A Crm Application To Manage Facilities Provided By An Institution

Table of Contents

    - Types of Testing (Unit, Integration, UAT, Performance, Security)
    - Testing Tools (Salesforce Developer Console, Sandbox)
    - Bug Tracking and Resolution
    - Quality Assurance Metrics
9. **Deployment and Maintenance**
    - Deployment Strategy
    - Post-Deployment Testing
    - Maintenance and Updates
    - User Support and Feedback Loop
    - Backup and Disaster Recovery
10. **Future Enhancements**
    - Scalability and Performance Enhancements
    - Advanced Analytics and Reporting
    - Mobile App Integration
    - Integration with Other Systems
    - Enhancing User Experience
    - Internationalization and Localization

1. Project Overview

Objective: This section introduces the CRM system, sets expectations, and clarifies the purpose of the project. It helps any reader (e.g., stakeholders, professors, or colleagues) understand what the project intends to achieve.

Detailed Outline:

- **Project Title**:
    - **CRM Application for Managing Institutional Facilities** — A descriptive title that specifies this CRM is built to manage facilities and other resources within an institution.
- **Purpose**:
    - Explain the rationale for the CRM system. Here, it serves as a centralized system to manage resources (facilities) offered by an institution and streamline the interactions and management around these resources.
    - **Example**: "This CRM system is designed to streamline resource allocation within the institution, provide an intuitive user interface for bookings, enable feedback, and generate insights on resource usage. It aims to reduce administrative overhead and improve user satisfaction."
- **Platform**:

    - **Salesforce**: Mention Salesforce as the chosen platform because of its CRM capabilities, customizable features, and pre-built integrations.
    - Describe briefly why Salesforce was chosen, e.g., for its scalability, integration options, and security.
- **Scope**:
    - Define the scope of the project by specifying key features and boundaries.
    - **Example**: "The scope includes tracking and managing facilities like study rooms, sports areas, and lab spaces. The CRM will support the booking process, feedback collection, and reporting, but it will not cover physical infrastructure maintenance or inventory management."

2. Objectives

This section breaks down the primary goals of the CRM system. It clarifies what the project aims to accomplish and outlines specific, measurable outcomes to guide development.

Detailed Outline:

- **Track and Manage All Facilities**:
    - The CRM should maintain an updated catalog of all facilities (e.g., classrooms, labs, sports areas) with details like capacity, location, availability, and condition.
    - Goal: Administrators and staff should be able to view, update, and manage facility records in real time.
- **Allow Administrators to View and Analyze Facility Data**:
    - Provide administrators with tools to monitor how often facilities are used, the types of bookings being made, and peak usage times.
    - Goal: Generate reports and visual dashboards that enable easy analysis of usage data to make data-driven decisions (e.g., scheduling maintenance or upgrading popular facilities).
- **Enable Stakeholders to Request and Provide Feedback on Facilities**:
    - Enable students, faculty, and other stakeholders to request facilities through a booking system and to leave feedback after usage.
    - Goal: Users should find it easy to browse available facilities, request booking time slots, and share comments on their experience.
- **Streamline Communication for Facility Availability and Maintenance**:
    - Automate notifications for users about their booking status, reminders for upcoming bookings, or maintenance updates.
    - Goal: Improve the user experience by keeping all parties informed and minimizing last-minute booking conflicts.
- **Facilitate Improved Service Delivery**:

        - By centralizing feedback and booking data, administrators should be able to quickly identify facilities that require attention or additional resources.
        - Goal: Create a seamless experience that empowers the institution to proactively manage and enhance facility quality.

3. System Architecture

This section details the architecture of the CRM system, covering the data model, relationships, and how the components interact. This is crucial to ensure a scalable, well-organized structure for the project.

Detailed Outline:

- **Data Model**:
    - **Overview**: Define the primary entities and their attributes that will form the foundation of the CRM's data structure. This typically includes key objects like `User`, `Facility`, `Booking`, `Feedback`, and `Notification`.
    - Example:
        - **User**: Represents students, staff, and administrators.
        - **Facility**: Represents a resource provided by the institution, such as a study room, lab, or sports area.
        - **Booking**: Records each user's request to reserve a facility.
        - **Feedback**: Stores user reviews or comments about their experience with a facility.
        - **Notification**: Manages communications to users regarding booking confirmations, updates, or feedback responses.
- **Salesforce Objects**:
    - **Custom Objects**: Outline the custom objects that are not part of Salesforce's standard objects but are essential for this CRM system.
        - **Examples**:
            - `Facility`: Custom object that includes fields like `Facility Name`, `Type`, `Capacity`, `Location`, and `Status`.
            - `Booking`: Custom object with fields such as `User`, `Facility`, `Booking Date`, `Start Time`, `End Time`, and `Status` (e.g., Pending, Approved, Rejected).
            - `Feedback`: Custom object that holds `User`, `Facility`, `Rating`, `Comment`, and `Date`.
    - **Standard Objects**: Mention any Salesforce standard objects that will be utilized, such as `User` for account management or `Contact` to manage user information.

