Project Documentation: A CRM Application to Handle the Clients and their property Related Requirements.

Category: Salesforce

Skills Required:
Salesforce Admin,Salesforce Developer

Link :https://www.salesforce.com/trailblazer/shobhitabhatt

Project Description:

Dreams World Properties integrates Salesforce to streamline customer interactions. Website engagement triggers automated record creation in Salesforce, capturing customer details and preferences. Salesforce categorizes users as approved or non-approved, offering tailored property selections to approved users. This enhances user experience and efficiency, providing personalized recommendations and broader listings. Seamless integration optimizes operations, improving customer engagement and facilitating growth in the real estate market.

Objectives:

- To automatically capture client information using JotForm and create corresponding records in Salesforce.
- To manage customer and property data through custom objects.
- To define roles and profiles that mirror the organizational hierarchy.
- To automate property approval processes using record-triggered flows.
- To create a user-friendly application interface with custom components.

Scope:

Inclusions:

- Integration of JotForm with Salesforce.
- Creation of custom objects for managing customer and property data.
- Definition of roles, profiles, and users.
- Development of an automated approval process for property records.
- Creation of custom Lightning Web Components (LWC) and app pages.

Exclusions:

- Integration with external property listing platforms.
- Development of mobile applications.

Stakeholders:

- **Customer**

- **Role:** End-user interacting with the sales team and CRM system.
    - **Responsibilities:** Provide feedback, track inquiries, and engage with the sales team.
- **Manager**
    - **Role:** Oversees sales operations and CRM alignment with business objectives.
    - **Responsibilities:** Set goals, monitor performance, and utilize CRM reports.
- **Sales Executive**
    - **Role:** Manages client relationships and transactions.
    - **Responsibilities:** Use CRM for lead management, interactions, and tracking.
- **Sales Representative**
    - **Role:** Engages with clients to gather requirements and facilitate sales.
    - **Responsibilities:** Manage client data, track property status, and support sales executives.

System Requirements:

Functional Requirements

- **JotForm Integration:**
    - Create a JotForm to capture customer information.
    - Automatically create a customer record in Salesforce when the form is submitted.
- **Custom Objects:**
    - Create **Customer** and **Property** objects using data from a provided spreadsheet.
    - Define relationships between these objects.
- **Role Hierarchy:**
    - Create roles for **Sales Executive** (reports to Sales Representative), **Sales Manager** (reports to Sales Executive), and **Customer** (reports to Sales Manager).
- **Property Details App:**
    - Develop an app to manage and view property details within Salesforce.
- **Profiles and Users:**
    - Create profiles for **Customer** and **Manager** with appropriate access permissions.
    - Create 4 users and assign them to the respective roles.
- **Approval Process:**
    - Create an approval process for the Property object to manage property-related approvals.
    - Implement a record-triggered flow to automatically submit the approval process.
- **App Page and LWC:**
    - Create a custom Lightning Web Component (LWC).
    - Create an app page and drag the LWC onto the page.
- **Apex Class Permissions:**
    - Assign access to specific Apex classes based on profiles.

Non-Functional Requirements

- **Scalability**: The system should handle an increasing number of customer and property records without performance degradation.

- **Performance**: User interactions with the CRM should have a response time of fewer than 2 seconds.
- **Security**: Data security is paramount; access controls should be enforced based on roles and profiles.
- **Usability**: The CRM application should have an intuitive user interface for ease of use.

System Design:

Architecture Overview

The CRM system is built on Salesforce, utilizing its robust platform for customer and property management. Key components include:

- **Salesforce Platform**: The backbone for data storage, business logic, and user management.
- **JotForm Integration**: For capturing client data and automatically creating Salesforce records.
- **Custom Objects:** Customer and Property objects tailored to the business needs.
- **Approval Processes and Flows**: Automating the approval of property records.
- **Lightning Web Components (LWC)**: Custom UI components for enhanced user interaction.
- **Profiles and Roles**: Configured to enforce data access and security.

Data Model

- **Customer Objec**t: Fields include Name, Contact Information, Property Preferences, and Interaction History.
- **Property Object**: Fields include Property Name, Location, Price, Status, and Customer Associations.
- **Relationships**: Customer objects are linked to Property objects to track inquiries and transactions.

User Interface Design

- **App Page:** Centralized page for managing customers and properties, featuring LWC components.
- **LWC Component**: A custom component displaying detailed property information, including status and customer interactions.
- **Navigation**: Easy access to customer and property records, approval processes, and reports.

