

Dispatcher,Parcel delivery and tracking application

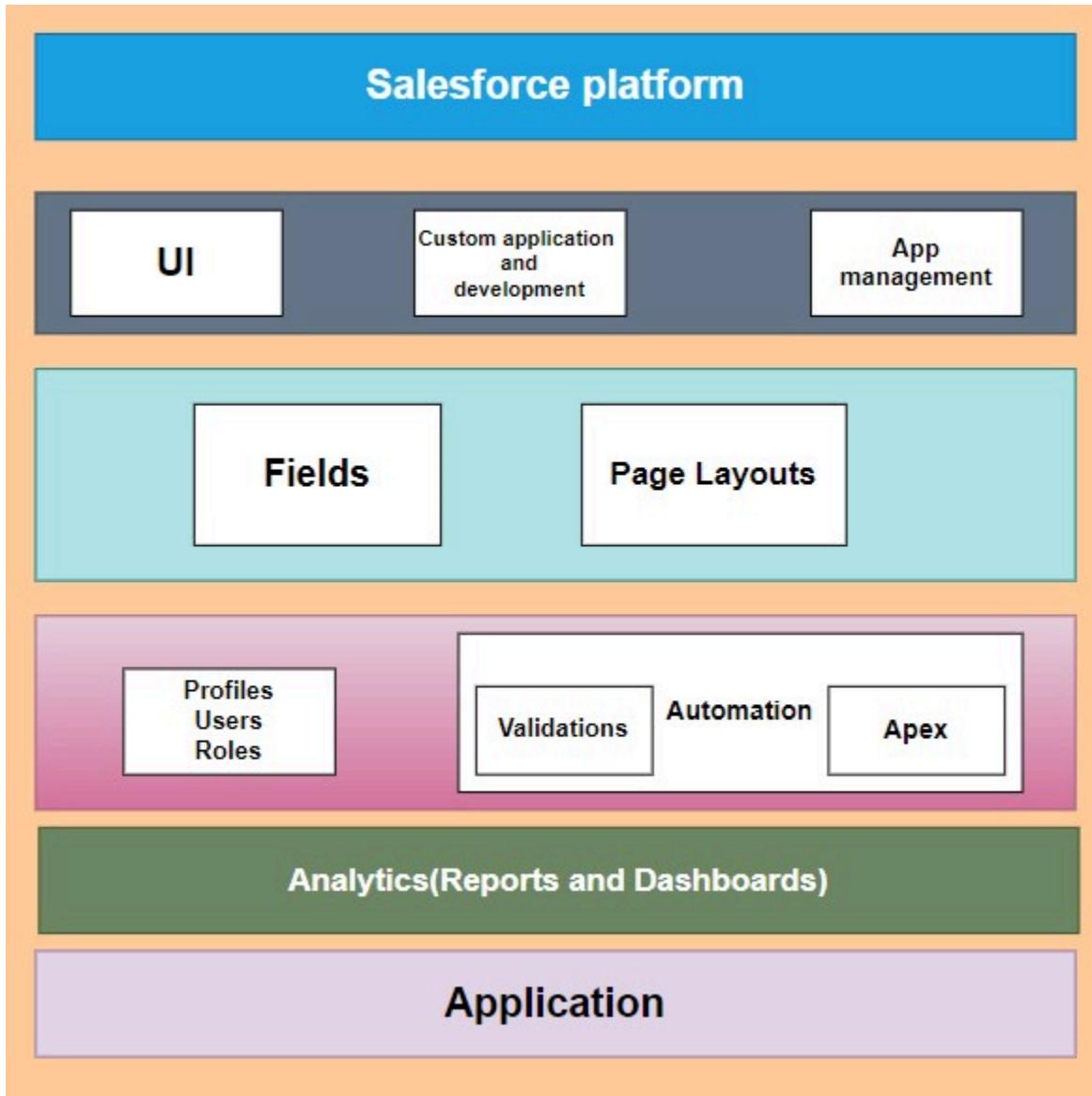
User Story :

Streamline your parcel delivery operations with our Salesforce-powered application. Effortlessly assign parcels to delivery agents, monitor real-time tracking, and enhance customer satisfaction. In today's fast-paced world, efficient parcel delivery and tracking services are essential for businesses and organizations of all sizes. Whether you're managing a small courier service or overseeing a large-scale logistics operation, our Salesforce-powered Dispatcher Parcel Delivery and Tracking Application is designed to meet your specific needs, optimize your processes, and enhance the overall customer experience.

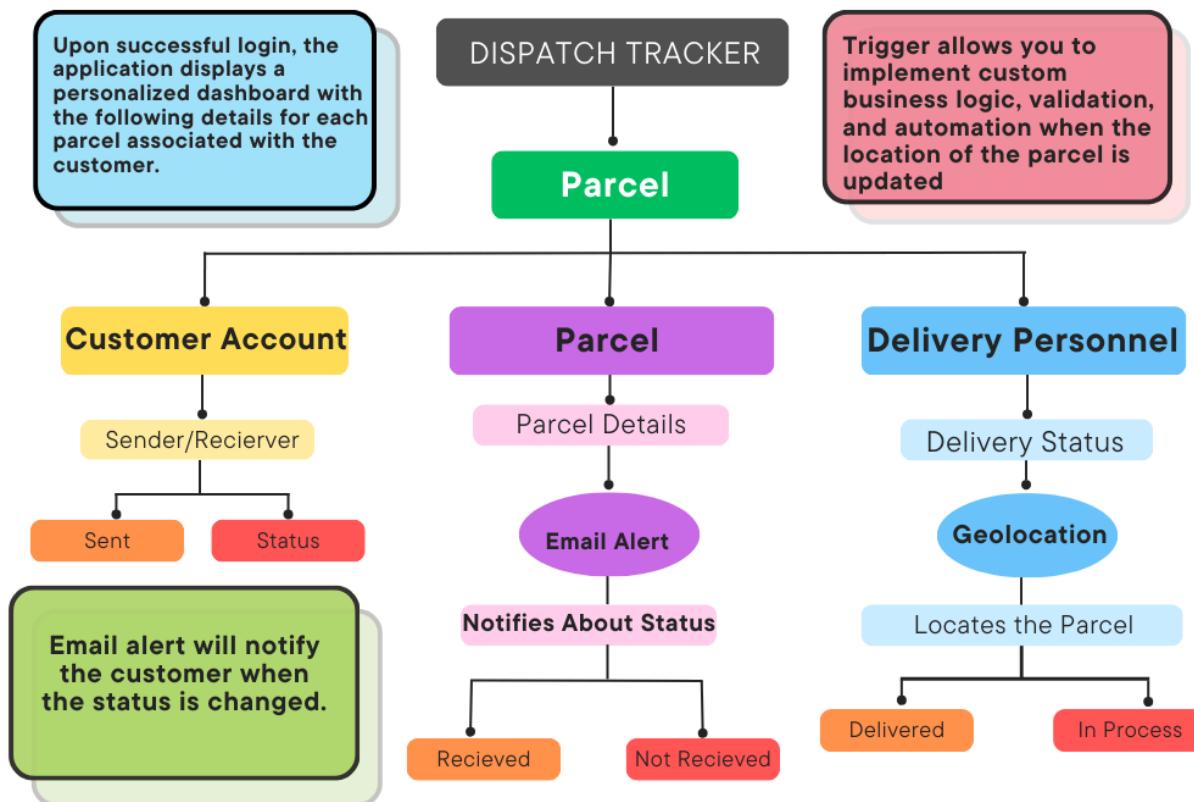
What you'll learn

1. Real Time Salesforce Project
2. Object & Relationship in Salesforce
3. Formula fields.
4. Apex
5. Apex Triggers
6. Lightning Web Components
7. Schedule Apex
8. Reports and Dashboards
9. Flows

Technical Architecture:



Project Flow:



Milestone 1- Create a Salesforce Developer Account :

What Is a Salesforce Developer Account?

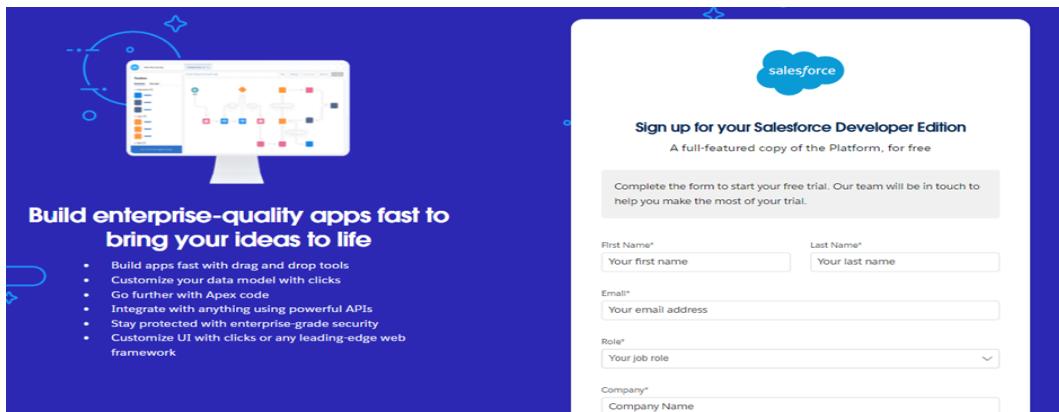
A Salesforce Developer Account, often referred to as a Developer Org, is a free Salesforce environment provided by Salesforce for developers to build, test, and experiment with Salesforce applications and customizations. It is a fully functional Salesforce instance, but it comes with some limitations and features tailored for development purposes.

Activity 1:

Creating Developer Account:

Creating a developer org in salesforce.

1. Go to <https://developer.salesforce.com/signup>
2. On the sign up form, enter the following details :



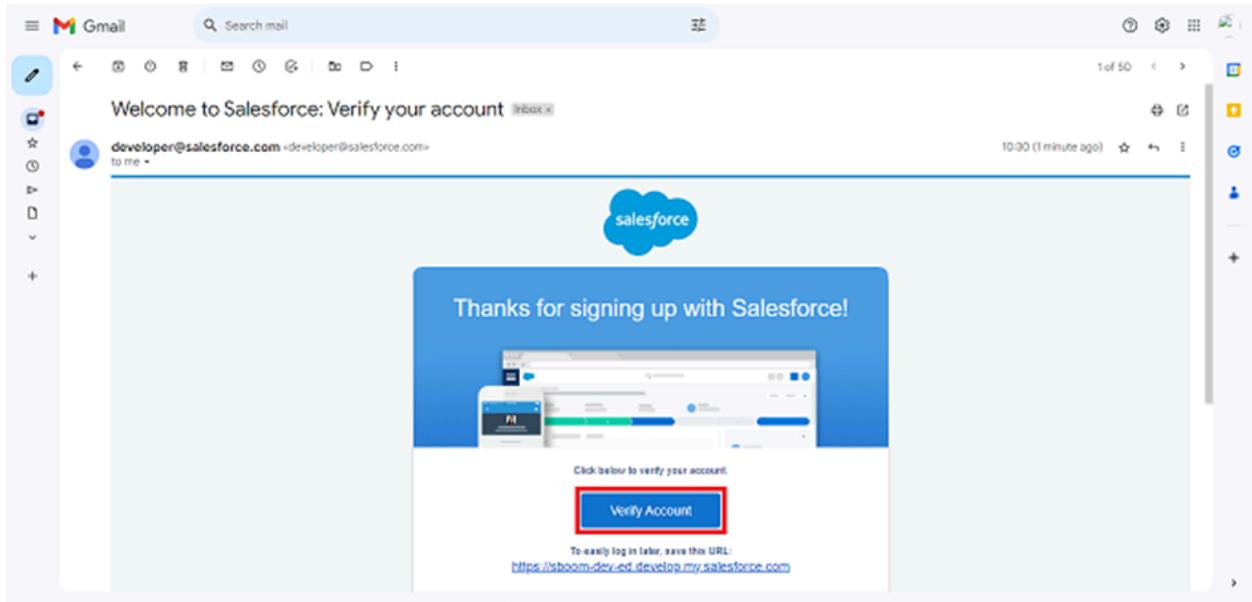
- 1) First name & Last name
- 2) Email
- 3) Role : Developer
- 4) Company : College Name
- 5) County : India
- 6) Postal Code : pin code
- 7) Username : should be a combination of your name and company

This need not be an actual email id, you can give anything in the format : username@organization.com

Click on sign me up after filling these.

Activity 2: Account Activation:

1. Go to the inbox of the email that you used while signing up. Click on the verify account to activate your account. The email may take 5-10mins.



2. Click on Verify Account.
3. Give a password and answer a security question and click on change password.



Change Your Password

Enter a new password for lead@sb.oom.
Make sure to include at least:

- 8 characters
- 1 letter
- 1 number

* New Password
 Good

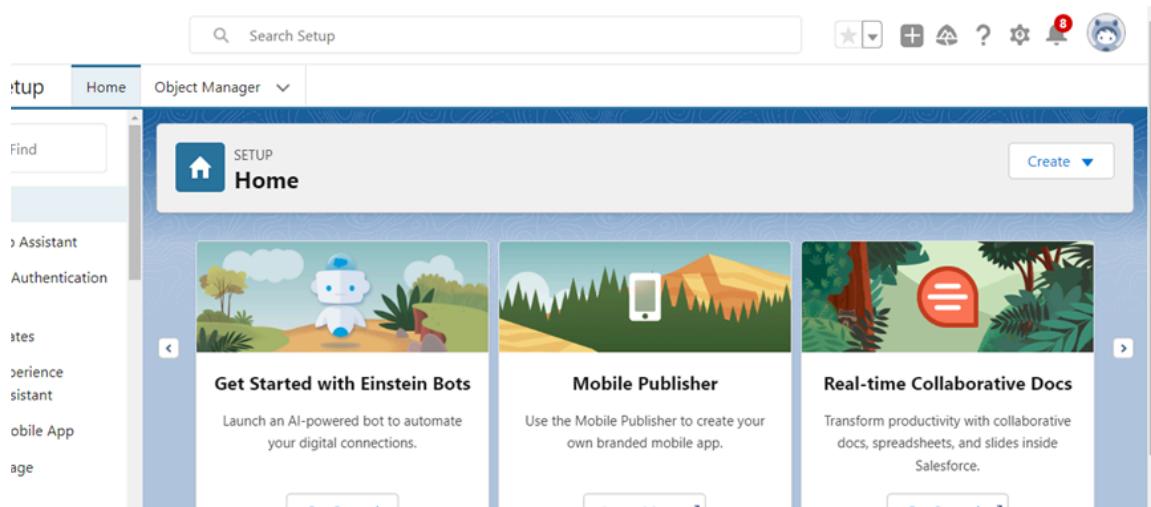
* Confirm New Password
 Match

Security Question
In what city were you born?