- **Relationships**:
    - **User to Booking (One-to-Many)**: A single user can have multiple bookings. This relationship helps to identify each user's reservation history.
    - **Facility to Booking (One-to-Many)**: Each facility can be booked multiple times, but each booking is tied to a single facility.
    - **User to Feedback (One-to-Many)**: A user can leave feedback multiple times, each associated with a specific facility booking.
    - **Facility to Feedback (One-to-Many)**: Each facility can have multiple feedback entries from different users, enabling feedback collection.
- **Data Flow and Component Interaction**:
    - Describe how data flows through the system. For instance:
        - When a user submits a booking request, the `Booking` object is updated.
        - After approval, a `Notification` is sent to the user.
        - Following usage, the user may submit `Feedback` that links back to both the `User` and `Facility`.

4. Features and Functionalities

This section provides an in-depth look at the core features of the CRM application, detailing the primary functionalities users will interact with. Each feature description should cover how the feature works and the business logic involved.

Detailed Outline:

- **User Management**:
    - **Roles**:
        - Define user roles such as Administrator, Staff, and Student.
        - **Administrator**: Full access to all functionalities, including facility management, booking approvals, and report generation.
        - **Staff**: May have permissions to manage facilities and view bookings related to their department.
        - **Student**: Limited access, primarily focused on booking facilities, viewing booking history, and providing feedback.
    - **Access Control**:
        - Describe permissions for each role to ensure data security.
        - Example: Only Administrators can approve bookings, while Students can only view available facilities and submit booking requests.
- **Facility Management**:
    - **Facility Information**:

    - Include details such as facility name, type (e.g., lab, gym), location, capacity, and availability status.
    - Each facility record should also have a maintenance status (e.g., Available, Under Maintenance) to inform users of its current condition.
  - **Managing Facility Data**:
    - Administrators or Staff can add, update, or archive facilities.
    - Example: When a new study room is created, it should appear in the available facility list for booking.
- **Booking and Reservation**:
  - **Booking Process**:
    - Step-by-step guide on how users submit booking requests.
      - Example: A Student selects a facility, chooses a date and time, and submits a booking request.
    - The system should automatically check facility availability and display a confirmation if available.
  - **Approval Workflow**:
    - For facilities requiring approval, administrators receive notifications for pending requests.
    - Approval or rejection updates the booking status and sends a notification to the user.
  - **Booking Constraints**:
    - Set constraints, such as maximum booking duration, booking time windows, or restrictions on consecutive bookings.
    - Example: A student may only book a study room for a maximum of 2 hours per day.
- **Feedback Collection**:
  - **Feedback Submission**:
    - After using a facility, users should be able to submit feedback that includes a rating (e.g., 1 to 5 stars) and comments.
    - Users should see their feedback history for reference.
  - **Feedback Management**:
    - Administrators can view and analyze feedback to make improvements or address issues.
    - Example: If feedback consistently highlights cleanliness issues, administrators can schedule additional cleaning.
- **Reporting and Analytics**:
  - **Usage Reports**:
    - Administrators can generate reports on facility usage, bookings, peak usage times, and user satisfaction ratings.
    - Visual dashboards, like bar charts and pie charts, help to quickly identify trends.
  - **Data Insights**:

        - Provide insights on resource allocation, enabling the institution to optimize facility availability based on demand.
        - Example: If a particular lab is frequently overbooked, the institution might consider adding similar resources.

5. Technical Specifications

This section details the technical aspects of the CRM, including configurations, custom fields, validation rules, automation, and any custom code. This provides an understanding of how the system functions under the hood.

Detailed Outline:

- **Custom Fields and Validation Rules**:
    - **Custom Fields**: List the custom fields created for each object to store specific data.
        - **Example for `Facility` Object**:
            - `Facility Type`: Dropdown with options like Classroom, Lab, Gym.
            - `Capacity`: Integer field for the maximum number of users allowed.
            - `Status`: Picklist with values like Available, Under Maintenance, Closed.
        - **Example for `Booking` Object**:
            - `Booking Date`: Date field for when the booking is made.
            - `Start Time` and `End Time`: Time fields to specify the booking duration.
            - `Approval Status`: Picklist with values Pending, Approved, Rejected.
    - **Validation Rules**:
        - Ensure data integrity and consistency.
        - Examples:
            - `Facility Capacity` cannot be zero or negative.
            - `Booking End Time` must be after `Start Time`.
            - Bookings can only be made for facilities with `Status = Available`.
- **Automations and Workflows**:
    - **Workflows**: Describe automated processes that trigger actions based on criteria.
        - **Examples**:

      - Send a notification to the administrator when a booking request is submitted.
      - Automatically change the `Booking Status` to `Expired` if the booking time has passed and no action was taken.
  - **Triggers**: Use Apex triggers if custom logic is needed beyond standard workflows.
    - **Example**: Trigger to validate if a user has exceeded their booking quota for the month before allowing a new booking.
  - **Process Builder and Flow**:
    - Use Salesforce's Process Builder or Flow to create complex logic without code.
    - Example: Build a Flow to guide users through the booking process step-by-step, checking availability and prompting for feedback submission after a booking ends.
- **Apex Code and Visualforce Pages (if applicable)**:
  - **Apex Classes**: Describe any custom logic written in Apex, such as classes that manage the booking approval process or batch jobs to clean up old bookings.
    - **Example**: A custom Apex class to send reminders for upcoming bookings.
  - **Visualforce Pages / Lightning Components**:
    - If you've built custom interfaces, describe them here.
    - Example: A custom Lightning component for displaying feedback statistics or managing booking requests.
- **API Integrations (if applicable)**:
  - **External Integrations**: Mention any external services that the CRM connects to.
    - **Examples**:
      - Email service API to send booking confirmations.
      - SMS API for real-time notifications if required by the institution.
  - **Security and Access**:
    - Detail authentication methods for accessing external APIs to ensure secure data exchange.
    - Example: Use OAuth for secure connections to external services like a calendar API or notification system.