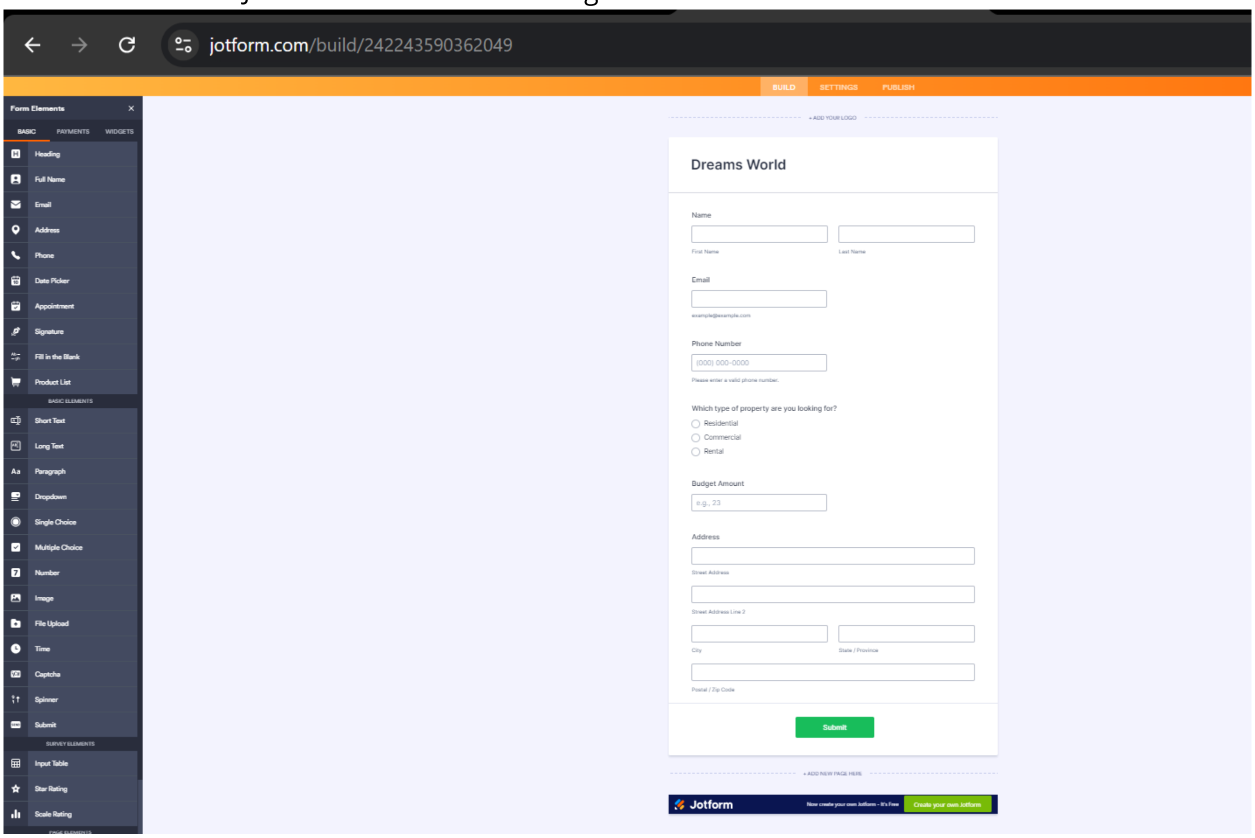
Milestones and Implementation Plan

Milestone 1: Create a JotForm and Integrate with Salesforce

- **Create JotForm:**
    - Design fields to capture essential customer information.

- Configure form settings for Salesforce integration.

- **Integrate JotForm with Salesforce:**
  - Set up the integration using JotForm's Salesforce connector.
  - Map JotForm fields to Salesforce Customer object fields.
- **Test Integration:**
  - Verify successful creation of Customer records in Salesforce.
  - Conduct end-to-end tests to ensure data accuracy and integration reliability.
- **Documentation and Training:**
  - Document the integration process and provide training for users who will manage the JotForm and Salesforce integration.



SCREENSHOT: JotForm creation interface showing form design and settings.

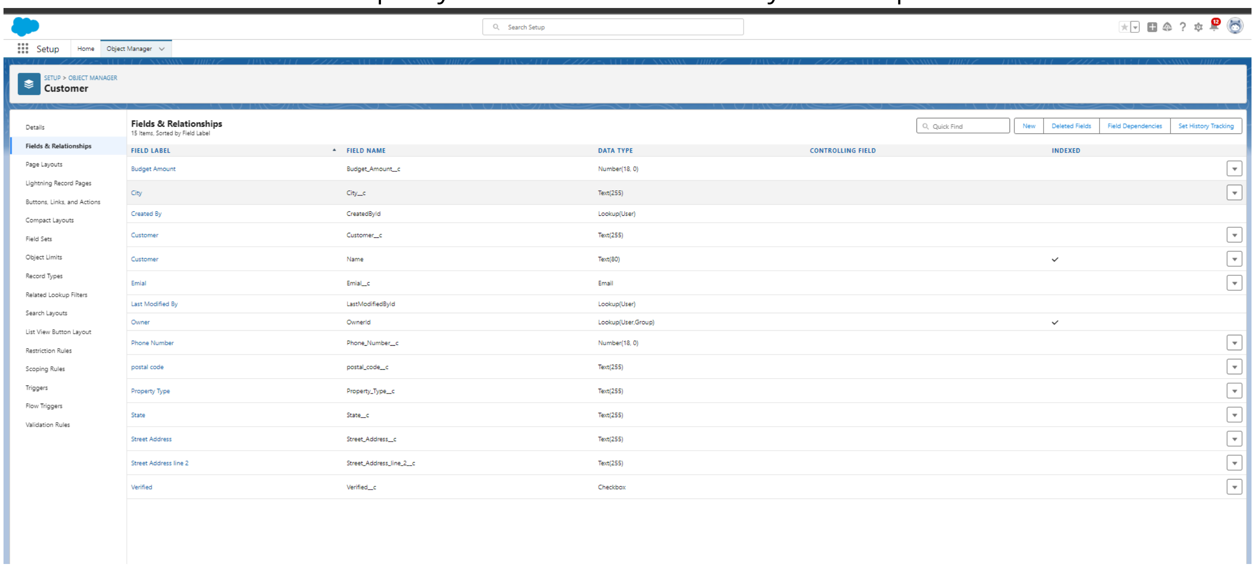
URL: https://form.jotform.com/242243590362049

Milestone 2: Create Objects from Spreadsheet

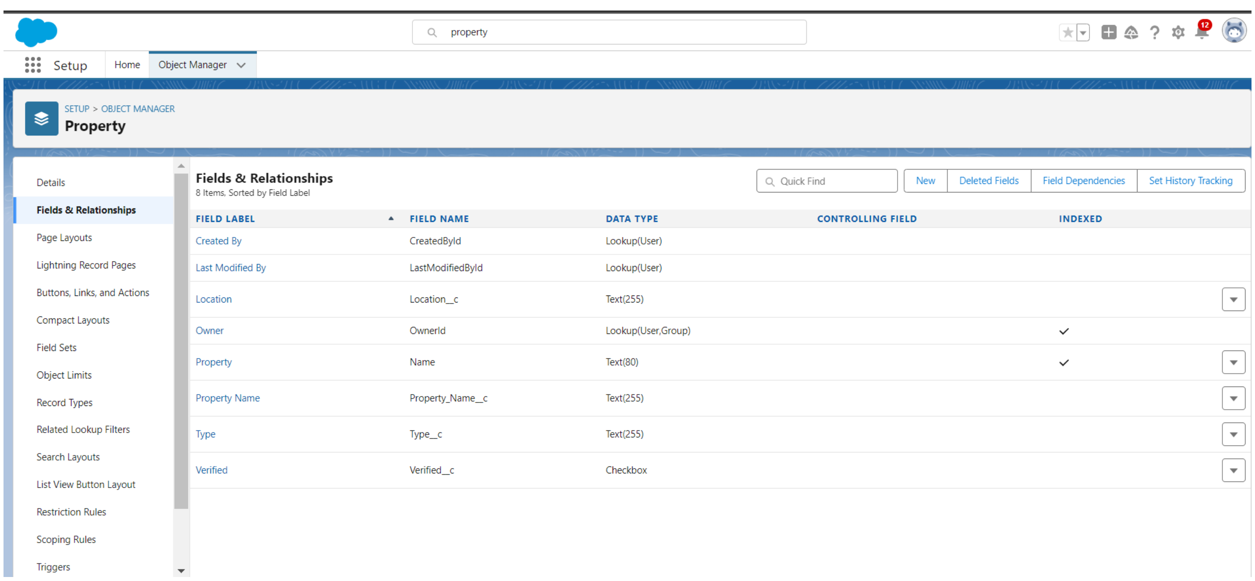
- **Create Customer Object:**
  - Define object fields based on spreadsheet data.
  - Set up field types, validation rules, and record types.
- **Create Property Object:**
  - Configure fields to capture property details and statuses.
  - Establish relationships between Customer and Property objects.
- **Data Import:**