* Answer
 asdfghijkl

Change Password

4. Then you will redirect to your salesforce setup page.



The screenshot shows the Salesforce Setup Home page. The top navigation bar includes 'Search Setup', 'Home', 'Object Manager', and various icons for account management. The main content area is titled 'SETUP Home' and features three promotional cards:

- Get Started with Einstein Bots**: Launch an AI-powered bot to automate your digital connections.
- Mobile Publisher**: Use the Mobile Publisher to create your own branded mobile app.
- Real-time Collaborative Docs**: Transform productivity with collaborative docs, spreadsheets, and slides inside Salesforce.

Milestone 2: Object

What are the objects required ?

Salesforce objects are database tables that permit you to store data that is specific to an organization. It consists of fields (columns) and records (rows).

Custom objects that are created by users for our case are Parcel object saves the essential information that a parcel holds i.e, parcel ID etc, similarly the other objects hold the information that is related to them, Customer Account and Delivery Personnel. They supply information that is essential to this project.

Activity -1

Create a custom object for Parcel

To create a custom object, follow these steps :

1. Click on the Gear Icon → From setup click on object manager
2. Click create, select custom object.
3. Fill in the label as "Parcel".
4. Fill in the plural label as "Parcel".
5. Record name : "Parcel Name"
6. Select the data type as "Text".
7. In the Optional Features section, select Allow Reports and Track Field History.
8. In the Deployment Status section, ensure Deployed is selected.
9. In the Search Status section, select Allow Search.
10. In the Object Creation Options section, select select these options:
Add Notes and Attachments related list to default page layout
Launch New Custom Tab Wizard after saving this custom object
11. Leave everything else as is, and click Save.

Activity -2

Create a custom object for Customer Account

To create a custom object, follow these steps :

1. Click on the Gear Icon → From setup click on object manager
2. Click create, select custom object.
3. Fill in the label as "Customer Account".
4. Fill in the plural label as "Customer Accounts".

5. Record name : "Customer Accounts"
6. Select the data type as "Text".
7. In the Optional Features section, select Allow Reports and Track Field History.
8. In the Deployment Status section, ensure Deployed is selected.
9. In the Search Status section, select Allow Search.
10. In the Object Creation Options section, select select these options:
Add Notes and Attachments related list to default page layout
Launch New Custom Tab Wizard after saving this custom object
11. Leave everything else as is, and click Save.

Activity - 3

Create a custom object for Delivery Personnel

1. Click on the Gear Icon → From setup click on object manager
2. Click create, select custom object.
3. Fill in the label as "Delivery Personnel".
4. Fill in the plural label as "Delivery Personnels".
5. Record name : "Delivery Personnel"
6. Select the data type as "Text".
7. In the Optional Features section, select Allow Reports and Track Field History.
8. In the Deployment Status section, ensure Deployed is selected.
9. In the Search Status section, select Allow Search.
10. In the Object Creation Options section, select select these options:
Add Notes and Attachments related list to default page layout
Launch New Custom Tab Wizard after saving this custom object
11. Leave everything else as is, and click Save.

Milestone 3 : Tabs

What is the requirement of Tab?

Tab is a user interface element that allows users to navigate to different sections of the platform, such as Customer Account, Parcel and Delivery Personnel. Tabs are used to access custom objects and custom pages. These are located at the top of the screen and can be customized to fit the needs.

Activity - 1

How to create a tab

As we selected to launch a custom tab wizard in step 10, a custom tab wizard appears wherein We customize the look of the "Customer Account". object's tab. To do that :

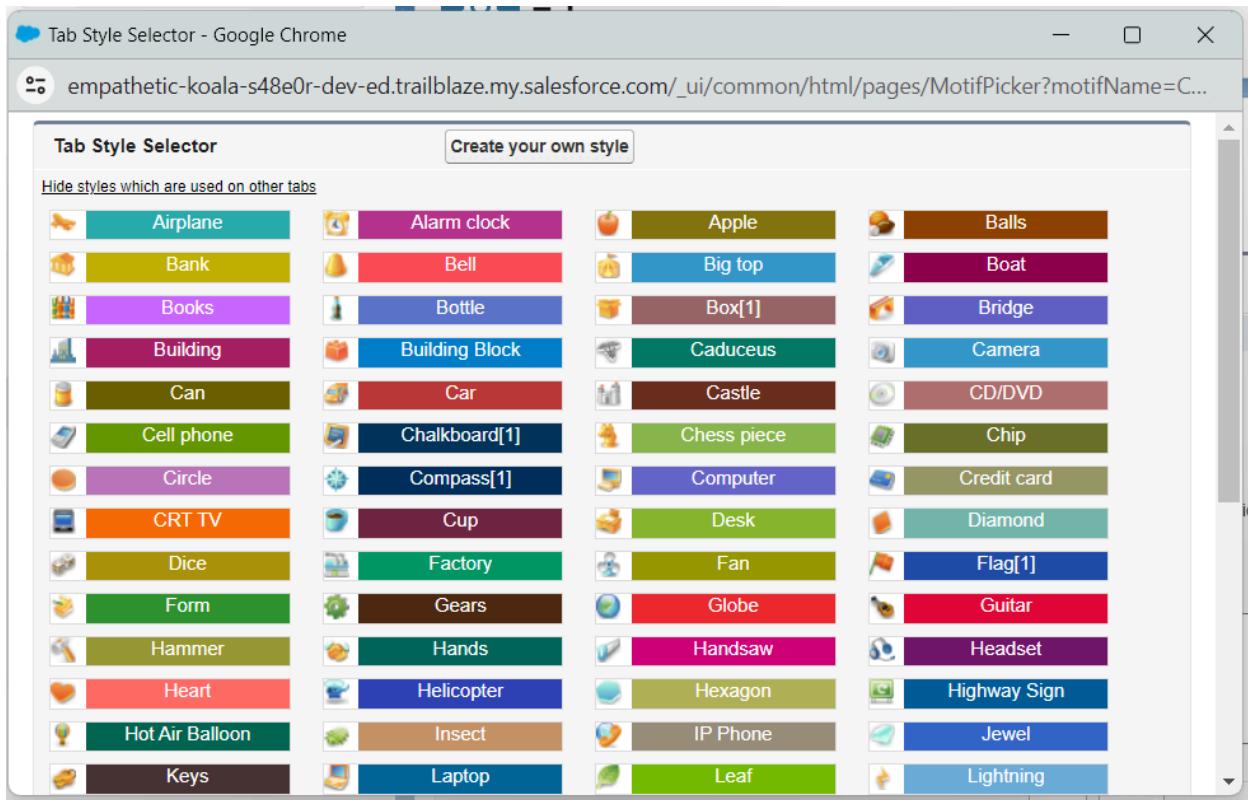
1. To Create the tab → click on the gear Icon → then click Home button →In the quick find box search for the Tabs.

The screenshot shows the Salesforce Setup interface under the 'Tabs' section. At the top, there is a header with a gear icon labeled 'SETUP' and the word 'Tabs'. Below the header, the title 'Custom Tabs' is displayed. A sub-header states: 'You can create new custom tabs to extend Salesforce functionality or to build new application functionality.' A detailed description follows: 'Custom Object tabs look and behave like the standard tabs provided with Salesforce. Web tabs allow you to embed external web applications and content within the Salesforce window. Visualforce tabs allow you to embed Visualforce pages. Lightning Component tabs allow you to add Lightning components to the navigation menu in Lightning Experience and the mobile app. Lightning Page tabs allow you to add Lightning Pages to Lightning Experience and the mobile app.' The main content area is divided into several sections: 'Custom Object Tabs', 'Web Tabs', 'Visualforce Tabs', and 'Lightning Component Tabs'. The 'Custom Object Tabs' section contains three entries:

Action	Label	Tab Style	Description
Edit Del	Customer Accounts	People	
Edit Del	Delivery Personnel	Presenter	
Edit Del	Parcel's	Box	

The 'Web Tabs' section shows: 'No Web Tabs have been defined'. The 'Visualforce Tabs' section shows: 'No Visualforce Tabs have been defined'. The 'Lightning Component Tabs' section shows: 'No Lightning Component Tabs have been defined'.

2. Then Click on the new button to create tab, in the Tab style you can select whatever you want to select



3. Than select the Customer Account Object and select the Icon from the search button and click next next and save

For creating the Remaining tabs follow the above steps

- For creating a tab for Parcel Value Tab follow the above steps which are being created in step1 for customer account.

- b. For creating a tab for Delivery Personnel Tab follow the above steps which are being created in step1 for customer account.

Milestone 4:Fields & Relationships

Fields represent the data stored in the columns of a relational database. It can also hold any valuable information that you require for a specific object. Hence, the overall searching, deletion, and editing of the records become simpler and quicker.

Types of Fields

- Standard Fields
- Custom Fields

Standard Fields:

As the name suggests, the Standard Fields are the predefined fields in Salesforce that perform a standard task. The main point is that you can't simply delete a Standard Field until it is a non-required standard field. They are

- Created By
- Owner
- Last Modified

Custom Fields:

Custom Fields are highly flexible, and users can change them according to requirements. Moreover, each organizer or company can use them if necessary. It means you need not always include them in the records, unlike Standard fields. Hence, the final decision depends on the user, and he can add/remove Custom Fields of any given form.

Activity -1

1. Go to setup → click on Object Manager → type object name in search bar → click on the object→ "Customer Account" → Field and Relationship→ then click on new →Field Data Type(Text)

SETUP > OBJECT MANAGER
Customer Account

Fields & Relationships

- Date
- Date/Time
- Email
- Geolocation
- Number
- Percent
- Phone
- Picklist
- Picklist (Multi-Select)
- Text**
- Text Area
- Text Area (Long)
- Text Area (Rich)
- Text (Encrypted) (1)
- Time
- URL

Allows users to enter a date or pick a date from a popup calendar.
Allows users to enter a date and time, or pick a date from a popup calendar. When users click a date in the pop-up, that date and the current time are entered into the Date/Time field.
Allows users to enter an email address, which is validated to ensure proper format. If this field is specified for a contact or lead, users can choose the address when clicking Send an Email. Note that custom email addresses cannot be used for mass emails.
Allows users to define locations. Includes latitude and longitude components, and can be used to calculate distance.
Allows users to enter any number. Leading zeros are removed.
Allows users to enter a percentage number, for example, '10' and automatically adds the percent sign to the number.
Allows users to enter any phone number. Automatically formats it as a phone number.
Allows users to select a value from a list you define.
Allows users to select multiple values from a list you define.
Allows users to enter any combination of letters and numbers.
Allows users to enter up to 255 characters on separate lines.
Allows users to enter up to 131,072 characters on separate lines.
Allows users to enter formatted text, add images and links. Up to 131,072 characters on separate lines.
Allows users to enter any combination of letters and numbers and store them in encrypted form.
Allows users to enter a local time. For example, "2:40 PM", "14:40", "14:40:00", and "14:40 50 600" are all valid times for this field.
Allows users to enter any valid website address. When users click on the field, the URL will open in a separate browser window.

Next Cancel

2. Click On Next → Field Label → Customer Name → next → next → Save. Always require a value in this field in order to save a record.

Validation Rules (0)

Custom Field Definition Detail

Edit Set Field-Level Security View Field Accessibility Where is this used?

Field Information

Field Label	Customer Name	Object Name	Cust
Field Name	Customer_Name	Data Type	Text
API Name	Customer_Name__c		
Description	Name of the customer		
Help Text			
Data Owner			
Field Usage			
Data Sensitivity Level			
Compliance Categorization			
Created By	Sanjana Tunk, 22/10/2023, 3:10 pm	Modified By	Sanj.

General Options

Required	<input checked="" type="checkbox"/>
Unique	<input type="checkbox"/>
Case Sensitive	<input type="checkbox"/>
External ID	<input type="checkbox"/>
Default Value	

Text Options

Length	40
--------	----

3. Follow the above steps and create the Remaining Fields.

- Customer Contact (Phone) - Always require a value in this field in order to save a record
- Customer Email Id(Email) - Always require a value in this field in order to save a record
- Password (Password) - Make sure to check unique checkbox-(Treat "ABC" and "abc" as duplicate values (case insensitive)) - Length is 30.
- Customer Address (Geolocation) - Decimal Places 15 - Always require a value in this field in order to save a record.

4. Click On Save & New → Lookup Relationship → Select Parcel from related to list → next → Field label Parcel → Save.

Activity -2

1. Go to setup → click on Object Manager → type object name in search bar → click on the object→ “Parcel” → Field and Relationship→ then click on new →Field Data Type(Text)

The screenshot shows the Salesforce Object Manager interface for the 'Parcel' object. The 'Fields & Relationships' tab is active. On the left, a sidebar lists various configuration options like Details, Page Layouts, and Record Types. On the right, a list of field types is displayed with their descriptions:

- Date
- Date/Time
- Email
- Geolocation
- Number
- Percent
- Phone
- Picklist
- Picklist (Multi-Select)
- Text** (selected)
- Text Area
- Text Area (Long)

Descriptions for each field type are provided on the right side of the list.