6. User Guide

The User Guide serves as a step-by-step manual to help users understand how to interact with the CRM. It includes clear instructions, covering login, navigation, and the main features of the system.

Detailed Outline:

- **Login and Navigation**:
    - **Logging In**:
        - Instructions on how to access the Salesforce CRM application.
        - Example: "Open the Salesforce login page, enter your credentials (username and password), and select the CRM app from the main dashboard."
    - **Navigating the Interface**:
        - Describe the main sections of the interface and what users can expect to find in each section.
        - **Examples**:
            - **Home Page**: Displays quick links to recent bookings, upcoming bookings, and notifications.
            - **Facility Management**: Accessible only to Admins and Staff, this section shows a list of facilities with options to add, update, or archive.
            - **My Bookings**: Shows current, past, and upcoming bookings for the logged-in user.
- **Booking a Facility**:
    - **Step-by-Step Booking Process**:
        - A detailed guide for users on how to make a reservation.
        - **Example**:
            - Step 1: Go to the **Facility Booking** page and select a facility.
            - Step 2: Choose a date, start time, and end time.
            - Step 3: Click “Check Availability.” If available, proceed to submit the request.
            - Step 4: After submission, a notification will confirm the booking or inform the user if admin approval is needed.
    - **Booking Status**:
        - Explain the different booking statuses (e.g., Pending, Approved, Rejected, Completed) and what they mean for the user.
        - Example: "A Pending status means the booking is waiting for administrator approval, while Approved indicates it’s ready for use."
- **Providing Feedback**:
    - **Feedback Submission**:
        - Step-by-step guide on how users can submit feedback on a facility they used.
        - Example:
            - Step 1: Go to the **My Bookings** section and select a completed booking.
            - Step 2: Click on “Submit Feedback.”

      - Step 3: Rate the facility on a scale (e.g., 1-5 stars) and add any comments.
      - Step 4: Submit to complete feedback.
  - **Viewing Feedback History**:
    - Users can view their past feedback submissions for reference, typically from a "My Feedback" section.
- **Generating Reports**:
  - **Accessing Reports and Dashboards**:
    - Provide instructions for administrators on accessing and generating reports to analyze facility usage and feedback trends.
    - Example: "Go to the Reports tab, select the pre-built reports, or use the dashboard to view visual summaries. Reports on bookings, user feedback, and usage trends are available."
  - **Customizing Reports**:
    - Brief instructions on how to create custom reports if necessary, for advanced users.
    - Example: "To create a custom report, select 'New Report', choose the appropriate data source (e.g., Facility, Booking), and define the filters."

7. Security and Compliance

This section addresses the security measures and compliance standards in place to protect user data and ensure the CRM system adheres to regulatory requirements. It’s crucial for any system managing sensitive information.

Detailed Outline:

- **User Authentication and Authorization**:
  - **Authentication**:
    - Explain the authentication process used to secure user access, typically Salesforce’s built-in login.
    - Example: "Users must log in using their unique credentials. For enhanced security, multi-factor authentication (MFA) can be enabled, requiring a second verification step such as a code sent to their mobile device."
  - **Authorization and Access Control**:
    - Role-based access controls restrict access to different system areas based on user roles (e.g., Administrator, Staff, Student).

        - Example: "Only Administrators have access to manage facilities and view reports, while Students have limited access to booking and feedback functionalities."

- **Data Security**:
    - **Encryption**:
        - Data is encrypted both at rest and in transit to protect against unauthorized access.
        - Example: "Salesforce ensures data is encrypted during transmission (HTTPS) and at rest using AES encryption."
    - **Field-Level Security and Data Masking**:
        - Sensitive information like user contact details or booking history can be masked or restricted based on user roles.
        - Example: "Only administrators can view full user information, while students only see limited details related to their bookings."
- **Data Privacy and Compliance**:
    - **Data Protection Regulations**:
        - Describe how the CRM complies with data protection regulations (e.g., GDPR, CCPA) relevant to the institution.
        - Example: "The system adheres to GDPR by allowing users to access, update, or delete their personal data upon request. Data retention policies ensure that user data is stored only for as long as necessary."
    - **Audit Logging**:
        - Maintain logs of user actions, such as booking submissions, feedback, or data modifications, to monitor system activity.
        - Example: "Audit logs capture key user actions for tracking and compliance, enabling administrators to review logs if there are any security concerns."
- **Backup and Recovery**:
    - **Regular Backups**:
        - Ensure data is backed up regularly to prevent data loss in the event of a failure.
        - Example: "The system performs nightly backups of all CRM data, allowing for recovery to a prior state if necessary."
    - **Disaster Recovery Plan**:
        - Outline a recovery plan in case of unexpected outages or data loss.
        - Example: "Salesforce provides disaster recovery support, including data replication across multiple data centers to ensure availability and swift recovery from outages."
- **User Education and Awareness**:
    - **Security Training**:
        - Users are educated on best practices for maintaining security, such as recognizing phishing attempts and creating strong passwords.