- Use Salesforce Data Import Wizard or Data Loader for initial data upload.
- Validate data integrity post-import.

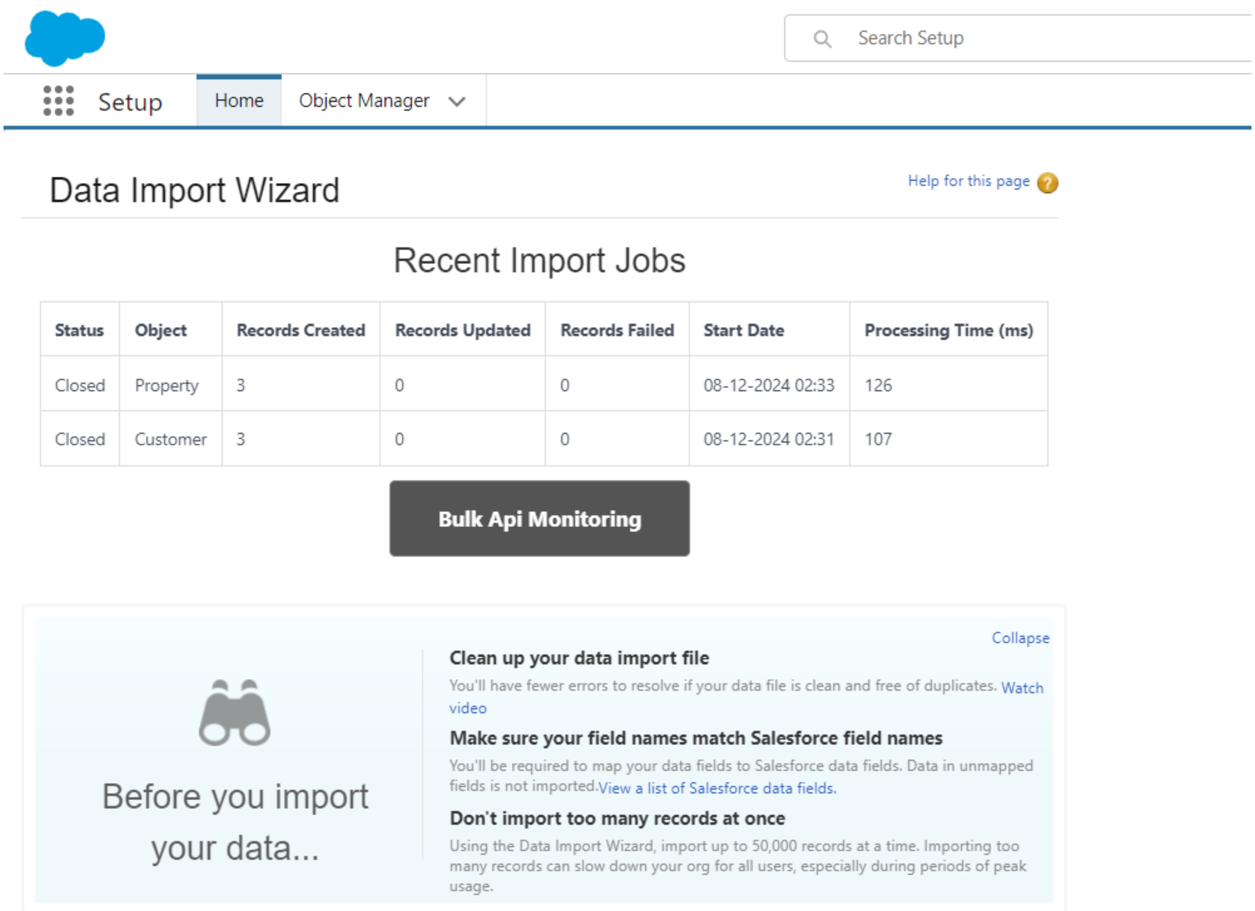
- **Data Quality Checks:**
    - Conduct data quality checks to ensure accuracy and completeness.