2. Click On Next→ Field Label→ Sender Name→ next→next→Save. Always require a value in this field in order to save a record.

The screenshot shows the configuration of the 'Sender Name' field for the 'Parcel' object. The 'General Options' section is active. The 'Required' checkbox is checked, indicating that a value is always required. Other settings include:

- Data Owner: User
- Field Usage: --None--
- Data Sensitivity Level: --None--
- Compliance Categorization: Available (PII, HIPAA, GDPR, PCI) and Chosen (Selected)
- General Options:
 - Required: Always require a value in this field in order to save a record
 - Unique: Do not allow duplicate values
 - Treat "ABC" and "abc" as duplicate values (case insensitive)
 - Treat "ABC" and "abc" as different values (case sensitive)
 - External ID: Set this field as the unique record identifier from an external system
 - Default Value: Show Formula Editor
- Text Options:
 - Length: 40

3. Follow the above steps and create the Remaining Fields.

- Receiver Name (Text) - Always require a value in this field in order to save a record
- Size of Parcel (Number) - Always require a value in this field in order to save a record decimal places (3,3)
- Parcel Weight (Number) - Always require a value in this field in order to save a record decimal places (6,3)
- Tracking Number (Number) - Make sure to check unique checkbox -Always require a value in this field in order to save a record - Length is 6.

- Delivery Address (Geolocation) - Decimal Places 15 - Always require a value in this field in order to save a record.
- Parcel Id (Text) - Make sure to check unique checkbox (case insensitive) -Always require a value in this field in order to save a record - Length is 16.
- Delivery Date (Date) - Always require a value in this field in order to save a record.
- Pickup Date (Date) - Always require a value in this field in order to save a record.
- Parcel Status (Picklist) - Always require a value in this field in order to save a record in a separate line to be entered - In Transit: The parcel is currently being shipped or transported.
Out for Delivery: The parcel is out for delivery and is expected to be delivered to the recipient.
Delivered: The parcel has been successfully delivered to the recipient.
On Hold: The delivery of the parcel has been temporarily delayed or put on hold.
Scheduled for Pickup: The parcel is scheduled to be picked up by the carrier.
Failed Delivery Attempt: An attempt to deliver the parcel was unsuccessful.
Returned: The parcel has been returned to the sender or a return has been initiated.

4. Click On Save & New→ Lookup Relationship → Select Delivery Personnel from related to list→ next→Field label Delivery Personnel→Save.

Activity -3

1. Go to setup → click on Object Manager → type object name in search bar → click on the object→ “Delivery Personnel” → Field and Relationship→ then click on new →Field Data Type(Text)

2. Click On Next→ Field Label→ Delivery Personnel Name→ next→next→Save. Always require a value in this field in order to save a record.

3. Follow the above steps and create the Remaining Fields.

- Contact Number (Phone) - Always require a value in this field in order to save a record
- Email Id>Email) - Always require a value in this field in order to save a record
- Route (Geolocation) - Decimal Places 15 - Always require a value in this field in order to save a record.

4. Click On Save & New→ Lookup Relationship → Select Parcel from related to list→ next→Field label Parcel→Save.

SETUP > OBJECT MANAGER

Delivery Personnel

Fields & Relationships

Geolocation

Allows users to define locations. Includes latitude and longitude components, and can be used to calculate distance.

Number

Allows users to enter any number. Leading zeros are removed.

Percent

Allows users to enter a percentage number, for example, '10' and automatically adds the percent sign to the number.

Phone

Allows users to enter any phone number. Automatically formats it as a phone number.

Picklist

Allows users to select a value from a list you define.

Picklist (Multi-Select)

Allows users to select multiple values from a list you define.

Text

Allows users to enter any combination of letters and numbers.

Text Area

Allows users to enter up to 255 characters on separate lines.

Text Area (Long)

Allows users to enter up to 131,072 characters on separate lines.

Text Area (Rich)

Allows users to enter formatted text, add images and links. Up to 131,072 characters on separate lines.

Text (Encrypted) i

Allows users to enter any combination of letters and numbers and store them in encrypted form.

Time

Allows users to enter a local time. For example, "2:40 PM", "14:40", "14:40:00", and "14:40:50:600" are all valid times for this field.

URL

Allows users to enter any valid website address. When users click on the field, the URL will open in a separate browser window.

[Next](#) [Cancel](#)

SETUP > OBJECT MANAGER

Delivery Personnel

Fields & Relationships

Delivery Personnel Custom Field

Delivery Personnel Name

[Back to Delivery Personnel](#)

[Validation Rules \[0\]](#)

Custom Field Definition
[Edit](#)
[Set Field-Level Security](#)
[View](#)

[Where is this used?](#)

Field Information

Field Label	Delivery Personnel Name	Object Name
Field Name	Delivery_Personnel_Name	Data Type
API Name	Delivery_Personnel_Name__c	
Description		
Help Text		
Data Owner		
Field Usage		
Data Sensitivity Level		
Compliance Categorization		
Created By	Sanjana Tunk , 22/10/2023, 3:20 pm	Modified By

General Options

Required	<input checked="" type="checkbox"/>
----------	-------------------------------------

Milestone 5 : Create App

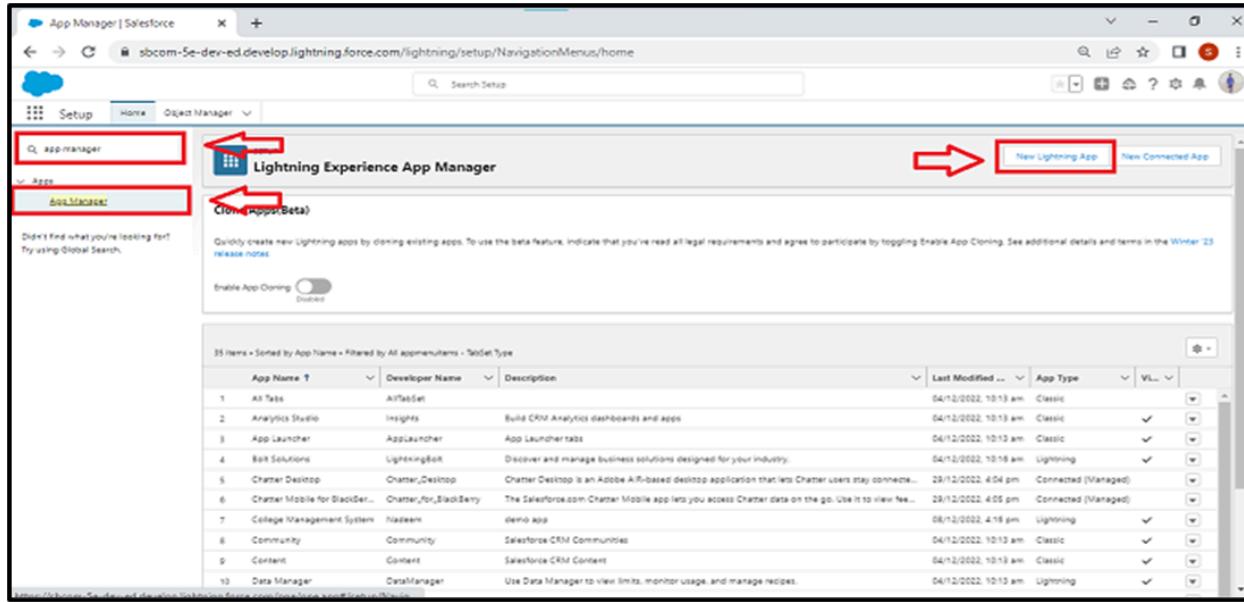
Creating a Parcel Dispatcher app involves designing a solution that helps manage and track the dispatch and delivery of parcels efficiently. The app is built in Lightning Experience , depending on your organization's preferences and requirements. Below is a brief overview of how you might design such an app :

- Design a Lightning App using the Lightning App Builder.
- Include tabs for the Parcel object, standard objects like Parcel and Delivery Personnel, and custom Lightning components.
- Customize the app's color scheme and logo to give it a branded look.

Activity -1

Create the Lightning App

1.Go to the Gear Icon → Click on setup → click on the Home Button → Go to the quick find box and select the App Manager

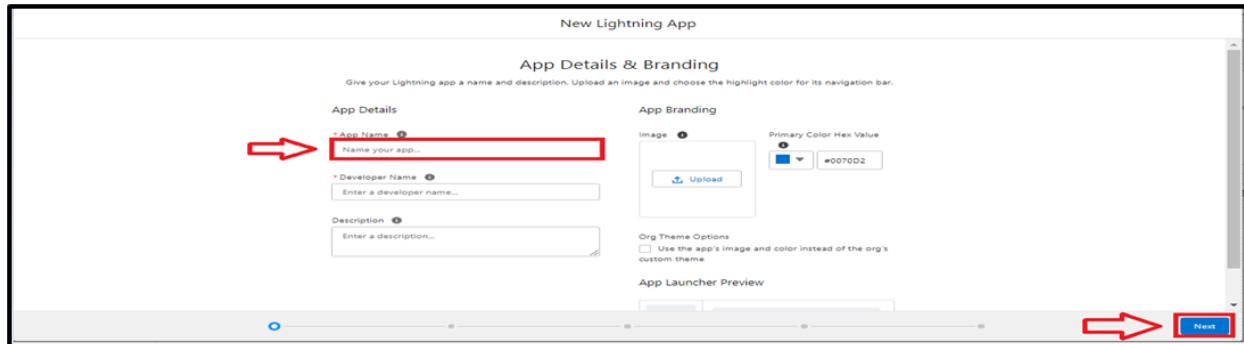


The screenshot shows the Salesforce App Manager interface. At the top, there are navigation tabs: Setup, Home, and Object Manager. Below these are search and filter tools, including a 'Search Setup' bar and a 'New Lightning App' button. A red box highlights the 'New Lightning App' button. On the left, there are sidebar links for 'App Manager' (highlighted with a red box), 'My Apps', and 'App Manager'. A red arrow points from the 'App Manager' link to the 'Cloud Apps(Beta)' section. Another red box highlights the 'Cloud Apps(Beta)' link. The main area displays a table of existing apps, with a red box highlighting the first row. The table columns include App Name, Developer Name, Description, Last Modified, App Type, and Version. The table shows 35 items, sorted by App Name, filtered by All app types, and set to English.

App Name	Developer Name	Description	Last Modified	App Type	Ver.
1 All Tabs	AlTabSet		04/12/2022, 10:13 am	Classic	
2 Analytics Studio	Insights	Build CRM Analytics dashboards and apps	04/12/2022, 10:13 am	Classic	
3 App Launcher	AppLauncher	App Launcher tabs	04/12/2022, 10:13 am	Classic	
4 Built Solutions	LightningSolt	Discover and manage business solutions designed for your industry.	04/12/2022, 10:13 am	Lightning	
5 Chatter Desktop	Chatter/Desktop	Chatter Desktop is an Adobe AIR-based desktop application that lets Chatter users stay connected...	29/12/2022, 4:04 pm	Connected (Managed)	
6 Chatter Mobile for BlackBerry	Chatter_for BlackBerry	The Salesforce.com Chatter Mobile app lets you access Chatter data on the go. Use it to view fe...	29/12/2022, 4:05 pm	Connected (Managed)	
7 College Management System	Nadeem	demo app	08/12/2022, 4:16 pm	Lightning	
8 Community	Community	Salesforce CRM Communities	04/12/2022, 10:13 am	Classic	
9 Content	Content	Salesforce CRM Content	04/12/2022, 10:13 am	Classic	
10 Data Manager	DataManager	Use Data Manager to view limits, monitor usage, and manage recipes.	04/12/2022, 10:13 am	Lightning	

2.Fill the app name as an in app details and branding →Next → (App option page) keep it as default → Next

3. Utility Items keep it as default → Next → (Add Navigation Items)(add tabs Customer Accounts, Parcel's Delivery Personnel) → Next → (Add User Profile) Add System Administrator, Salesforce platform user, Standard User → Next.

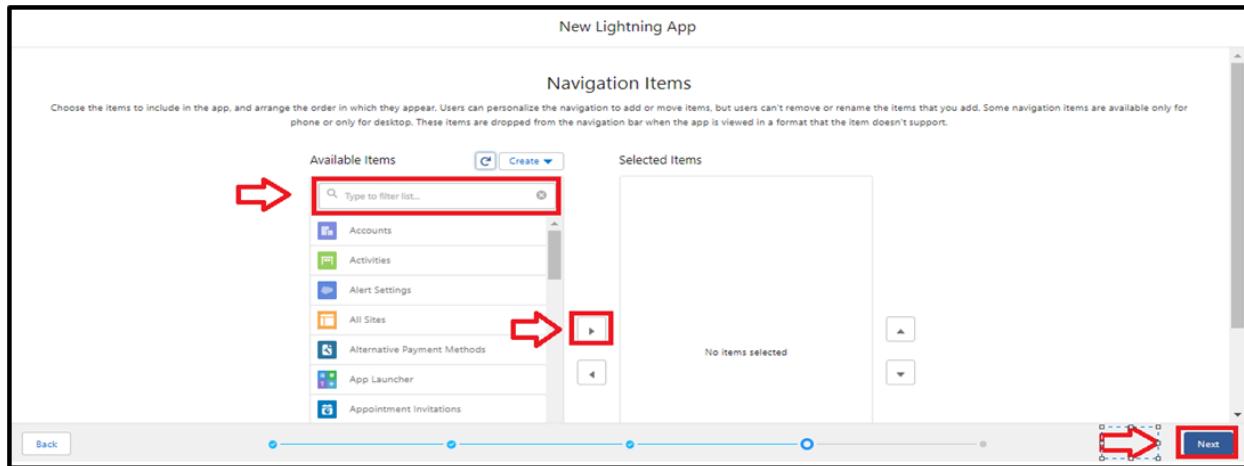


4. To Add Navigation Items:

Select the items from the search bar and move it using the arrow button → Next. select all the tabs which you have created

5. To Add User Profiles:

Search profiles in search bar → click on the arrow button & select Standard user, standard Platform user & System Admin Profile→ save & finish.



