needs with the app's features.

2. Requirement Analysis

- Analyzed collected requirements and identified functional & non-functional requirements for the app.
- Worked with SMEs (Subject Matter Experts) to define cancellation policies, refund logic, and dependency on third-party APIs (hotels, airlines, etc.).
- Created use case diagrams and process flows in MS Visio to visualize workflows like cancellation requests.
- Collaborated with the development team to finalize the Software Requirement Specification (SRS) document.

3. Design

- Assisted UX/UI designers by providing user journey insights for key features like booking and cancellations.
- Created wireframes and prototypes in Axure to visualize user interactions and process flows.

- Reviewed data flow diagrams (DFDs) to ensure smooth data exchanges between the app, payment gateways, and external booking providers.
- Facilitated design approval meetings with stakeholders to confirm alignment with business needs.

4. Development

- Acted as a bridge between business and technical teams, ensuring developers understood business logic.
- Conducted regular requirement clarification sessions to address doubts and refine features.
- Assisted in creating API documentation for third-party integrations (e.g., payment gateways, airline booking systems).
- Reviewed early-stage UI builds and functionality to ensure alignment with business goals.

5. Testing

- Worked with QA teams to define test scenarios and acceptance criteria based on requirements.
- Conducted UAT (User Acceptance Testing), simulating real-world scenarios like cancellations, refunds, and booking modifications.
- Identified and documented bugs, inconsistencies, and missing features, ensuring they were addressed before deployment.
- Ensured that the app met regulatory compliance related to data security, refund policies, and customer rights.

6. Deployment

- Assisted in preparing user guides and FAQs for customer support teams.
- Conducted training sessions for stakeholders on how to use the app effectively.
- Monitored post-launch feedback, collecting customer reviews and analytics to suggest improvements for future updates.
- Facilitated post-deployment meetings to assess system performance and ensure business goals were met.