Customer object schema and field definitions



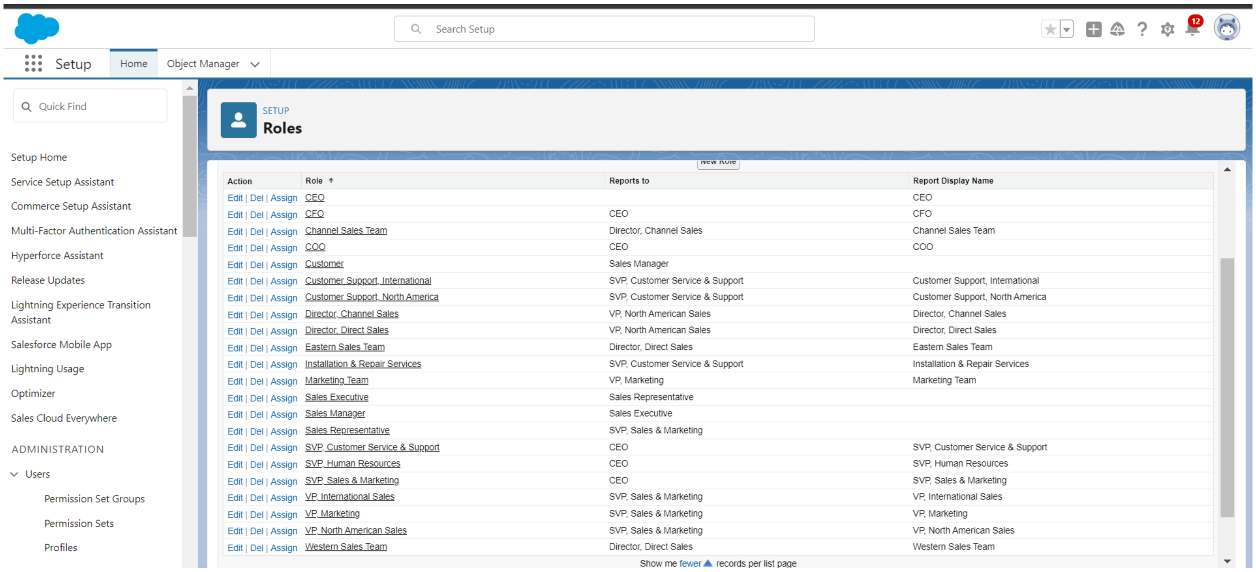
Property object schema and field definitions



Data Import Wizard used for importing customer and property data and Data quality check report after import.

Milestone 3: Create Role Hierarchy

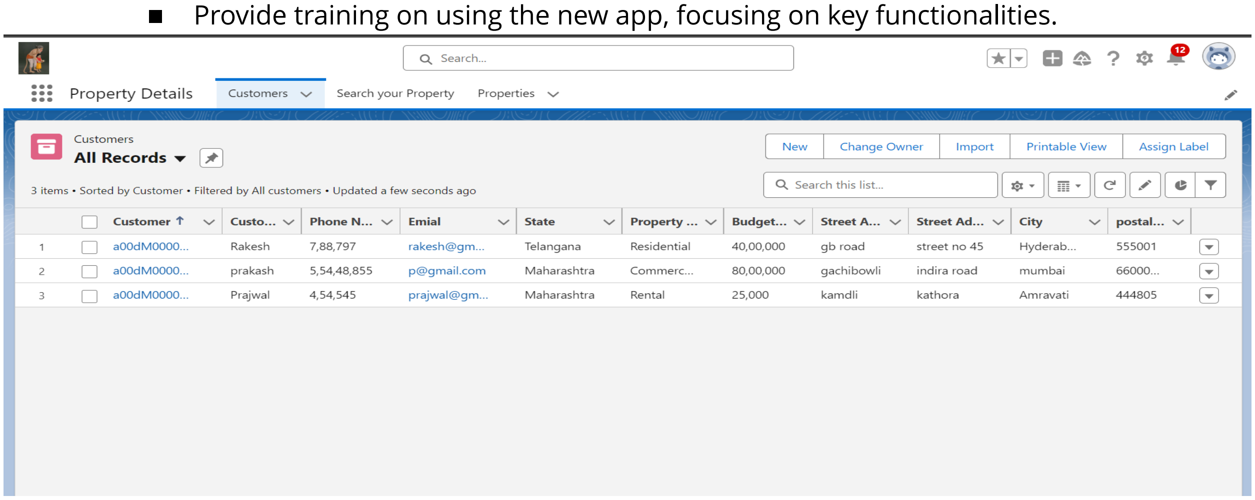
- **Define Roles:**
  - **Sales Executive:** Manage client interactions.
  - **Sales Manager:** Oversee Sales Executives and manage performance.
  - **Customer:** End-user role with limited access.
- **Create and Assign Roles:**
  - Configure roles in Salesforce and assign them to appropriate users.
- **Review and Test Roles:**
  - Ensure role hierarchy supports the required reporting structures.
  - Test role-based access to verify permissions and data visibility.
- **Documentation:**
  - Document role definitions and reporting structures.



Role hierarchy showing Sales Executive, Sales Manager, and Customer roles

Milestone 4: Create Property Details App

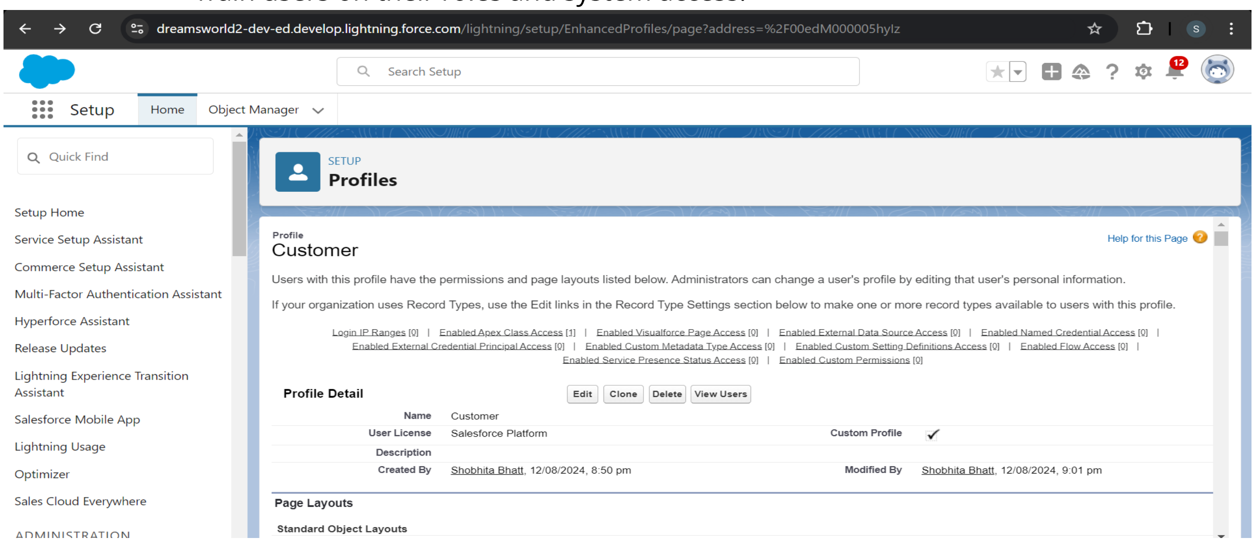
- **Create App:**
  - Use the Salesforce App Builder to create the "Property Details App."
  - Include tabs for Customer and Property objects.
- **Customize Layout:**
  - Configure page layouts and object views for usability.
  - Add relevant related lists and quick actions.
- **Testing:**
  - Perform functionality tests to ensure all features work as intended.
  - Gather feedback from potential users and make adjustments.
- **User Training:**
  - Provide training on using the new app, focusing on key functionalities.