Milestone 6: Profile

A profile is a group/collection of settings and permissions that define what a user can do in salesforce. Profile controls “Object permissions, Field permissions, User permissions, Tab settings, App settings, Apex class access, Visualforce page access, Page layouts, Record Types, Login hours & Login IP ranges.

You can define profiles by the user's job function. For example System Administrator, Developer, Sales Representative.

Custom Profiles:

- Custom ones defined by us i.e; Delivery Personnel, Customer etc.
- They can be deleted if there are no users assigned with that particular one

Activity-1

To create a new profile:

1. Go to setup → type profiles in quick find box → click on profiles → clone the desired profile (standard platform user is pref) and clone that profile

The screenshot shows the Salesforce Setup interface with the following details:

- Search Bar:** Contains the text "profile".
- Setup Header:** Shows "SETUP" and "Profiles".
- Left Navigation:** Under "Users", the "Profiles" tab is selected and highlighted with a red box, labeled "2".
- Page Title:** "Profiles".
- Actions:** "All Profiles" dropdown, "Edit | Delete | Create New View" buttons.
- Table:** A list of profiles with columns: Action, Profile Name, User License, and Custom. The table includes rows for Sales Manager, Salesforce API Only System Integrations, School profile, Silver Partner User, Solution Manager, Standard Platform User (highlighted with a red box, labeled "3"), Standard User, and System Administrator.
- Page Bottom:** Pagination and search controls.

2. Enter a Profile Name(Delivery Personnel) And click on Save
3. Click on the new created profile
4. While still on the profile page, then click Edit.

Profile
Delivery Personnel

Users with this profile have the permissions and page layouts listed below. Administrators can change a user's profile by editing that user's personal information.

If your organization uses Record Types, use the Edit links in the Record Type Settings section below to make one or more record types available to users with this profile.

[Login IP Ranges](#) | [Enabled Apex Class Access](#) | [Enabled Visualforce Page Access](#) | [Enabled External Data Source Access](#) | [Enabled Named Credential Access](#) | [Enabled External Credential Principal Access](#) | [Enabled Custom Metadata Type Access](#) | [Enabled Custom Setting Definitions Access](#) | [Enabled Flow Access](#) | [Enabled Service Presence Status Access](#) | [Enabled Custom Permissions](#)

Profile Detail		Edit	Clone	Delete	View Users
Name	Delivery Personnel				
User License	Salesforce Platform				Custom Profile <input checked="" type="checkbox"/>
Description					
Created By	Sanjana Tunk, 25/10/2023, 2:22 pm				Modified By salesforce.com, inc., 26/10/2023, 3:20 pm

Page Layouts

Standard Object Layouts							
Global	Global Layout [View Assignment]			Data Use Legal Basis	Data Use Legal Basis Layout [View Assignment]		
Email Application	Not Assigned [View Assignment]			Data Use Purpose	Data Use Purpose Layout [View Assignment]		
Home Page Layout	Home Page Default [View Assignment]			Email Message	Email Message Layout [View Assignment]		
Access	Access Layout [View Assignment]			Event	Event Layout [View Assignment]		
Account	Account Layout [View Assignment]			Feed Item	Feed Item Layout [View Assignment]		
Asset	Asset Layout [View Assignment]			Idea	Varies by Record Type		

5. For the Delivery Personnel profile give the following access and save it.

Custom Object Permissions

	Basic Access						Data Administration					
	Read	Create	Edit	Delete	View All	Modify All	Read	Create	Edit	Delete	View All	Modify All
Customer Accounts	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Delivery Personnel	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>						

Session Settings

Session Times Out After: 2 hours of inactivity

Session Security Level Required at Login: None

Enable different Experience Cloud login policies for employees:

- Separate Experience Cloud site and Salesforce login authentication for employees:
- Relax login IP restrictions:
- Skip employee device activation during Experience Cloud site login:
- Allow OAuth for employees:

Password Policies

User passwords expire in: 90 days

Enforce password history: 3 passwords remembered

Minimum password length: 8

Password complexity requirement: Must include alpha and numeric characters

Password question requirement: Cannot contain password

6. Similarly clone the system admin profile and give the name as Dispatch Manager And give the access to the as of Dispatch manager as the follows

SETUP Profiles

Custom Object Permissions

	Basic Access						Data Administration						
	Read	Create	Edit	Delete	View All	Modify All		Read	Create	Edit	Delete	View All	Modify All
Customer Accounts	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Parcel's	<input checked="" type="checkbox"/>					
Delivery Personnel	<input checked="" type="checkbox"/>												

Session Settings

Session Times Out After: 2 hours of inactivity

Session Security Level Required at Login: -None--

Enable different Experience Cloud login policies for employees:

- Separate Experience Cloud site and Salesforce login authentication for employees.
- Relax login IP restrictions
- Skip employee device activation during Experience Cloud site login
- Allow OAuth for employees

Password Policies

User passwords expire in: 90 days

Enforce password history: 3 passwords remembered

Minimum password length: 8

Password complexity requirement: Must include alpha and numeric characters

Password question requirement: Cannot contain password

7. Similarly clone the standard platform profile and give the name as Customer Account And give the access to the as of Customer Account as the follows

SETUP Profiles

Custom Object Permissions

	Basic Access						Data Administration						
	Read	Create	Edit	Delete	View All	Modify All		Read	Create	Edit	Delete	View All	Modify All
Customer Accounts	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Parcel's	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Delivery Personnel	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>							

Session Settings

Session Times Out After: 2 hours of inactivity

Session Security Level Required at Login: -None--

Enable different Experience Cloud login policies for employees:

- Separate Experience Cloud site and Salesforce login authentication for employees.
- Relax login IP restrictions
- Skip employee device activation during Experience Cloud site login
- Allow OAuth for employees

Password Policies

User passwords expire in: 90 days

Enforce password history: 3 passwords remembered

Minimum password length: 8

Password complexity requirement: Must include alpha and numeric characters

Password question requirement: Cannot contain password

Milestone-7: User Adoption

As a new administrator, you perform user management tasks like creating and editing users, resetting passwords, granting permissions, configuring data access, and much more. In this unit, you will learn about users and how you add users to your Salesforce org.

Every user in Salesforce has a user account. The user account identifies the user, and the user account settings determine what features and records the user can access. Each user account contains at least the following:

- Username
- Email Address
- User's First Name (optional)
- User's Last Name
- Alias
- Nickname
- License
- Profile
- Role (optional)

Now create the users with 3 profiles which we have created.

Activity - 1

User1 Dispatch Manager

1. Go to the Gear Icon → Click the setup → click on the home button and in the quick find box search for the user → Click on the user → then click on new → fill the fields.

User Edit
User1 Manger Dispatch

Help for this Page 

User Edit Save Save & New Cancel

General Information Required Information

First Name	User1 Manger	Role	Customer Support, International
Last Name	Dispatch	User License	Salesforce
Alias	udisp	Profile	Dispatch Manager
Email	disuser@gmail.com	Active	<input checked="" type="checkbox"/>
Username	dispatch@manager.com	Marketing User	<input type="checkbox"/>
Nickname	User169822333375775899	Offline User	<input type="checkbox"/>
Title	Dispatch Manager	Knowledge User	<input type="checkbox"/>
Company	Dispatch tracing	Flow User	<input type="checkbox"/>
Department	Sales	Service Cloud User	<input type="checkbox"/>
Division		Site.com Contributor User	<input type="checkbox"/>
		Site.com Publisher User	<input type="checkbox"/>
		WDC User	<input type="checkbox"/>
		Data.com User Type	—None—
		Data.com Monthly Addition Limit	300
		Accessibility Mode (Classic Only)	<input type="checkbox"/>
		High-Contrast Palette on Charts	<input type="checkbox"/>

Activity - 2

User2 Delivery Personnel

1.Go to the Gear Icon→ Click the setup→click on the home button and in the quick find box search for the user →Click on the user→ then click on new →fill the fields.

The screenshot shows the 'User Edit' page for 'User2 Delivery Personnel'. The top navigation bar includes a blue gear icon, 'SETUP', and 'Users'. The main title is 'User Edit' and the sub-title is 'User2 Delivery Personnel Delivery'. On the right, there's a 'Help for this Page' link with a question mark icon. The page has a header 'General Information' with a note '= Required Information'. It contains two columns of input fields:

Field	Value
First Name	User2 Delivery Personnel
Last Name	Delivery
Alias	Dperson1
Email	sanjanatunk@gmail.com
Username	dpersonnel@gmail.com
Nickname	DeliveryPersonnel
Title	Delivery
Company	Dispatcher Tracing
Department	Sales
Division	
Role	Customer Support, International
User License	Salesforce Platform
Profile	Delivery Personnel
Active	<input checked="" type="checkbox"/>
Marketing User	<input type="checkbox"/>
Offline User	<input type="checkbox"/>
Knowledge User	<input type="checkbox"/>
Flow User	<input type="checkbox"/>
Service Cloud User	<input type="checkbox"/>
Site.com Contributor User	<input type="checkbox"/>
Site.com Publisher User	<input type="checkbox"/>
WDC User	<input type="checkbox"/>
Data.com User Type	--None--
Data.com Monthly Addition Limit	300
Accessibility Mode (Classic Only)	<input type="checkbox"/>
High-Contrast Palette on Charts	<input type="checkbox"/>

Activity - 3

User3 Customer Account

1.Go to the Gear Icon→ Click the setup→click on the home button and in the quick find box search for the user →Click on the user→ then click on new →fill the fields.

User Edit
User3 Customer Accounts

User Edit Save Save & New Cancel

General Information

First Name	User3 Customer	Role	Customer Support, International
Last Name	Accounts	User License	Salesforce Platform
Alias	uacco	Profile	Customer Account
Email	sanjanatunk@gmail.com	Active	<input checked="" type="checkbox"/>
Username	customers1@gmail.com	Marketing User	<input type="checkbox"/>
Nickname	Customer1	Offline User	<input type="checkbox"/>
Title	Customers	Knowledge User	<input type="checkbox"/>
Company	Dispatcher Tracing	Flow User	<input type="checkbox"/>
Department	Sales	Service Cloud User	<input type="checkbox"/>
Division		Site.com Contributor User	<input type="checkbox"/>
		Site.com Publisher User	<input type="checkbox"/>
		WDC User	<input type="checkbox"/>
		Data.com User Type	--None--
		Data.com Monthly Addition Limit	300
		Accessibility Mode (Classic Only)	<input type="checkbox"/>
		High-Contrast Palette on Charts	<input type="checkbox"/>

Milestone 8-OWD(Organization Wide Default)

OWD in a Parcel Dispatcher project is critical for establishing a secure baseline of data access. They help ensure that sensitive parcel information is appropriately protected, and access is granted based on organizational needs and compliance requirements.

Primarily, there are four levels of access that can be set in Salesforce OWD and they are

- Public Read/Write/Transfer (only available of Enquiry and Cases)
- Public Read/Write
- Public Read/Only
- Private

With parcel-related data stored in a custom object (e.g., "Parcels"), setting the OWD to Private ensures that users can only access the records they own or have been granted access to explicitly.

If sender and recipient information is stored in standard or custom objects, OWDs help control access.

Depending on the business requirements, you may set different OWDs for objects or fields related to tracking and delivery status.

Activity-1

Create OWD Setting

1. Setup, use the Quick Find box to find Sharing Settings.
2. Click Edit in the Organization-Wide Defaults area.
3. For each object, select the default access you want to give everyone.
4. For Every custom object give private as a record level security

The screenshot shows the 'Sharing Settings' page in the Salesforce Setup. The left sidebar has a search bar and navigation links for Security, Sharing Settings, and Help. The main content area is titled 'Sharing Settings' with a sub-header 'Manage sharing settings for: All Objects'. It includes a 'Default Sharing Settings' section and a 'Disable External Sharing Model' button. The 'Organization-Wide Defaults' table lists sharing settings for various objects:

Object	Default Internal Access	Default External Access	Grant Access Using Hierarchies
Lead	Public: ReadWrite/Transfer	Private	<input checked="" type="checkbox"/>
Account and Contract	Public: ReadWrite	Private	<input checked="" type="checkbox"/>
Contact	Controlled by Parent	Controlled by Parent	<input checked="" type="checkbox"/>
Order	Controlled by Parent	Controlled by Parent	<input checked="" type="checkbox"/>
Asset	Controlled by Parent	Controlled by Parent	<input checked="" type="checkbox"/>
Opportunity	Public: ReadWrite	Private	<input checked="" type="checkbox"/>
Case	Public: ReadWrite/Transfer	Private	<input checked="" type="checkbox"/>
Campaign	Public: Full Access	Private	<input checked="" type="checkbox"/>
Campaign Member	Controlled by Campaign	Controlled by Campaign	<input checked="" type="checkbox"/>
User	Public: Read Only	Private	<input checked="" type="checkbox"/>

Milestone 9-Reports

Reports give you access to your Salesforce data. You can examine your Salesforce data in almost infinite combinations, display it in easy-to-understand formats, and share the resulting insights with others. Before building, reading, and sharing reports, review these reporting basics.