        - Example: "New users are provided with a guide on CRM security best practices, including avoiding sharing passwords and recognizing suspicious activity."

8. Testing and Quality Assurance

This section outlines the testing and quality assurance processes for the CRM system, ensuring it performs as expected and meets all functional and non-functional requirements.

Detailed Outline:

- **Types of Testing**:
    - **Unit Testing**:
        - Each component or module is tested individually to ensure it performs as expected.
        - Example: "Apex classes and triggers are unit tested to verify that custom business logic, like booking validation, functions correctly."
    - **Integration Testing**:
        - Ensures that different components work together seamlessly. This includes testing workflows, data flow between objects, and external integrations if any.
        - Example: "Integration testing verifies that a booking request updates the status across the Facility and Booking objects and that notifications are sent."
    - **User Acceptance Testing (UAT)**:
        - Real users, such as students and administrators, test the system to confirm that it meets their needs and functions intuitively.
        - Example: "During UAT, users go through end-to-end booking, feedback, and reporting processes, providing feedback to identify any user experience issues."
    - **Performance Testing**:
        - Evaluates the CRM's performance under different load conditions to ensure it can handle expected usage levels.
        - Example: "Testing simulates high booking requests to measure system response time, ensuring it can handle peak usage without lagging."
    - **Security Testing**:
        - Confirms that the CRM is secure and that sensitive data is protected against unauthorized access.
        - Example: "Penetration testing assesses system vulnerabilities, ensuring roles and permissions restrict data access as expected."
- **Testing Tools**:
    - **Salesforce Developer Console**:

        - Used for running and debugging Apex code tests.
        - Example: "Custom logic is tested through Salesforce’s Developer Console, with automated tests verifying that business rules are applied."
    - **Salesforce Test Environment (Sandbox)**:
        - A safe, isolated environment for comprehensive testing before deploying changes to production.
        - Example: "A Salesforce Sandbox is used to conduct UAT and integration tests, allowing real users to interact with the CRM without impacting live data."
- **Bug Tracking and Resolution**:
    - **Issue Tracking**:
        - Issues and bugs discovered during testing are logged in a tracking system, with details on reproduction steps, priority, and assignment.
        - Example: "All bugs identified during UAT are documented in an issue tracker, prioritized based on severity, and assigned to developers for resolution."
    - **Defect Resolution and Retesting**:
        - Developers address logged issues, and retesting is conducted to ensure fixes are effective.
        - Example: "Once bugs are resolved, retesting verifies the fixes, ensuring no other components are affected."
- **Quality Assurance Metrics**:
    - **Code Coverage**:
        - Salesforce requires 75% code coverage for Apex code to be deployed, ensuring thorough testing.
        - Example: "All Apex triggers and classes must meet the minimum coverage threshold, ensuring business logic is thoroughly tested."
    - **User Satisfaction**:
        - Metrics from UAT and feedback scores determine how well the system meets user needs.
        - Example: "User feedback from UAT is analyzed to make usability improvements, increasing satisfaction before final deployment."

9. Deployment and Maintenance

This section outlines the deployment strategy for the CRM system and describes the ongoing maintenance required to keep the system functional, updated, and responsive to user needs.

Detailed Outline:

- **Deployment Strategy**:
    - **Deployment Environment**:
        - Identify the Salesforce environment where the CRM will be deployed (typically, production for live use and sandbox for testing).
        - Example: "The production environment is used for the live system accessible to users, while a sandbox environment is maintained for testing new features and updates."
    - **Version Control and Change Management**:
        - Track code changes and configuration updates using a version control system.
        - Example: "All Apex code and configuration changes are stored in version control, ensuring a clear history and enabling rollback if needed."
    - **Deployment Tools**:
        - **Change Sets**: Salesforce Change Sets allow easy migration of customizations from sandbox to production.
        - **Metadata API**: Used for larger or more complex deployments where more detailed control is needed.
        - Example: "Salesforce Change Sets are used for regular updates, while Metadata API handles bulk deployments and complex changes."
    - **Deployment Process**:
        - Describe the step-by-step deployment process, including testing and approval steps.
        - Example:
            1. Test new features in the sandbox environment.
            2. Perform final approval in UAT.
            3. Deploy changes to production using Change Sets or Metadata API.
            4. Conduct post-deployment testing to ensure all features work as expected.
- **Post-Deployment Testing**:
    - Verify that the system functions correctly after deployment, especially new or modified features.
    - Example: "Post-deployment testing includes checking workflows, validating data consistency, and ensuring access permissions align with security standards."
- **Maintenance and Updates**:
    - **Routine Maintenance**:
        - Schedule periodic maintenance tasks such as data cleanup, reviewing audit logs, and monitoring performance.
        - Example: "Routine tasks include monthly data cleanup to archive old booking data and weekly log reviews to monitor user activity."
    - **System Updates**:
        - Salesforce regularly releases updates. Outline a strategy to keep the CRM compatible with these updates.