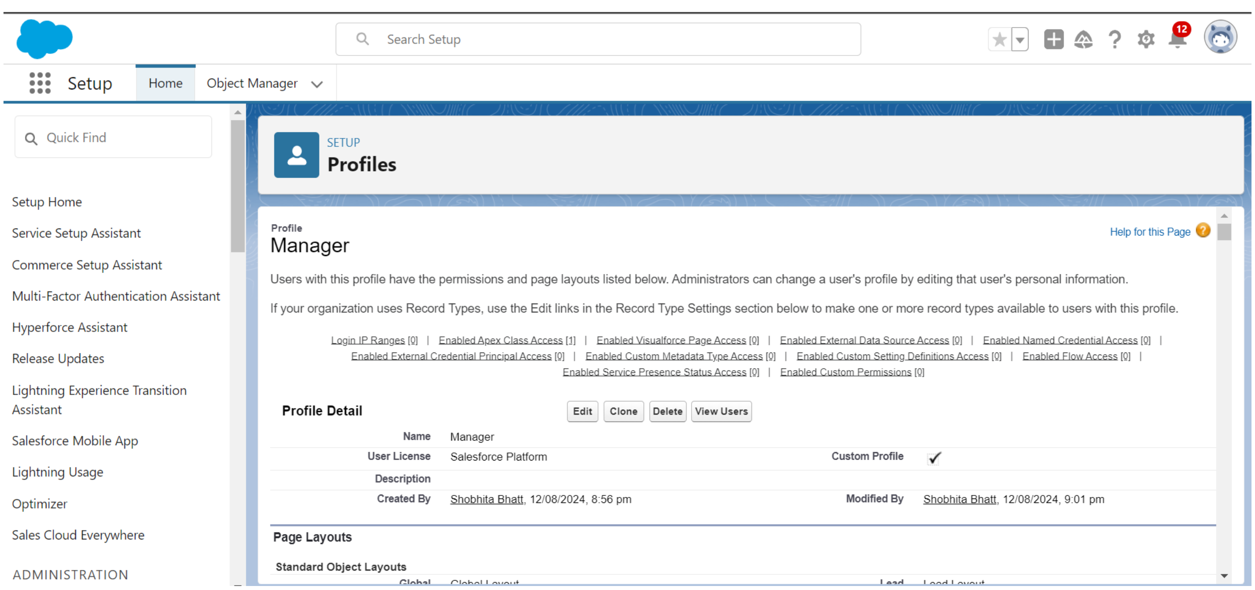
App Builder setup for the Property Details App.

Milestone 5: Create Profiles and Users

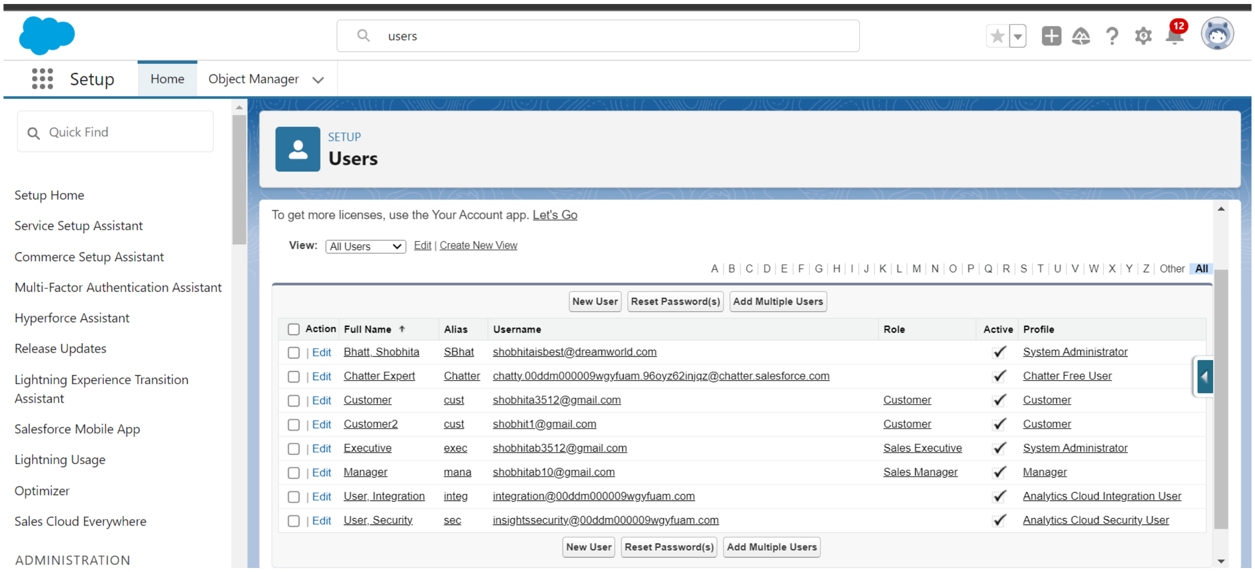
- **Create Profiles:**
  - Define profiles for **Customer** and **Manager** with appropriate permissions.
- **Create Users:**
  - Set up 4 users and assign them roles and profiles.
- **Profile Permissions:**
  - Configure object and field-level permissions based on profile needs.
- **User Testing:**
  - Test user access and permissions to ensure compliance with defined roles.
- **Training and Support:**
  - Train users on their roles and system access.