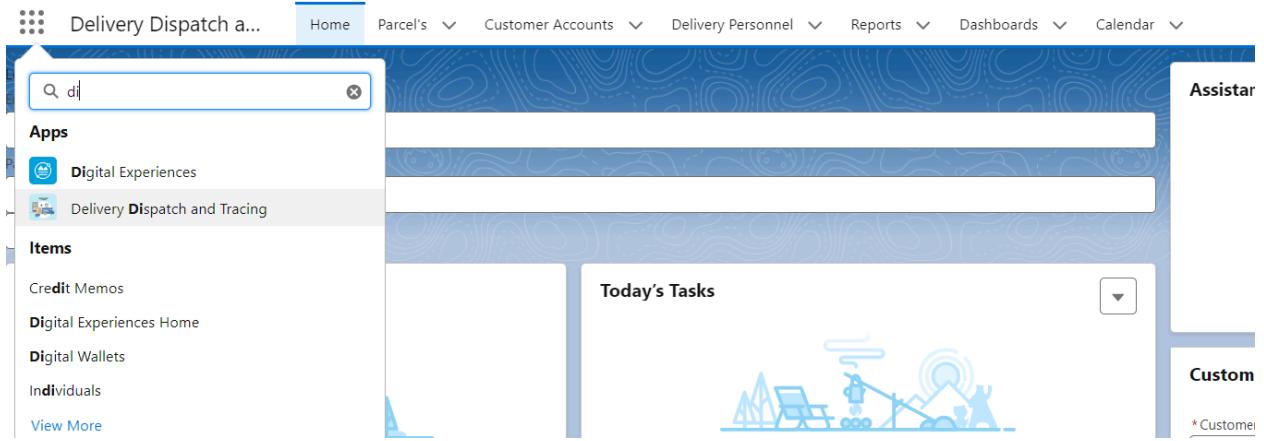
Types of Reports in Salesforce

- Tabular
- Summary
- Matrix
- Joined Reports

Activity-1

Create the Report

1. Go to the gear icon → click on the setup→click on the home button→ in the quick find box search for app manager select the App and go to the navigation→ There select the Report and include in the app → than go to App There you will find the Tab



2. Click create folder → Public Reports → New Report → Create the Report → Select the Parcel's Report → columns → Parcel Name → Receivers Name.

Select Start Report:

3. For to the reports and select the → Parcel's

Report Type Name	Category
Content Report	Standard
Library and User Report	Standard
File and Content Report	Standard
Individual	Standard
Individual History	Standard
Parcel's	Standard
Parcel's with Customer Account	Standard
Parcel's with Delivery Personnel	Standard
Parcel History	Standard
Customer Accounts	Standard
Customer Accounts with Parcel	Standard

4. Select the following option for the Rows and columns.

The screenshot shows the Salesforce Report Builder interface. The report is titled "Parcel's Report" and is set to run for the "Parcel's" object. The report outline includes fields for "Parcel ID", "Sender Name", "Delivery Date", and "Subtotal". The report preview shows data for two parcels: one from Bharathi and one from Neeti, both marked as "In Transit". The "Out for Delivery" column shows the expected delivery date as 24/10/2023. The report also includes a "Total" row. Reporting options at the bottom include "Row Counts", "Detail Rows", "Subtotals", "Grand Total", and "Stacked Summaries".

Parcel ID	Sender Name	Delivery Date	Subtotal	Out for Delivery	Subtotal	Total
a005j00000S2dR0	Bharathi	Record Count	1	28/10/2023	1	1
		Subtotal	1		0	1
a005j00000S2dVq	Neeti	Record Count	0	24/10/2023	1	1
		Subtotal	0		1	1
		Total	1		1	2
		Record Count	1		1	1

Milestone 10: Apex Trigger

Apex can be invoked by using triggers. Apex triggers enable you to perform custom actions before or after changes to Salesforce records, such as insertions, updates, or deletions.

A trigger is Apex code that executes before or after the following types of operations:

- insert
- update
- delete
- merge
- upsert
- undelete

Apex triggers, handlers, and a scheduler class are used to manage and notify users about changes in the status of parcels in a Parcel Dispatcher project.

There are primarily two types of Apex Triggers:

After Trigger: This type of trigger in Salesforce is used to access the field values set by the system and affect any change in the record. In other words, the after trigger makes changes to the value from the data inserted in some other record.

Event Triggered: after update - This trigger fires after records of the Parcel_c custom object are updated.

Record Update:

- When a Parcel_c record is updated, the trigger (ParcelStatusUpdate) fires.
- The trigger checks for changes in the ParcelStatus_c field.

Notification Handling:

- If the status meets certain criteria, the trigger adds the parcel to parcelsToUpdate and sends email notifications using the handler (ParcelNotificationClass).

Field Update:

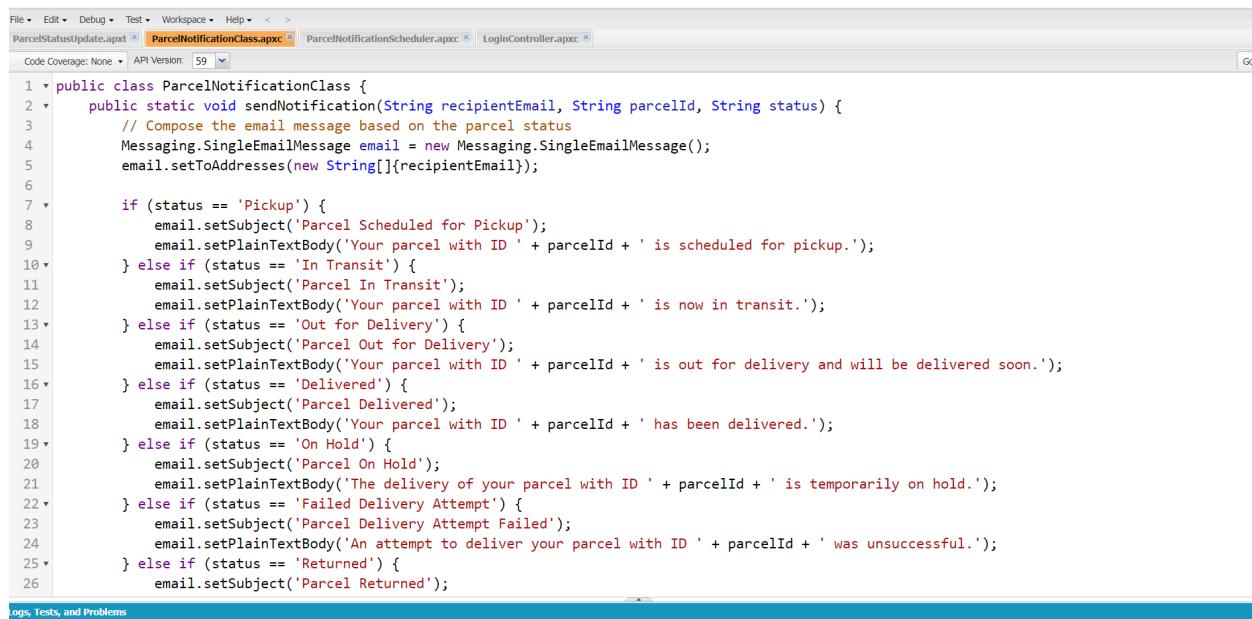
- The trigger updates the ParcelStatus__c field for certain statuses.

Scheduled Notification:

- The scheduler class (ParcelNotificationScheduler) simulates sending a notification for a delivered parcel every weekday at 8:00 AM.

Activity- 1

Defining the logic for automating actions based on trigger events, Apex class to handle various parcel stages, including "Pickup," "In Transit," "Out for Delivery," "Delivered," "On Hold," "Failed Delivery Attempt," and "Returned." It includes methods for sending notifications for these stages



```
File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < >
ParcelStatusUpdate.apxc ParcelNotificationClass.apxc ParcelNotificationScheduler.apxc LoginController.apxc
Code Coverage: None API Version: 59 Go
1  public class ParcelNotificationClass {
2    public static void sendNotification(String recipientEmail, String parcelId, String status) {
3      // Compose the email message based on the parcel status
4      Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
5      email.setToAddresses(new String[]{recipientEmail});
6
7      if (status == 'Pickup') {
8        email.setSubject('Parcel Scheduled for Pickup');
9        email.setPlainTextBody('Your parcel with ID ' + parcelId + ' is scheduled for pickup.');
10     } else if (status == 'In Transit') {
11       email.setSubject('Parcel In Transit');
12       email.setPlainTextBody('Your parcel with ID ' + parcelId + ' is now in transit.');
13     } else if (status == 'Out for Delivery') {
14       email.setSubject('Parcel Out for Delivery');
15       email.setPlainTextBody('Your parcel with ID ' + parcelId + ' is out for delivery and will be delivered soon.');
16     } else if (status == 'Delivered') {
17       email.setSubject('Parcel Delivered');
18       email.setPlainTextBody('Your parcel with ID ' + parcelId + ' has been delivered.');
19     } else if (status == 'On Hold') {
20       email.setSubject('Parcel On Hold');
21       email.setPlainTextBody('The delivery of your parcel with ID ' + parcelId + ' is temporarily on hold.');
22     } else if (status == 'Failed Delivery Attempt') {
23       email.setSubject('Parcel Delivery Attempt Failed');
24       email.setPlainTextBody('An attempt to deliver your parcel with ID ' + parcelId + ' was unsuccessful.');
25     } else if (status == 'Returned') {
26       email.setSubject('Parcel Returned');
27     }
28   }
29 }
```

Apex Class :

```
public class ParcelNotificationClass {

    public static void sendNotification(String recipientEmail, String parcelId, String status)
    {
```

```
// Compose the email message based on the parcel status

Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();

email.setToAddresses(new String[]{recipientEmail});

if (status == 'Pickup') {

    email.setSubject('Parcel Scheduled for Pickup');

    email.setPlainTextBody('Your parcel with ID ' + parcelId + ' is scheduled for
pickup.');

} else if (status == 'In Transit') {

    email.setSubject('Parcel In Transit');

    email.setPlainTextBody('Your parcel with ID ' + parcelId + ' is now in transit.');

} else if (status == 'Out for Delivery') {

    email.setSubject('Parcel Out for Delivery');

    email.setPlainTextBody('Your parcel with ID ' + parcelId + ' is out for delivery and
will be delivered soon.');

} else if (status == 'Delivered') {

    email.setSubject('Parcel Delivered');

    email.setPlainTextBody('Your parcel with ID ' + parcelId + ' has been delivered.');

} else if (status == 'On Hold') {

    email.setSubject('Parcel On Hold');

    email.setPlainTextBody('The delivery of your parcel with ID ' + parcelId + ' is
temporarily on hold.');
```

```
    } else if (status == 'Failed Delivery Attempt') {

        email.setSubject('Parcel Delivery Attempt Failed');

        email.setPlainTextBody('An attempt to deliver your parcel with ID ' + parcelId + ' was unsuccessful.');

    } else if (status == 'Returned') {

        email.setSubject('Parcel Returned');

        email.setPlainTextBody('Your parcel with ID ' + parcelId + ' has been returned to the sender.');

    } else {

        // Handle other statuses or provide a default message if needed

        email.setSubject('Parcel Status Update');

        email.setPlainTextBody('Parcel ID ' + parcelId + ' has changed status to ' + status);

    }

    // Send the email

    Messaging.sendEmail(new Messaging.SingleEmailMessage[]{email});

}

}
```

You can schedule a job to execute this method.

The screenshot shows the Salesforce Apex code editor with the following details:

- File menu: File, Edit, Debug, Test, Workspace, Help.
- Toolbar: None, API Version (set to 59), Go To.
- Open files: ParcelStatusUpdate.apxc, ParcelNotificationClass.apxc, ParcelNotificationScheduler.apxc (active tab), LoginController.apxc.
- Code Coverage: None.
- Code:

```
1 public class ParcelNotificationScheduler implements Schedulable {
2     public void execute(SchedulableContext sc) {
3         // Call the method within ParcelNotificationClass
4         ParcelNotificationClass.sendNotification('recipient@example.com', 'Parcel123', 'Delivered');
5     }
6 }
```

Activity- 2

Use Case: This Trigger works to handle parcel updates and automate actions based on trigger events, This apex trigger changes the status of parcels from "Pickup" to "In Transit" and sends a notification when this change occurs.