        - Example: "Each Salesforce release is reviewed to ensure compatibility. Changes affecting custom components are tested in the sandbox before updating production."
- **User Support and Feedback Loop**:
    - **User Training**:
        - Regularly provide training for users, especially after significant updates or feature additions.
        - Example: "Admin users are trained on new features and processes, ensuring they can help answer common questions from other users."
    - **Feedback Collection and Issue Resolution**:
        - Collect user feedback to continuously improve the CRM. This feedback informs future updates and prioritization of bug fixes.
        - Example: "A feedback form within the CRM allows users to report issues or request new features. Feedback is reviewed monthly to prioritize updates."
- **Backup and Disaster Recovery**:
    - **Data Backup**:
        - Ensure data is backed up regularly to safeguard against data loss.
        - Example: "Daily backups of CRM data ensure recent information can be restored in case of failure."
    - **Disaster Recovery Procedures**:
        - Define steps for recovering from outages or data loss.
        - Example: "Salesforce provides multi-site data centers for redundancy, enabling rapid recovery in the event of data center failure."

10. Future Enhancements

This section discusses potential improvements and features that could be added to the CRM system in the future, based on user feedback, evolving business needs, and technological advancements.

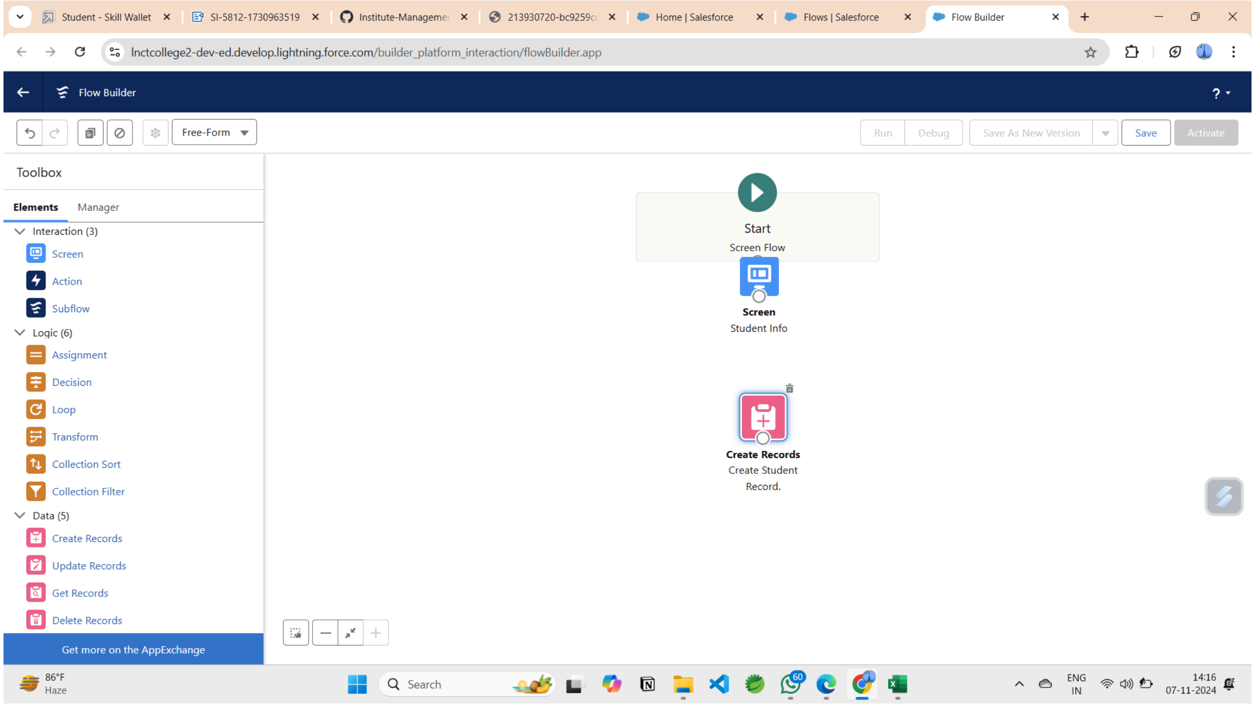
Detailed Outline:

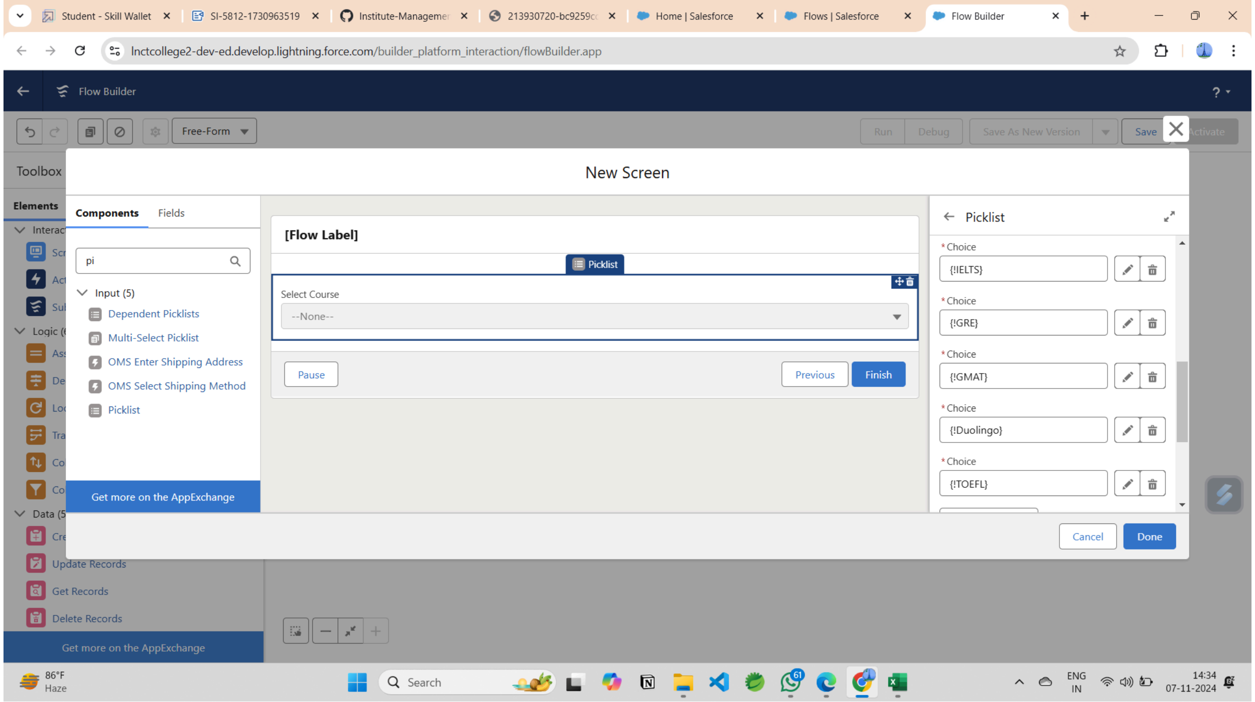
- **Scalability and Performance Enhancements**:
    - As the institution grows and the CRM sees increased usage, the system should scale to handle more data and users.
    - **Future Considerations**:
        - **Load Balancing**: Implement load balancing to distribute traffic and ensure that the CRM performs well during peak usage times.

        - **Database Optimization**: As the database grows, optimize queries and indexing to maintain fast response times.
        - Example: "Implementing database partitioning or indexing could improve performance as the number of bookings and feedback data increases."
- **Advanced Analytics and Reporting**:
    - **AI-Powered Insights**:
        - Leverage artificial intelligence (AI) to analyze facility usage patterns and predict future demand.
        - **Example**: "AI algorithms could suggest optimal booking times or facility upgrades based on usage trends and feedback data."
    - **Custom Dashboards**:
        - Allow users to create personalized dashboards to monitor key metrics.
        - Example: "Students could customize a dashboard to display their upcoming bookings, while administrators could track facility utilization across different departments."
- **Mobile App Integration**:
    - **Mobile Access**:
        - Develop a mobile version of the CRM, enabling students and administrators to access the system on the go.
        - Example: "A mobile app could allow students to book facilities or provide feedback directly from their smartphones, enhancing user experience and engagement."
    - **Push Notifications**:
        - Implement push notifications for important updates, such as booking approvals, reminders, or maintenance alerts.
        - Example: "Push notifications could be used to notify students about their booking status or remind them of upcoming bookings."
- **Integration with Other Systems**:
    - **Calendar Integration**:
        - Integrate with external calendar systems (like Google Calendar or Outlook) to automatically sync bookings and send calendar invites.
        - Example: "When a student books a facility, a calendar event is automatically created, and a reminder is sent via email or SMS."
    - **Room Booking Systems**:
        - Integrate with external room or facility management systems used by the institution for seamless operations.
        - Example: "Integrating with an existing building management system could allow the CRM to automatically check room availability based on real-time data."
- **Enhancing User Experience**:
    - **User Interface (UI) Overhaul**:

        - Continuously improve the UI to make it more intuitive and visually appealing.
        - Example: "A future UI update could include a cleaner, more modern design with improved navigation and better accessibility features."
    - **Accessibility Features**:
        - Enhance accessibility for users with disabilities by adding voice commands, screen reader support, and keyboard shortcuts.
        - Example: "Ensuring the CRM meets WCAG (Web Content Accessibility Guidelines) standards could improve accessibility for all users."
- **Internationalization and Localization**:
    - **Multi-language Support**:
        - Add support for multiple languages to cater to a wider range of users.
        - Example: "If the institution has an international student body, the CRM could be localized into different languages, allowing users to sel
        - ect their preferred language."
    - **Currency and Time Zone Support**:
        - Extend the CRM to handle multiple currencies and time zones if the institution has global facilities or branches.
        - Example: "For international institutions, the system could support currency conversion for certain facility bookings and adjust booking times based on users' time zones."