Profile settings for Customer profile



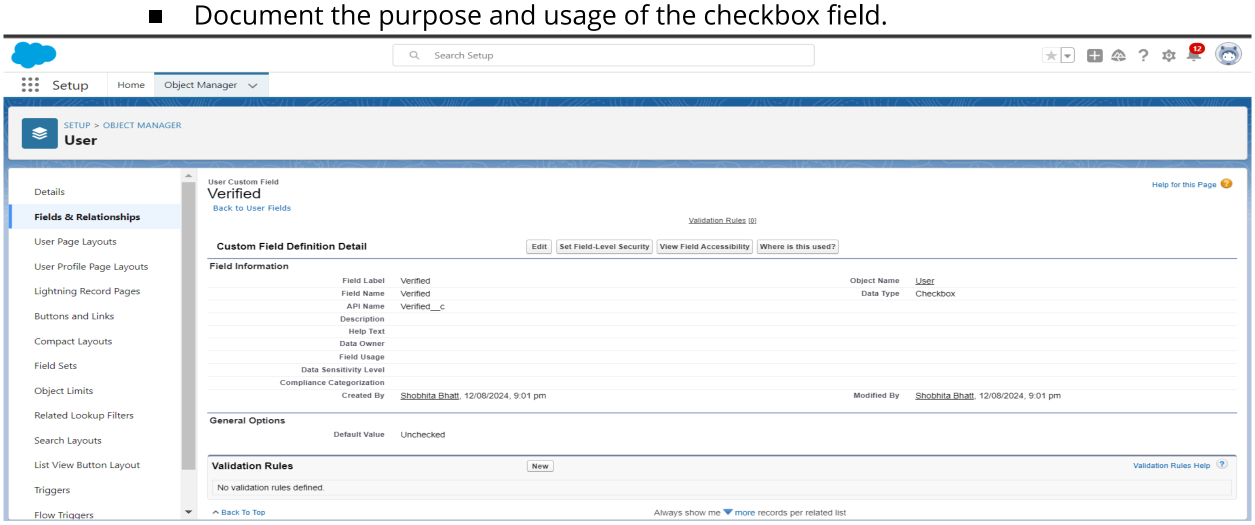
Profile settings for Manager profile



User creation and role assignment interface

Milestone 6: Create a Checkbox Field on User Object

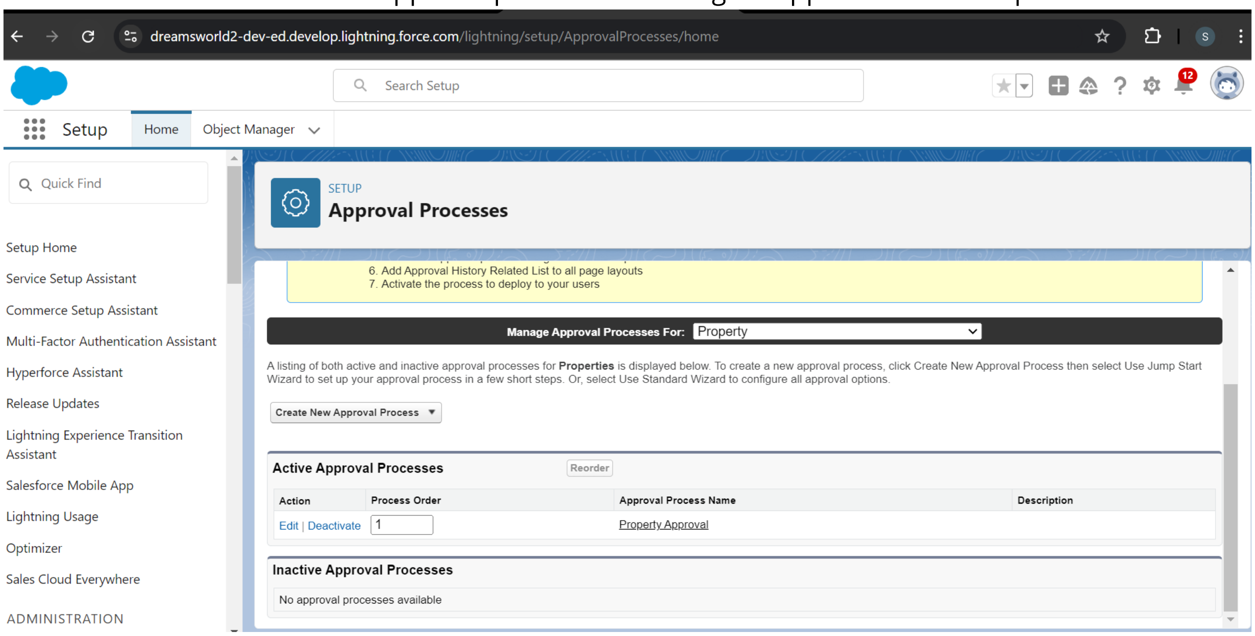
- **Create Checkbox Field:**
  - Define a checkbox field named “Verified” on the User object.
- **Update Layouts:**
  - Add the checkbox to relevant user page layouts.
- **Testing:**
  - Verify the functionality of the checkbox and its impact on approval processes.
- **Documentation:**
  - Document the purpose and usage of the checkbox field.



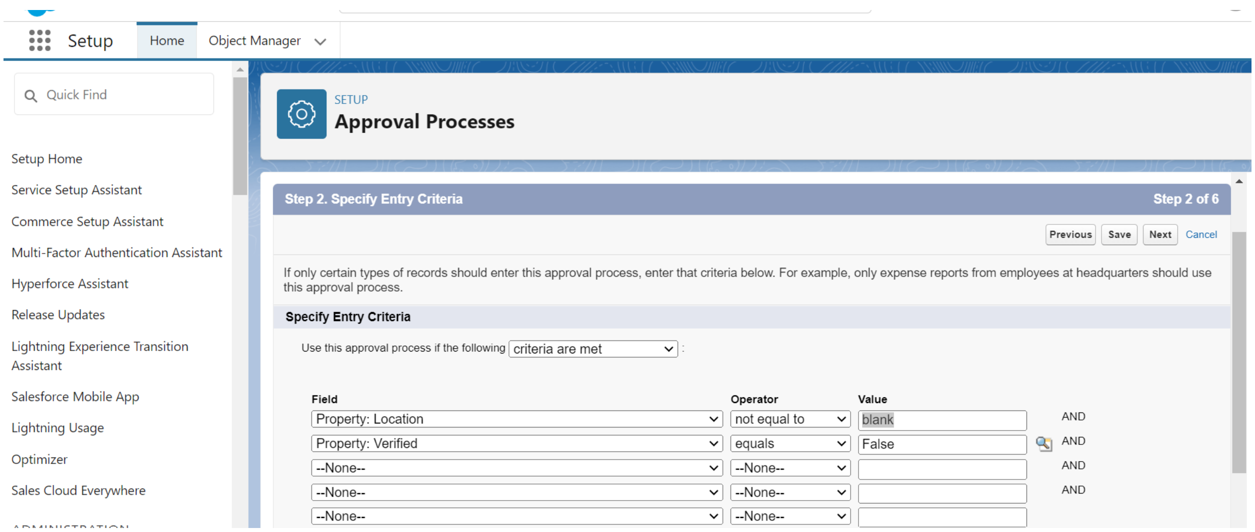
Creating the checkbox field on the User object.