Trigger:

The screenshot shows the Salesforce Apex code editor with the following details:

- File menu: File, Edit, Debug, Test, Workspace, Help.
- Toolbar: None, API Version (set to 59), Go To.
- Open files: ParcelStatusUpdate.apxc (active tab), ParcelNotificationClass.apxc, ParcelNotificationScheduler.apxc, LoginController.apxc.
- Code Coverage: None.
- Code:

```
1 trigger ParcelStatusUpdate on Parcel__c (after update) {
2     List<Parcel__c> parcelsToUpdate = new List<Parcel__c>();
3
4     for (Parcel__c parcel : Trigger.new) {
5         String oldStatus = Trigger.oldMap.get(parcel.Id).ParcelStatus__c;
6         String newStatus = parcel.ParcelStatus__c;
7
8         if (oldStatus != newStatus) {
9             String notificationMessage = 'Parcel ID ' + parcel.Id + ' has changed status from ' + oldStatus + ' to ' + newStatus;
10
11            if (newStatus == 'In Transit' || newStatus == 'Out for Delivery' || newStatus == 'Delivered' || newStatus == 'On Hold' || newStatus ==
12                parcelsToUpdate.add(parcel);
13
14            // Send a notification (you would implement this part according to your needs)
15            // Example: Sending an email
16            Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
17            email.setToAddresses(new String[] { 'recipient@example.com' });
18            email.setSubject('Parcel Status Update');
19            email.setPlainTextBody(notificationMessage);
20            Messaging.sendEmail(new Messaging.SingleEmailMessage[] { email });
21        }
22    }
23
24    // Update parcel status if needed
25    if (!parcelsToUpdate.isEmpty()) {
26        // Implementation for updating status
27    }
28 }
```

Trigger Code:

```
trigger ParcelStatusUpdate on Parcel__c (after update) {
    List<Parcel__c> parcelsToUpdate = new List<Parcel__c>();

    for (Parcel__c parcel : Trigger.new) {
        String oldStatus = Trigger.oldMap.get(parcel.Id).ParcelStatus__c;
        String newStatus = parcel.ParcelStatus__c;

        if (oldStatus != newStatus) {
            String notificationMessage = 'Parcel ID ' + parcel.Id + ' has changed status from ' + oldStatus +
                ' to ' + newStatus;

            if (newStatus == 'In Transit' || newStatus == 'Out for Delivery' || newStatus == 'Delivered' ||
                newStatus == 'On Hold' || newStatus == 'Scheduled for Pickup' || newStatus == 'Failed Delivery
                Attempt' || newStatus == 'Returned') {
                parcelsToUpdate.add(parcel);

                // Send a notification (you would implement this part according to your needs)
                // Example: Sending an email
                Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
                email.setToAddresses(new String[] { 'recipient@example.com' });
                email.setSubject('Parcel Status Update');
                email.setPlainTextBody(notificationMessage);
                Messaging.sendEmail(new Messaging.SingleEmailMessage[] { email });
            }
        }
    }

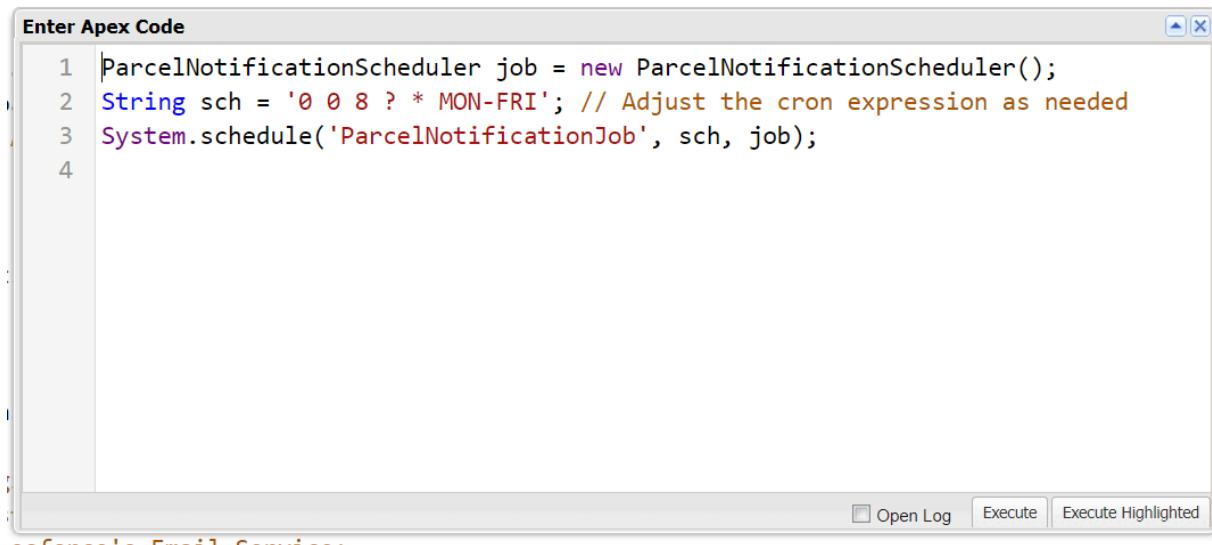
    // Update parcel status if needed
    if (!parcelsToUpdate.isEmpty()) {
        for (Parcel__c updatedParcel : parcelsToUpdate) {
            updatedParcel.ParcelStatus__c = updatedParcel.ParcelStatus__c;
        }
        update parcelsToUpdate;
    }
}
```

Activity- 3

Apex Scheduler Class :

```
public class ParcelNotificationScheduler implements Schedulable {  
    public void execute(SchedulableContext sc) {  
        // Call the method within ParcelNotificationClass  
        ParcelNotificationClass.sendNotification('recipient@example.com', 'Parcel123', 'Delivered');  
    }  
}
```

To schedule the job to run the method at specific times and frequencies, you need to use the Salesforce user interface or execute the scheduling code using anonymous Apex.



The screenshot shows a 'Enter Apex Code' window with the following code:

```
1 ParcelNotificationScheduler job = new ParcelNotificationScheduler();  
2 String sch = '0 0 8 ? * MON-FRI'; // Adjust the cron expression as needed  
3 System.schedule('ParcelNotificationJob', sch, job);  
4
```

At the bottom of the window, there are three buttons: 'Open Log', 'Execute', and 'Execute Highlighted'. A status bar at the bottom displays the message 'Salesforce Email Canvas'.

```
ParcelNotificationScheduler job = new ParcelNotificationScheduler();  
String sch = '0 0 8 ? * MON-FRI'; // Adjust the cron expression as needed  
System.schedule('ParcelNotificationJob', sch, job);
```

The above code schedules the "ParcelNotificationScheduler" class to run the execute method (which calls the sendNotification method in "ParcelNotificationClass") at 8 AM from Monday to Friday.

Milestone 11: Login Page Component (LWC)

Use Case upon successful login, the application displays a personalized dashboard with the following details for each parcel associated with the customer:

- Parcel ID
- Status
- Delivery Date
- Delivery Personnel Name
- Contact Number

Activity- 1

Open a Workspace:

Open Visual Studio Code and create a new workspace or open an existing one. A workspace is a directory that contains your Salesforce projects.

Create a New Salesforce Project:

Open the Command Palette (Ctrl + Shift + P) and run the command "SFDX: Create Project".

Choose the project type.

For LWC development, choose "Standard" for most cases.

Authorize an Org:

Open the Command Palette and run the command "SFDX: Authorize an Org".

Log in to your Salesforce org.

Create a New Lightning Web Component:

Open the Command Palette and run the command "SFDX: Create Lightning Web Component".

Enter a name for your component (Login) and choose the directory to save it.

Edit Your Lightning Web Component:

Open the newly created component in the force-app/main/default/lwc directory.

Edit the HTML, JavaScript, and CSS files as needed.

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure under "LWC". Key files include ".husky", ".sf", ".sfdx", "config", "force-app/main/default/applications", "aura/.eslintrc.json", "classes", "contentassets", "flexipages", "layouts", "lwc/login/_tests_.login.html", "lwc/login/login.js", "lwc/login/login.js-meta.xml", "tracker/.eslintrc.json", "objects", "permissionsets", "staticresources", "tabs", "triggers", "manifest", "scripts/apex", and ".OUTLINE", ".TIMELINE", and ".RUNNING TASKS".
- Code Editor (Right):** The active file is "login.js" (line 1). The code is as follows:

```
force-app/main/default/lwc/login>JS login.js > LoginPage > handlePasswordChange
1 import { LightningElement, track } from 'lwc';
2 import checkUserLogin from '@salesforce/apex/LoginController.checkUserLogin';
3
4 export default class LoginPage extends LightningElement {
5     @track email = '';
6     @track password = '';
7     @track showError = false;
8     @track isLoggedIn = false;
9     @track customerInfo = {};
10    @track parcelInfo = {};
11
12    handleEmailChange(event) {
13        this.email = event.target.value;
14    }
15
16    handlePasswordChange(event) {
17        this.password = event.target.value;
18    }
19
20    async handleLogin() {
21        this.showError = false;
22
23        try {
24            const result = await checkUserLogin({ email: this.email, password: this.password });
25
26            if (result.success) {
27                this.customerInfo = result.customer;
28                this.parcelInfo = result.parcel;
29                this.isLoggedIn = true;
30            } else {
31                this.showError = true;
32            }
33        } catch (error) {
34            console.error('Error during login:', error);
35            this.showError = true;
36        }
37    }
38}
```

Login (JavaScript) code:

```
import { LightningElement, track } from 'lwc';
import checkUserLogin from '@salesforce/apex/LoginController.checkUserLogin';

export default class LoginPage extends LightningElement {
    @track email = "";
    @track password = "";
    @track showError = false;
    @track isLoggedIn = false;
    @track customerInfo = {};
    @track parcelInfo = {};

    handleEmailChange(event) {
        this.email = event.target.value;
    }

    handlePasswordChange(event) {
```

```
        this.password = event.target.value;
    }

async handleLogin() {
    this.showError = false;

    try {
        const result = await checkUserLogin({ email: this.email, password: this.password
    });

    if (result.success) {
        this.customerInfo = result.customer;
        this.parcelInfo = result.parcel;
        this.isLoggedIn = true;
    } else {
        this.showError = true;
    }
} catch (error) {
    console.error('Error during login:', error);
    this.showError = true;
}
}
```

```

EXPLORER          ...
JS login.js      login.html      login.js-meta.xml
force-app > main > default > lwc > login > login.html > template > lightning-card > template > div.slds-p-around_medium > lightning-card
1   <template>
2     <lightning-card title="Login Page" icon-name="custom:custom14">
3       <template if:false={isLoggedin}>
4         <div class="slds-p-around_medium">
5           <lightning-input label="Email" type="email" value={email} onchange={handleEmailChange}></lightning-input>
6           <lightning-input label="Password" type="password" value={password} onchange={handlePasswordChange}></lightning-input>
7           <lightning-button label="Login" onclick={handleLogin} variant="brand"></lightning-button>
8         </div>
9       </template>
10      <template if:true={isLoggedin}>
11        <div class="slds-p-around_medium">
12          <lightning-card title="Customer Details" icon-name="standard:account">
13            <div class="slds-p-around_medium">
14              <p><strong>Email:</strong> {customerInfo.customer_Email_Id_c}</p>
15              <!-- Add other customer details here -->
16            </div>
17          </lightning-card>
18
19          <lightning-card title="Parcel Details" icon-name="standard:opportunity">
20            <div class="slds-p-around_medium">
21              <p><strong>Parcel ID:</strong> {parcelInfo.Parcel_Id_c}</p>
22              <p><strong>Status:</strong> {parcelInfo.ParcelStatus_c}</p>
23              <p><strong>Delivery Date:</strong> {parcelInfo.Delivery_Date_c}</p>
24              <p><strong>Delivery Personnel:</strong> {parcelInfo.Delivery_Personnel_Name_c}</p>
25              <p><strong>Contact Number:</strong> {parcelInfo.Contact_Number_c}</p>
26              <!-- Add other parcel details here -->
27            </div>
28
29          </lightning-card>
30        </div>
31      </template>
32    </lightning-card>
33  </template>
34 </template>
35

```

Login (html) code:

```

<template>
  <lightning-card title="Login Page" icon-name="custom:custom14">
    <template if:false={isLoggedin}>
      <div class="slds-p-around_medium">
        <lightning-input label="Email" type="email" value={email}>
          onchange={handleEmailChange}</lightning-input>
        <lightning-input label="Password" type="password" value={password}>
          onchange={handlePasswordChange}</lightning-input>
        <lightning-button label="Login" onclick={handleLogin}>
          variant="brand"</lightning-button>
        <div if:true={showError} class="slds-text-color_error">Invalid email or password</div>
      </div>
    </template>
  </template>

  <template if:true={isLoggedin}>
    <div class="slds-p-around_medium">
      <lightning-card title="Customer Details" icon-name="standard:account">

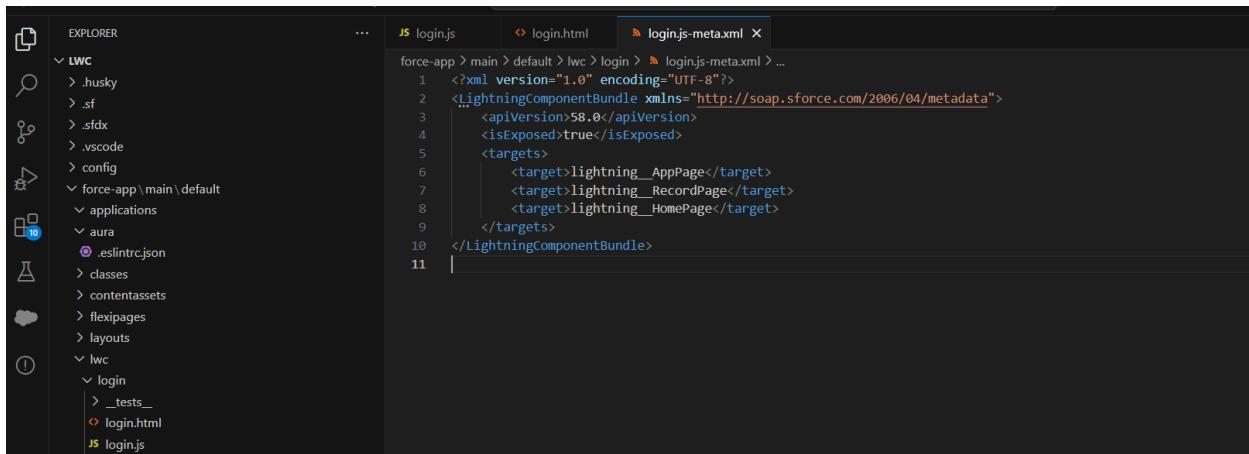
```

```

<div class="slds-p-around_medium">
    <p><strong>Email:</strong> {customerInfo.Customer_Email_Id__c}</p>
    <!-- Add other customer details here -->
</div>
</lightning-card>

<lightning-card title="Parcel Details" icon-name="standard:opportunity">
    <div class="slds-p-around_medium">
        <p><strong>Parcel ID:</strong> {parcellInfo.Parcel_Id__c}</p>
        <p><strong>Status:</strong> {parcellInfo.ParcelStatus__c}</p>
        <p><strong>Delivery Date:</strong> {parcellInfo.Delivery_Date__c}</p>
        <p><strong>Delivery Personnel:</strong>
{parcellInfo.Delivery_Personnel_Name__c}</p>
        <p><strong>Contact Number:</strong>
{parcellInfo.Contact_Number__c}</p>
        <!-- Add other parcel details here -->
    </div>
</lightning-card>
</div>
</template>
</lightning-card>
</template>