Screenshots





New Screen

pi

[Flow Label]

Picklist

Select Course

--None--

Picklist

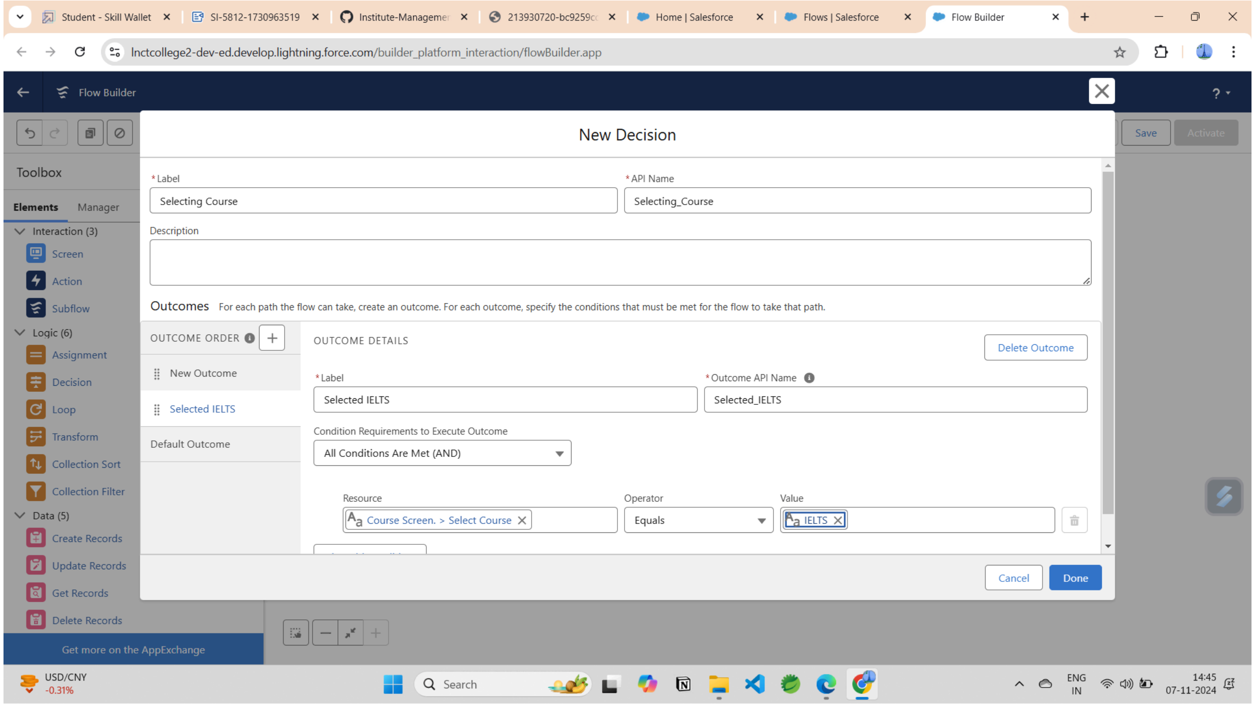
*Choice {!IELTS}

*Choice {!GRE}

*Choice {!GMAT}

*Choice {!Duolingo}

*Choice {!TOEFL}



New Decision

*Label: Selecting Course

*API Name: Selecting_Course

Description

Outcomes For each path the flow can take, create an outcome. For each outcome, specify the conditions that must be met for the flow to take that path.