Milestone 7: Create an Approval Process for Property Object

- **Define Approval Steps:**
  - Create a multi-step approval process for Property records.
- **Configure Entry Criteria:**
  - Set criteria for records that should trigger the approval process.
- **Assign Approvers:**
  - Configure roles and users responsible for each approval step.
- **Testing:**
  - Submit test records to validate the approval workflow.
- **Documentation:**
  - Document the approval process and configure approval email templates.



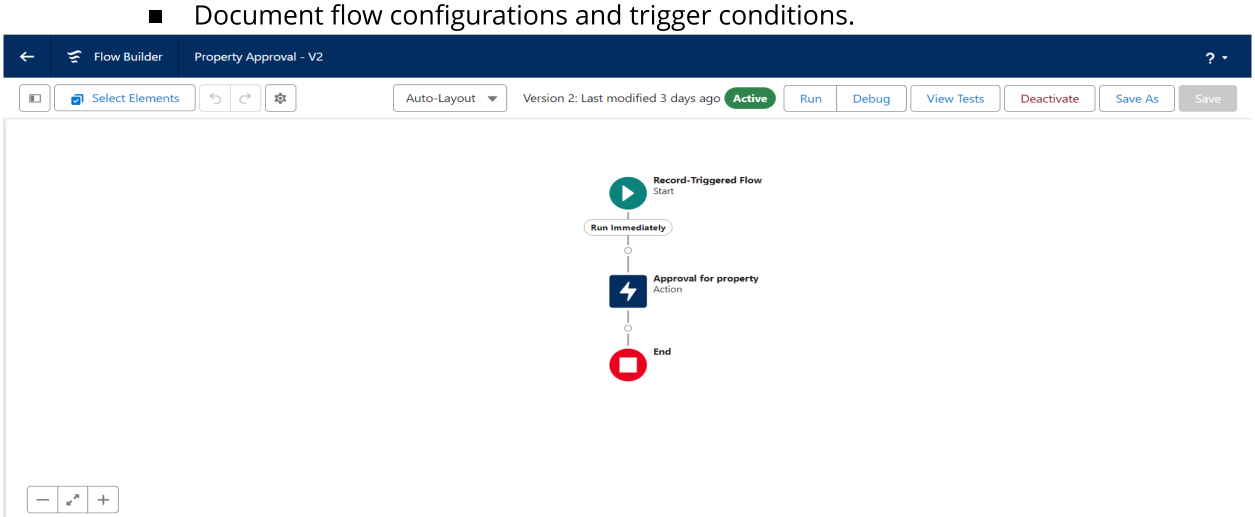
Setting up the approval process for the Property object.



Configuring entry criteria for property records

Milestone 8: Create a Record Trigger Flow to Automate Approval Process

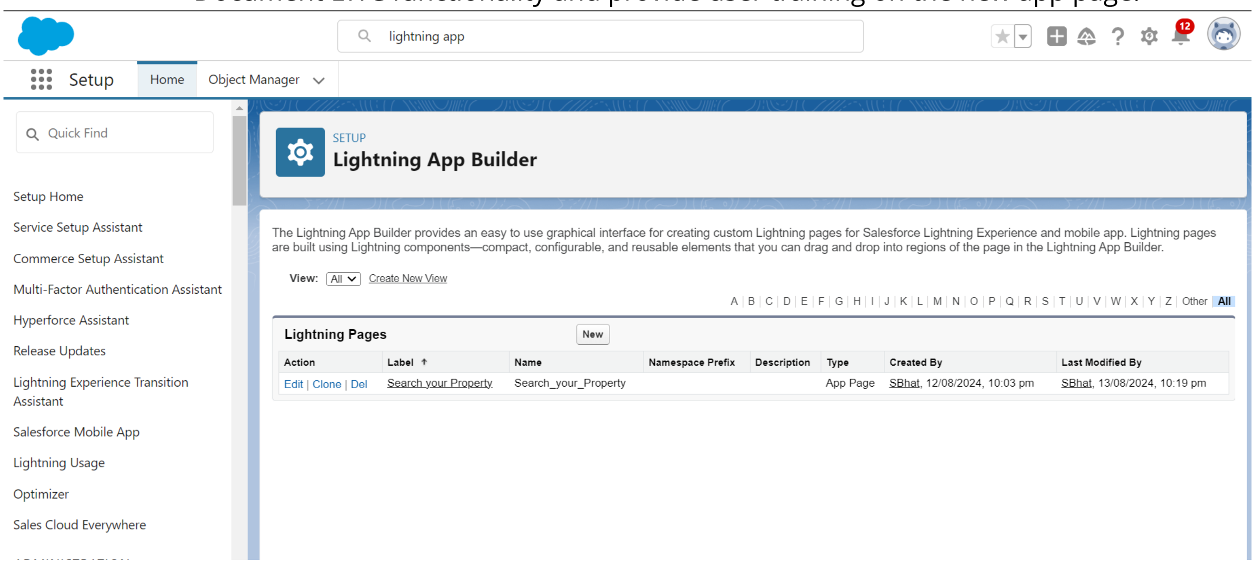
- **Create Flow:**
  - Develop a record-triggered flow in Salesforce Flow Builder to handle automatic submission of Property records for approval.
- **Configure Flow Logic:**
  - Define the flow's trigger conditions and actions.
- **Testing:**
  - Test the flow to ensure it submits records for approval under the correct conditions.
- **Documentation:**
  - Document flow configurations and trigger conditions.