```



The screenshot shows the Salesforce Dev Console interface. On the left is the Explorer sidebar with project files like .husky, .sf, .sfdx, .vscode, config, force-app/main/default/applications/aura/.eslintrc.json, classes, contentassets, flexipages, layouts, lwc/login/_tests_, login.html, and login.js. The main area has tabs for JS login.js, login.html, and login.js-meta.xml. The login.js-meta.xml tab is active, displaying the XML code:

```

force-app > main > default > lwc > login > login.js-meta.xml ...
1  <?xml version="1.0" encoding="UTF-8"?>
2  <LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
3      <apiVersion>58.0</apiVersion>
4      <isExposed>true</isExposed>
5      <targets>
6          <target>lightning__AppPage</target>
7          <target>lightning__RecordPage</target>
8          <target>lightning__HomePage</target>
9      </targets>
10     </LightningComponentBundle>
11

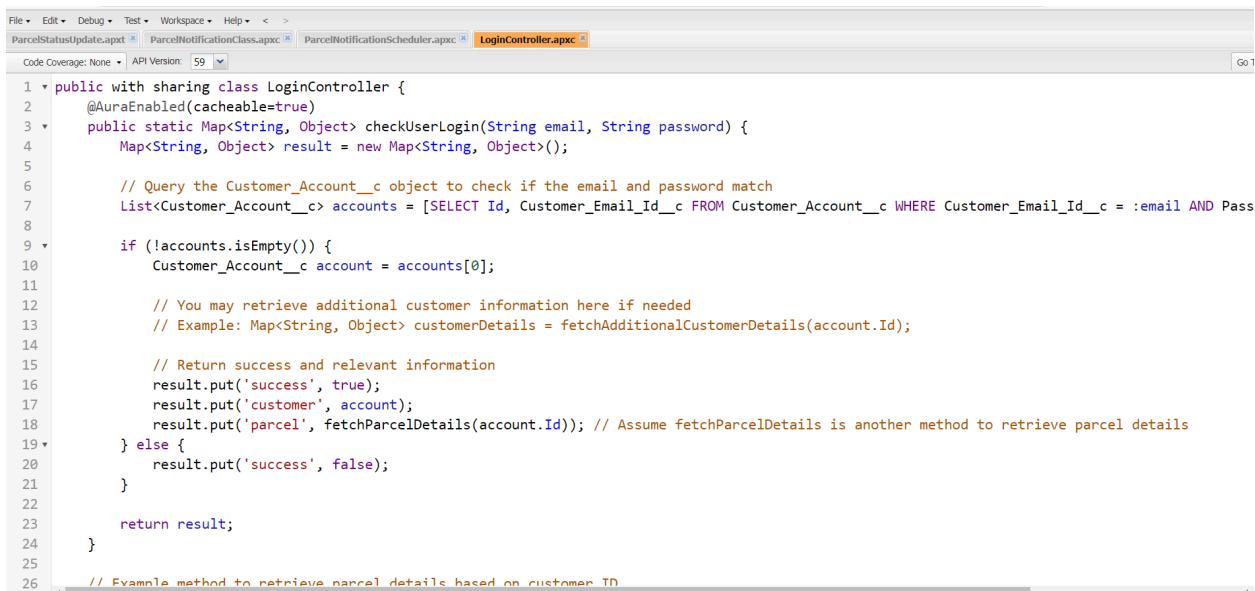
```

Login (xml) code:

```
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>58.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__AppPage</target>
        <target>lightning__RecordPage</target>
        <target>lightning__HomePage</target>
    </targets>
</LightningComponentBundle>
```

Activity- 2

Apex Controller (LoginController.cls):



The screenshot shows the Salesforce IDE interface with the LoginController.apxc file selected in the tabs. The code editor displays the following Apex class:

```
1 public with sharing class LoginController {
2     @AuraEnabled(cacheable=true)
3     public static Map<String, Object> checkUserLogin(String email, String password) {
4         Map<String, Object> result = new Map<String, Object>();
5
6         // Query the Customer_Account__c object to check if the email and password match
7         List<Customer_Account__c> accounts = [SELECT Id, Customer_Email_Id__c FROM Customer_Account__c WHERE Customer_Email_Id__c = :email AND Pass
8
9         if (!accounts.isEmpty()) {
10             Customer_Account__c account = accounts[0];
11
12             // You may retrieve additional customer information here if needed
13             // Example: Map<String, Object> customerDetails = fetchAdditionalCustomerDetails(account.Id);
14
15             // Return success and relevant information
16             result.put('success', true);
17             result.put('customer', account);
18             result.put('parcel', fetchParcelDetails(account.Id)); // Assume fetchParcelDetails is another method to retrieve parcel details
19         } else {
20             result.put('success', false);
21         }
22
23         return result;
24     }
25
26     // Example method to retrieve parcel details based on customer ID
}
```

```
public with sharing class LoginController {
    @AuraEnabled(cacheable=true)
    public static Map<String, Object> checkUserLogin(String email, String password) {
        Map<String, Object> result = new Map<String, Object>();
```

```

// Query the Customer_Account__c object to check if the email and password
match
List<Customer_Account__c> accounts = [SELECT Id, Customer_Email_Id__c FROM
Customer_Account__c WHERE Customer_Email_Id__c = :email AND Password__c =
:password LIMIT 1];

if (!accounts.isEmpty()) {
    Customer_Account__c account = accounts[0];

    // You may retrieve additional customer information here if needed
    // Example: Map<String, Object> customerDetails =
fetchAdditionalCustomerDetails(account.Id);

    // Return success and relevant information
    result.put('success', true);
    result.put('customer', account);
    result.put('parcel', fetchParcelDetails(account.Id)); // Assume fetchParcelDetails
is another method to retrieve parcel details
} else {
    result.put('success', false);
}

return result;
}

// Example method to retrieve parcel details based on customer ID
private static Map<String, Object> fetchParcelDetails(Id customerId) {
    Map<String, Object> parcelDetails = new Map<String, Object>();

    // Implement your logic to query parcel details based on the customer ID
    // Example: Parcel__c parcel = [SELECT Id, Parcel_Details__c FROM Parcel__c
WHERE Customer__c = :customerId LIMIT 1];

    // Add relevant parcel information to the parcelDetails map

```

```

// parcelDetails.put('Parcel_Id__c', parcel.Id);
// parcelDetails.put('Parcel_Details__c', parcel.Parcel_Details__c);

return parcelDetails;
}

}

```

Milestone 12 : Flows

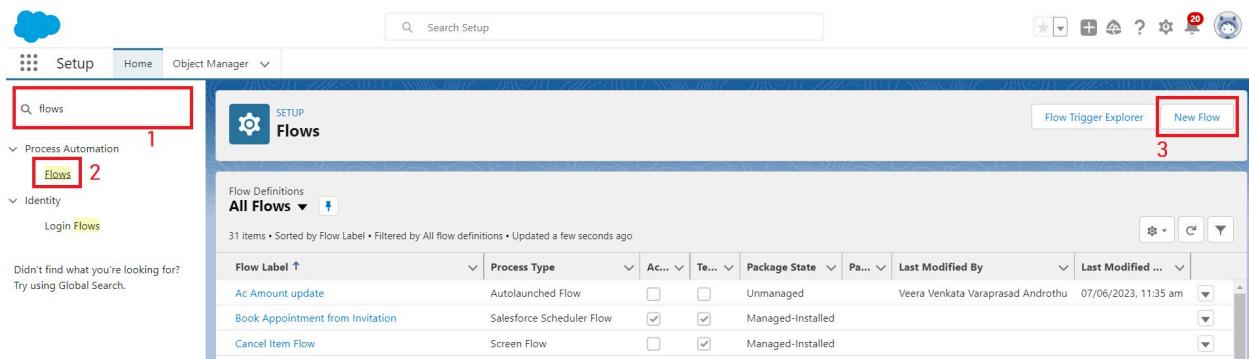
Why is Screen Flow required?

Screen Flows: These are used to guide users through a series of screens to collect or display information. Screen Flows are often used for data entry and updates. Using a screen flow for automating the creation of a customer account and record in a Parcel Dispatcher project enhances user experience, ensures data accuracy, and allows for a guided, efficient data entry process. The visual design and flexibility of screen flows make them a powerful tool for automating and optimizing business processes in Salesforce.

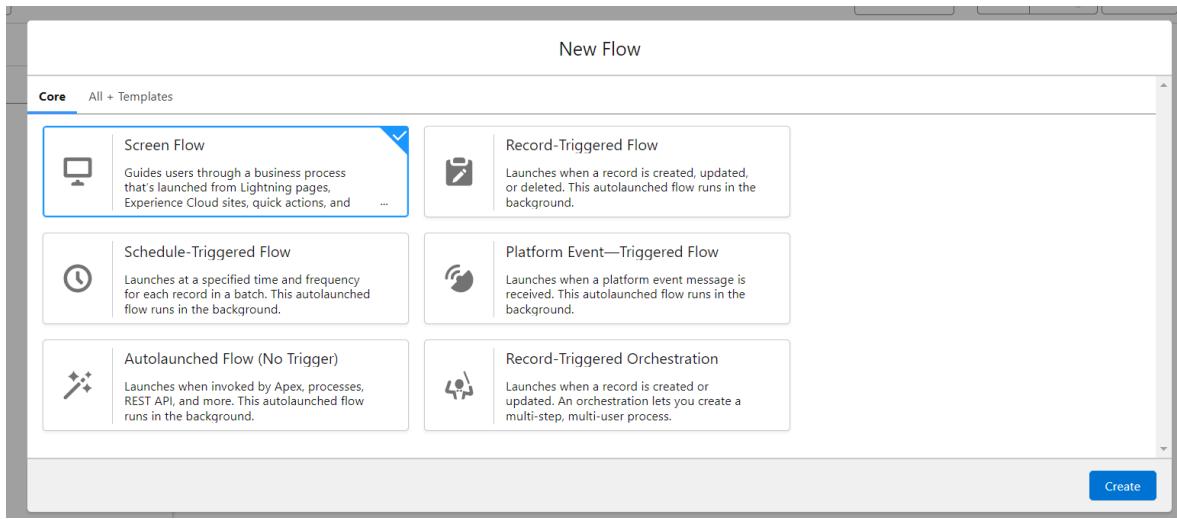
Activity 1:

Create a Screen Flow

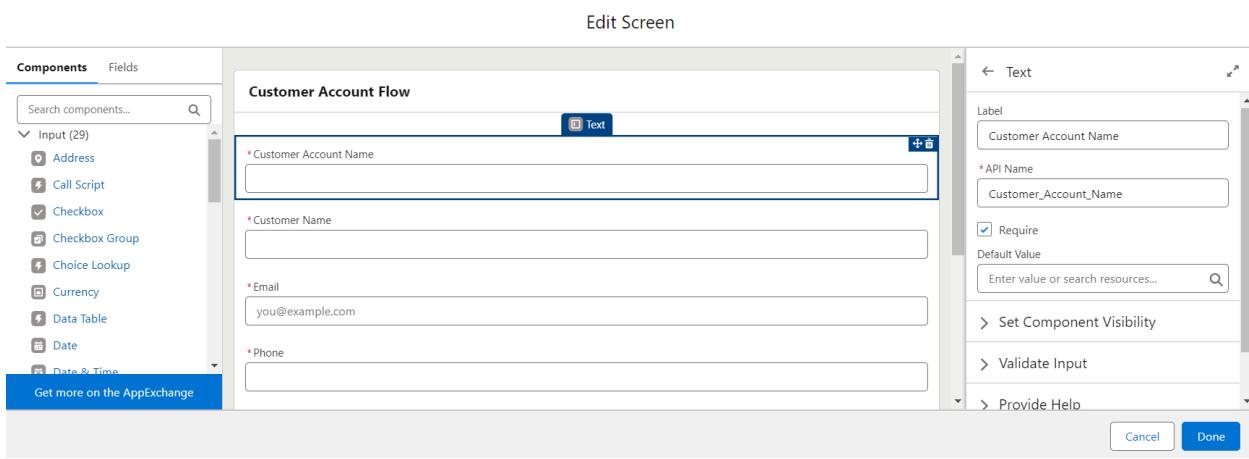
1. Go to setup → type Flow in quick find box → Click on the Flow and Select the New Flow.



2. Select the Screen flow and Click on Create.



3. Under the Screen Flow Click on “+” Symbol and In the Drop down List select the “Screen Element”.



4. Drag the “Text” component in to the flow element → Label “Customer Account Name”
→ Require checkbox should be selected → Done.
 5. Drag the “Text” component in to the flow element → Label “Customer Name”
→ Require checkbox should be selected → Done.

Edit Screen

Customer Account Flow

- * Customer Account Name
- * Customer Name
- * Email
- * Phone

Text

Label: Customer Name

API Name: Customer_Name

Require:

Default Value: Enter value or search resources...

Set Component Visibility

Validate Input

Provide Help

Cancel Done

6. Drag the “Email” component in to the flow element → API Name “EmailId”
 Label “EmailId” should be selected from the drop down → Disabled “!\$GlobalConstant.False”
 Read Only “!\$GlobalConstant.False” → Required “!\$GlobalConstant.True” → Done.