New Outcome

Selected IELTS

Default Outcome

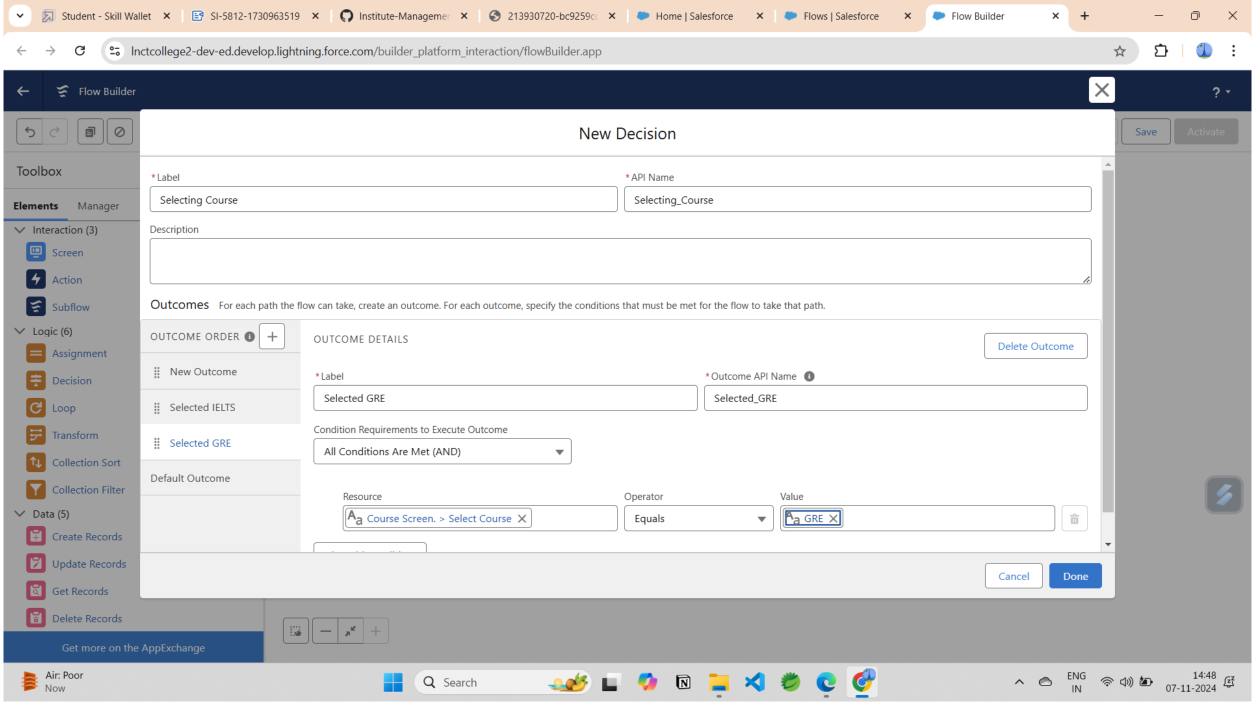
OUTCOME DETAILS

*Label: Selected IELTS

*Outcome API Name: Selected_IELTS

Condition Requirements to Execute Outcome: All Conditions Are Met (AND)

Resource	Operator	Value
Course Screen. > Select Course	Equals	IELTS



New Decision

*Label: Selecting Course

*API Name: Selecting_Course

Description

Outcomes For each path the flow can take, create an outcome. For each outcome, specify the conditions that must be met for the flow to take that path.

OUTCOME ORDER

- New Outcome
- Selected IELTS
- Selected GRE
- Default Outcome

OUTCOME DETAILS

Delete Outcome

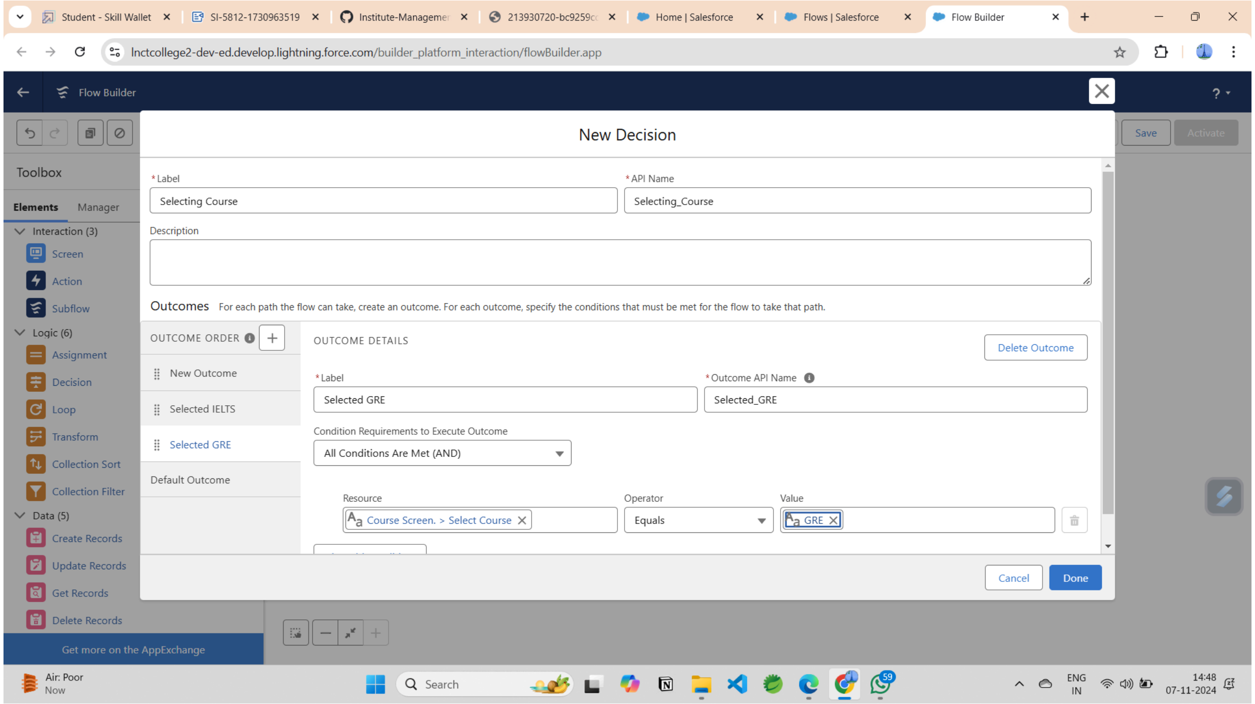
*Label: Selected GRE

*Outcome API Name: Selected_GRE

Condition Requirements to Execute Outcome: All Conditions Are Met (AND)

Resource	Operator	Value
Course Screen. > Select Course	Equals	GRE

Cancel Done



New Decision

*Label: Selecting Course

*API Name: Selecting_Course

Description

Outcomes For each path the flow can take, create an outcome. For each outcome, specify the conditions that must be met for the flow to take that path.

OUTCOME ORDER

- New Outcome
- Selected IELTS
- Selected GRE
- Default Outcome

OUTCOME DETAILS

Delete Outcome

*Label: Selected GRE

*Outcome API Name: Selected_GRE

Condition Requirements to Execute Outcome: All Conditions Are Met (AND)

Resource: Course Screen. > Select Course
Operator: Equals
Value: GRE

Cancel Done