Configuring flow logic for automatic approval submission

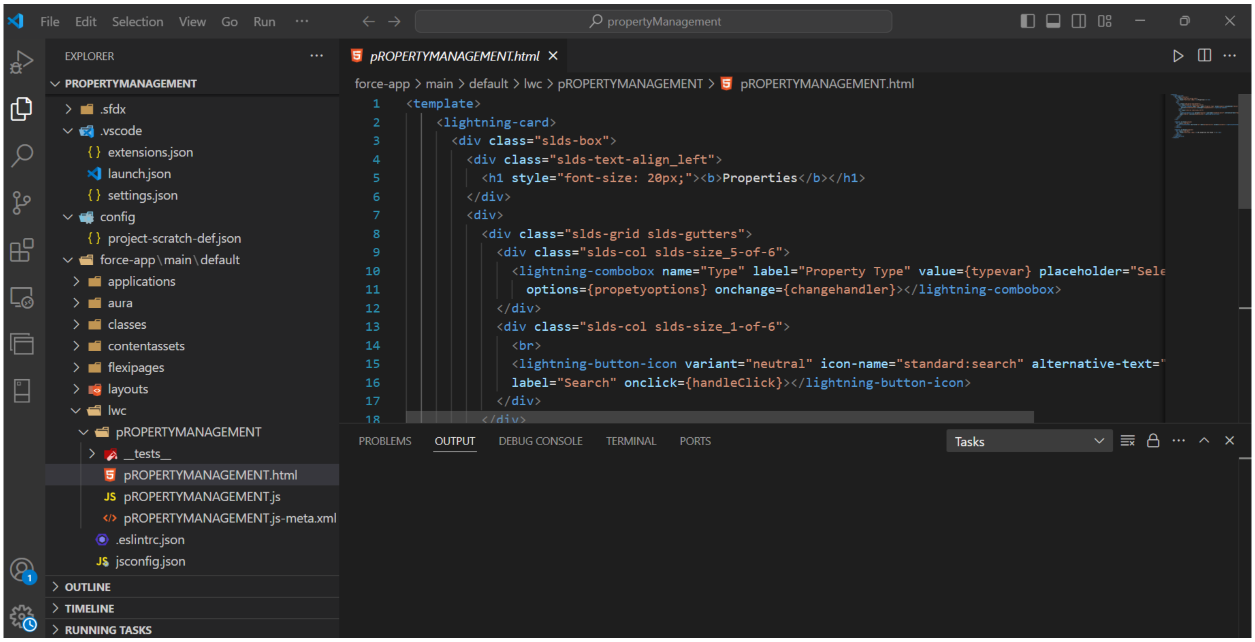
Milestone 9: Create an App Page and LWC Component

- **Develop LWC:**
  - Build a custom Lightning Web Component to display relevant property and customer information.
- **Create App Page:**
  - Use the App Builder to create a new page and add the LWC component.
- **Customize Page Layout:**
  - Adjust the page layout for optimal user experience and data presentation.
- **Testing:**
  - Test the LWC and app page to ensure functionality and performance.
- **Documentation and Training:**
  - Document LWC functionality and provide user training on the new app page.

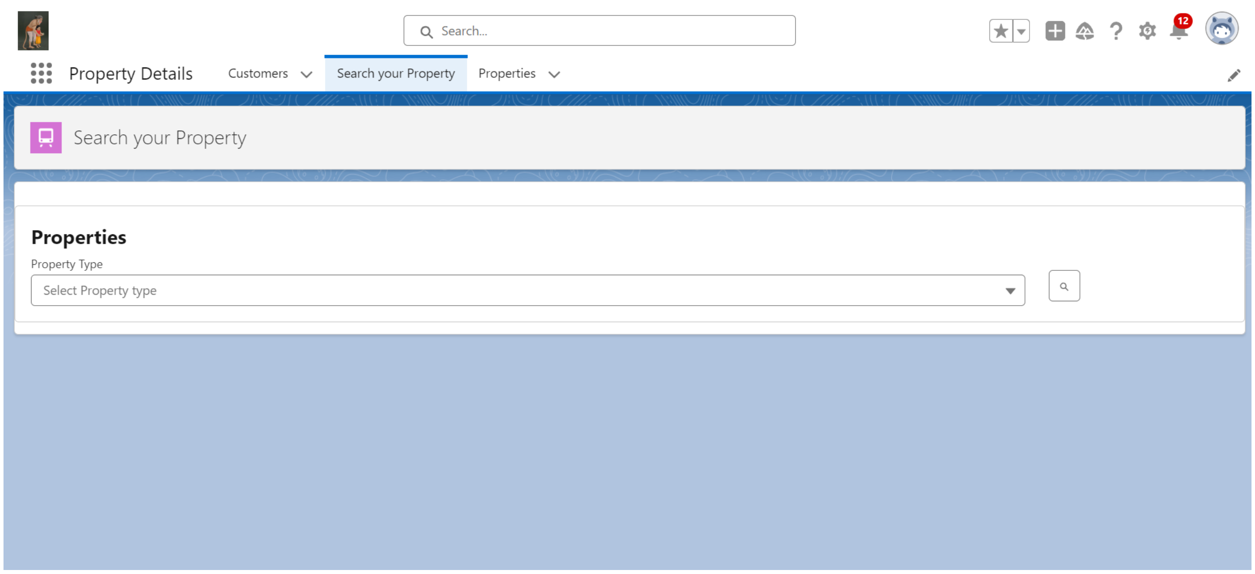


Milestone 10: Assign Apex Class Permissions

- **Identify Apex Classes:**
  - Determine which Apex classes need access based on functionality and user profiles.
- **Assign Permissions:**
  - Update profiles to grant access to specific Apex classes.
- **Testing:**
  - Verify that Apex class permissions work correctly and that users can execute necessary operations.
- **Documentation:**
  - Document permissions and any associated security considerations.



Code preview of the custom Lightning Web Component



Customized layout for the Property Details App page

Testing and Quality Assurance: Testing Strategy

Unit Testing: Verify individual components' functionality.

Integration Testing: Ensure all components and integrations work together.

System Testing: Validate