7. Drag the “Phone” component in to the flow element → API Name “Contact”
 Label “Phone” should be selected from the drop down → Required “!\$GlobalConstant.True”
 → Done.

Edit Screen

you@example.com

Phone

CustomerPassword

Latitude

Longitude

Pause Previous Finish

Phone

API Name: Contact

Label: Phone

Required: (!\$GlobalConstant.True)

Cancel Done

8. Drag the “Password” component in to the flow element → Label “Customer Password”
 → Required “!\$GlobalConstant.True” → Done.

Edit Screen

Components Fields

Search components...

Input (29)

- Address
- Call Script
- Checkbox
- Checkbox Group
- Choice Lookup
- Currency
- Data Table
- Date
- Date & Time

Get more on the AppExchange

you@example.com

*Phone

*CustomerPassword

Password

*Latitude ⓘ

*Longitude ⓘ

Pause Previous Finish

← Password

Label CustomerPassword

*API Name CustomerPassword

Require

Default Value Enter value or search resources...

> Set Component Visibility

> Validate Input

> Provide Help

Cancel Done

9. Drag the “Number” component in to the flow element → Label “Latitude”

→ Required “!\$GlobalConstant.True}” → Decimal Places “15.” →Done.

10. Drag the “Number” component in to the flow element → Label “Longitude”

→ Required “!\$GlobalConstant.True}” → Decimal Places “15.” →Done.

Edit Screen

Components Fields

Search components...

Input (29)

- Address
- Call Script
- Checkbox
- Checkbox Group
- Choice Lookup
- Currency
- Data Table
- Date
- Date & Time

Get more on the AppExchange

you@example.com

*Phone

*CustomerPassword

Number

*Latitude ⓘ

*Longitude ⓘ

Pause Previous Finish

← Number

Label Latitude

*API Name Latitude

Require

Default Value Enter value or search resources...

Decimal Places 15

> Set Component Visibility

Cancel Done

Edit Screen

Components Fields

Search components...

Input (29)

- Address
- Call Script
- Checkbox
- Checkbox Group
- Choice Lookup
- Currency
- Data Table
- Date
- Date & Time

Get more on the AppExchange

you@example.com

*Phone

*CustomerPassword

*Latitude ⓘ

Number

*Longitude ⓘ

Pause Previous Finish

← Number

Label Longitude

*API Name Longitude

Require

Default Value Enter value or search resources...

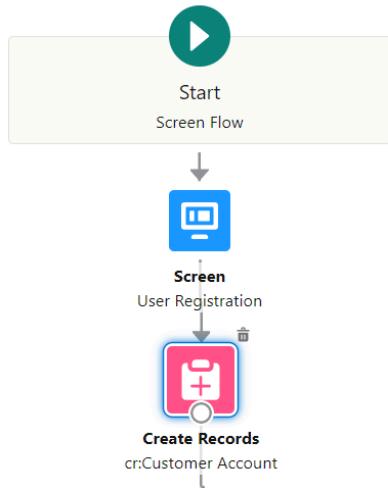
Decimal Places 15

> Set Component Visibility

Cancel Done

Activity 2:

- Under the Screen Element Click on “+” Symbol and In the Drop down List select the “Create Records Element”.



- Under the Create a record for this object “Customer Account” → Set Fields for Customer Account → Click Add Field And add the respective fields as shown → Done.

Edit Create Records

Create Salesforce records using values from the flow.

* Label cr:Customer Account	* API Name cr_Customer_Account
--------------------------------	-----------------------------------

Description

How Many Records to Create
 One
 Multiple

How to Set the Record Fields
 Use all values from a record
 Use separate resources, and literal values

Create a Record of This Object

* Object
Customer Account

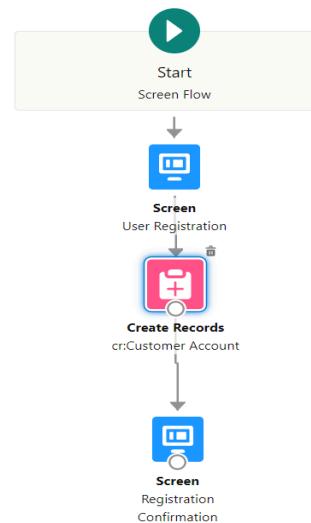
Edit Create Records

Set Field Values for the Customer Account

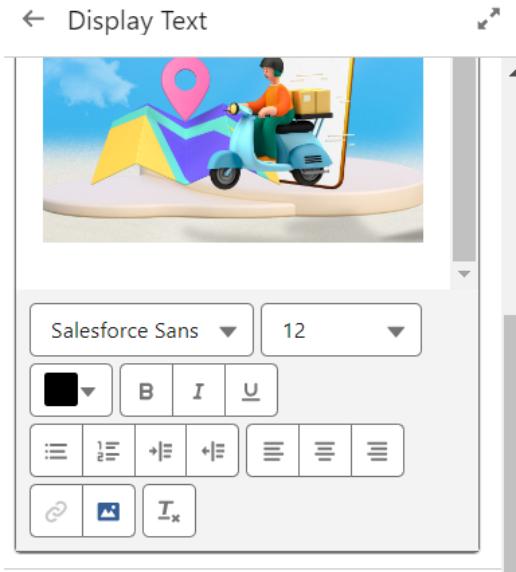
Field Customer_Address__Latitude__s	Value # Latitude X	<input type="button" value="Delete"/>
Field Customer_Address__Longitude__s	Value # Longitude X	<input type="button" value="Delete"/>
Field Customer_Contact__c	Value Aa Contact > Value X	<input type="button" value="Delete"/>
Field Customer_Email_Id__c	Value Aa EmailId > Value X	<input type="button" value="Delete"/>
Field Customer_Name__c	Value Aa Customer_Account_Name X	<input type="button" value="Delete"/>
Field Name	Value Aa Customer_Name X	<input type="button" value="Delete"/>
Field Password__c	Value Aa CustomerPassword X	<input type="button" value="Delete"/>

Activity 3:

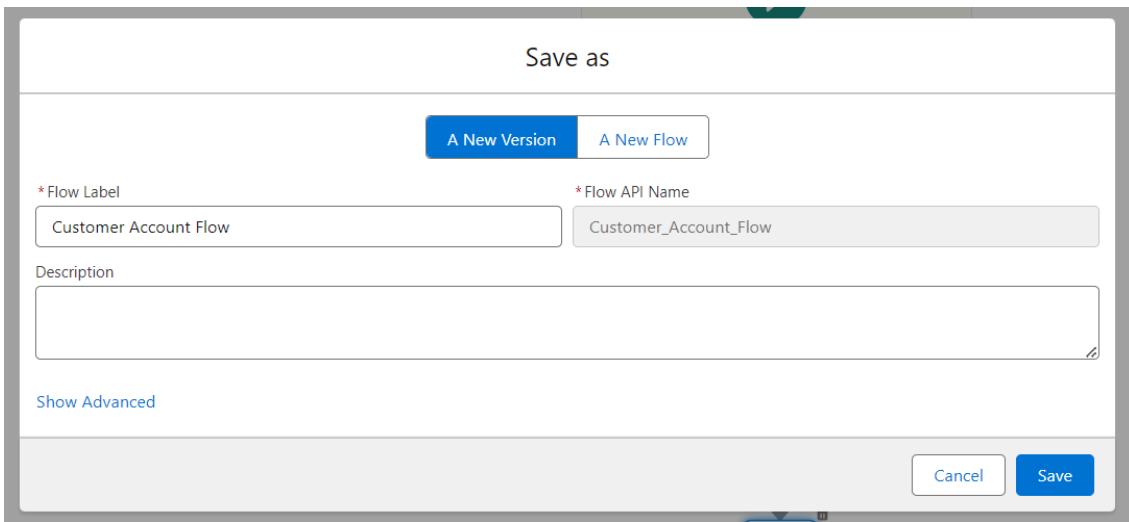
- Under the Create Records Element Click on “+” Symbol and In the Drop down List select the “Screen Element”.



2. Drag the “Display Text” component into the flow element → API Name “User_Registration_Confirmed” → Resource picker add a picture →Done.



3. Click on Save and save the flow as “Customer Registration”.



4. Click on the app launcher type “delivery Dispatch and Tracing” Click the gear icon then edit the page. Drag the Login page and flow component on the page.

The screenshot shows the Salesforce Setup interface. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. A search bar at the top right contains the text 'Search Setup'. On the left, a sidebar titled 'Apps' lists 'Digital Experiences', 'Playground Starter', and 'Delivery Dispatch and Tracing'. The main content area is titled 'Flows' and displays a list of flow definitions, with 'Delivery Dispatch and Tracing' highlighted. A message at the bottom indicates the list is sorted by Flow Label, filtered by All flow definitions, and updated 2 hours ago.

The screenshot shows the Salesforce Page Builder interface. The top navigation bar includes 'Desktop', 'Shrink To View', and a save button. On the left, a sidebar titled 'Components' lists various components like Accordion, App Launcher, Assistant, and Dashboard. The main canvas displays a 'Login Page' component with fields for Email and Password, and a 'Today's Events' and 'Today's Tasks' section. To the right, a configuration panel for a 'Page' component is open, showing fields for 'Label' (Home Page Default), 'API Name' (Home_Page_Default1), 'Page Type' (Home Page), 'Template' (Standard Home Page), and 'Description'.

5. Your page should look like this.

The screenshot shows the final Salesforce home page layout. The top navigation bar includes 'Delivery Dispatch a...', 'Home', 'Parcel's', 'Customer Accounts', 'Delivery Personnel', 'Reports', 'Dashboards', and 'Calendar'. The main content area features a 'Login Page' component and several other sections: 'Today's Events' (with a weather forecast icon), 'Today's Tasks' (empty), 'Key Deals - Recent Opportunities' (empty), and an 'Assistant' section with a placeholder message. To the right, a 'Customer Account Flow' form is displayed, requiring fields for Customer Account Name, Customer Name, Email (you@example.com), Phone, Customer Password, and Latitude.

Project Ends

Project Related Questions:

1. Which Salesforce object is used to store information about the parcels, including Parcel ID, Sender Name, Receiver Name, Weight, Size, and Delivery Address?

- a. Account
- b. Contact
- c. Custom Object - Parcel
- d. Opportunity

Solution: c

2. In the Parcel object, a formula field named "Delivery Time" is needed. What type of formula should be used to calculate the time taken for delivery based on the Pickup Date and Estimated Delivery Date?

- a. Text
- b. Number
- c. Date/Time
- d. Formula fields can't perform calculations on dates

Solution: c

3. Which of the following is true about Apex in the Salesforce platform?

- a. Apex is a declarative language.
- b. Apex is used for designing page layouts.
- c. Apex is executed on the client-side.
- d. Apex is a server-side programming language for building business processes.

Solution: d

4. What is the main purpose of an Apex trigger in the context of the Parcel Dispatcher project?

- a. To define custom fields on objects.
- b. To automate actions based on events like changing the status of a parcel.
- c. To create user interfaces for the project.
- d. To generate reports and analytics.

Solution: b

5.What is the primary advantage of using Lightning Web Components (LWC) in the Parcel Dispatcher project?

- a. LWC provides declarative features for building forms.
- b. LWC facilitates server-side processing.
- c. LWC allows for building responsive and efficient user interfaces.
- d. LWC is specifically designed for writing backend logic.

Solution: c

6. What is the purpose of using Schedule Apex in the Parcel Dispatcher project?

- a. To create schedules for parcel pickups.
- b. To automate certain tasks at specific times, such as sending notifications to customers.
- c. To define the delivery routes for delivery personnel.
- d. To generate reports on parcel dispatch efficiency.

Solution: b

7. Which type of Salesforce report would be most suitable for tracking the status of parcels based on different record types (Pickup, In Transit, Delivered)?

- a. Tabular Report
- b. Summary Report
- c. Matrix Report
- d. Joined Report

Solution: c

8.What role do Flows play in the Parcel Dispatcher project?

- a. Flows are used to design user interfaces for the application.
- b. Flows automate business processes and guide users through the steps.
- c. Flows manage the database relationships between objects.
- d. Flows are used for defining security settings.

Solution: b

9. In the project, which Salesforce object is used to store information about delivery personnel, including Name, Email, Phone, and Route?

- a. Account
- b. Contact
- c. Custom Object - Delivery Personnel
- d. Opportunity

Solution: c

10. Which of the following is true about formula fields in Salesforce?

- a. Formula fields can only reference fields on the same object.
- b. Formula fields can reference fields on related objects.
- c. Formula fields can execute Apex code.
- d. Formula fields are used for defining record types.

Solution: b

11. When does an Apex trigger execute in Salesforce?

- a. During the data loading process.
- b. After a record is inserted, updated, or deleted.
- c. When a user logs in.
- d. Apex triggers only run when explicitly invoked by the user.

Solution: b

12. What is the key feature of Lightning Web Components (LWC) that enhances performance?

- a. LWC uses server-side rendering.
- b. LWC components are automatically cached.
- c. LWC supports asynchronous loading of components.
- d. LWC components are built with Visualforce.

Solution: c

13. Which of the following statements about Schedule Apex is correct?

- a. Schedule Apex is used for real-time event handling.
- b. Schedule Apex can only be run once.

- c. Schedule Apex is suitable for background jobs and periodic tasks.
 - d. Schedule Apex is designed for user interface development. Solution: c

14.What is the primary purpose of dashboards in Salesforce?

- a. To define security settings.
 - b. To create workflows.
 - c. To visualize and analyze data from reports.
 - d. To design user interfaces.

Solution: c

15. In the Parcel Dispatcher project, how can Flows be utilized for automation?

- a. Flows can be used to design website layouts.
 - b. Flows can automate repetitive tasks and guide users through complex processes.
 - c. Flows are used to define relationships between objects.
 - d. Flows can generate scheduled reports.

Solution: b