

PROJECT REPORT

Naan Mudhalvan – Salesforce Developer

Project Title:

TripAdvisor E-Management

Team Members:

Name	NM ID	32 digit NM ID
JAYASEELAN B	AU710621104015	D21266394D9B17682C94C46A1FFCAD6E
SRIRAM V	AU710621104052	32DC584E0ADDCB198FB430D7A392D7B0
SANJAY B	AU710621104042	FD8DFDC131AB1EE9C14438EA95B86E39
DHANUSH T	AU710621104010	788C65A7EA23C7FAE9F4B875E4C1AED7

**CSI COLLEGE OF ENGINEERING
KETTI**

Report: TripAdvisor E-Management

Project Overview

Project Title: *TripAdvisor E-Management*

This project is focused on developing the *TripAdvisor E-Management System* using Salesforce to streamline travel-related services, such as hotel, food option, and flight management, along with customer discount automation and notification services. The project aims to address inefficiencies in data handling and communication within TripAdvisor's ecosystem by implementing a comprehensive and automated solution. The primary challenge was to ensure seamless integration of various services to enhance operational efficiency, reduce manual errors, and improve the user experience.

By leveraging Salesforce's robust platform features, such as automation tools, Apex triggers, and schedulable classes, this project provides a scalable, reliable, and efficient solution to meet the business needs of TripAdvisor.

Project Description:

The TripAdvisor E-Management system, integrated with Salesforce, aims to provide an all-in-one travel companion app that empowers users to plan, book, and make the most of their trips. This system brings TripAdvisor's massive repository of user-generated reviews and insights into the Salesforce ecosystem, allowing users to access information on hotels, flights, food options, and customer deals seamlessly. This report outlines the system requirements, acceptance criteria, and solutions developed for this integration, with a focus on automation and streamlined customer experience.

Short Description:

TripAdvisor E-Management Solution streamlines travel itinerary management, centralizing booking data, trip tracking, and user feedback, making it easier for organizations to oversee and enhance travel experiences.

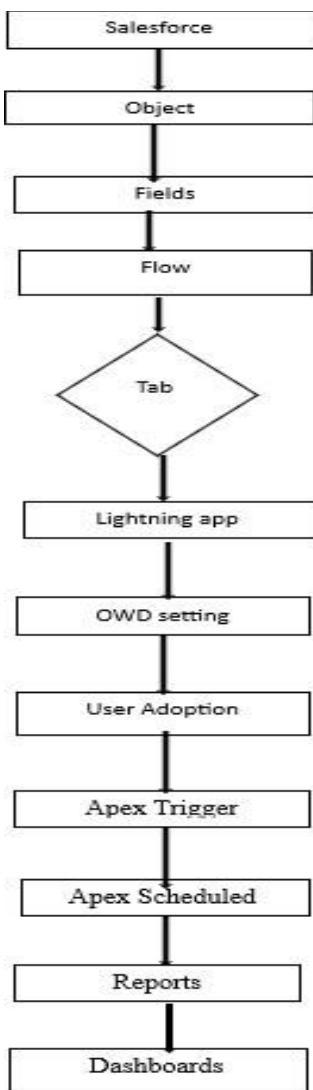
Objectives

Business Goals:

1. Automate the management of hotels, flights, and food options to reduce manual administrative tasks.
2. Provide a personalized customer experience through automated discount mechanisms.
3. Ensure timely communication with customers via email notifications for flight reminders.

Specific Outcomes:

1. Automated tracking and updating of hotel information based on food options.
2. Automated discount calculation and application based on customer purchase thresholds.
3. Flight reminder emails sent automatically 24 hours before flight departure, improving customer satisfaction.
4. Accurate, real-time reporting and data analytics for better business insights.



Detailed Steps to Solution Design

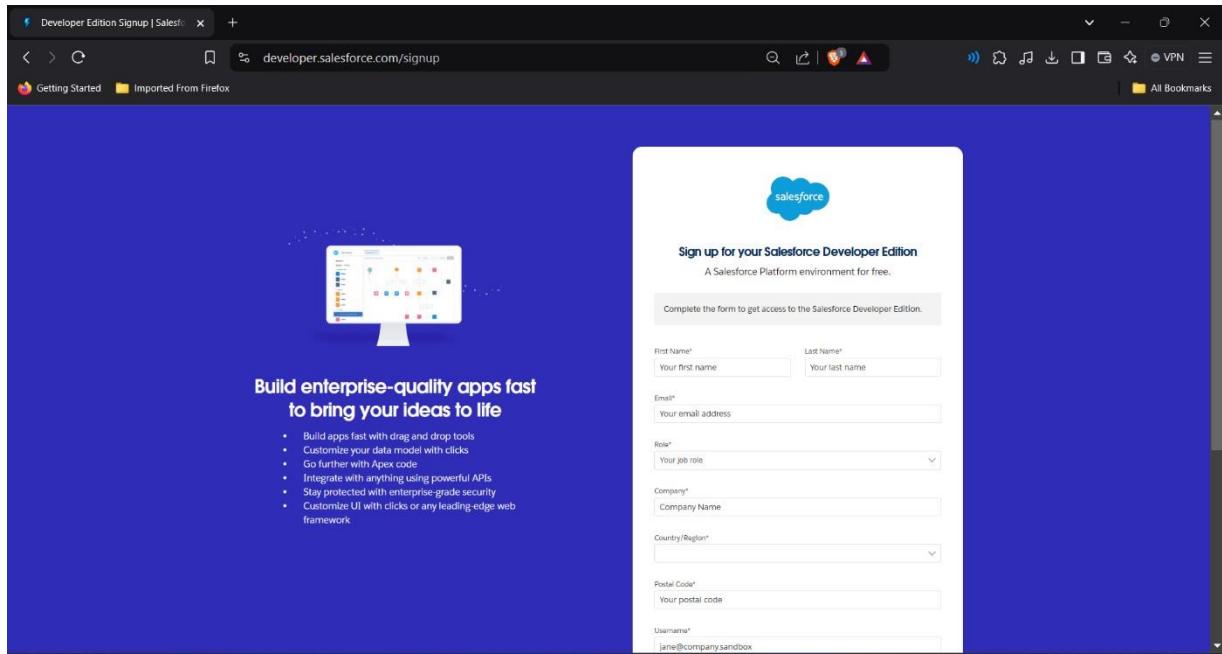
Developer Account Creation

To start working with Salesforce CRM, a developer account is essential. Follow these steps to create an account:

Sign-Up

- Go to [Salesforce Developer Sign-Up](#).
- Enter your **First and Last Name**, **Email**, and set **Role** as “Developer.”
- Input your **Company** (College Name), **Country** (India), **Postal Code**, and **Username** (formatted as username@organization.com).

Click **Sign Me Up** after filling out the form



Account Activation

- Open the inbox of the email used for registration, locate the Salesforce verification email, and click **Verify Account**.
- Set a password, choose a security question, and log into your Salesforce account to access the setup page.

Objects

Salesforce objects function as database tables for storing and organizing data relevant to the organization.

- **Standard Objects:** Provided by Salesforce by default (e.g., Accounts, Contacts).

- **Custom Objects:** User-defined objects to store unique organizational data.

System Development: Custom Objects

Hotel Object:

Hotel Object is created to ensure that when a new Food Option is added or updated with the necessary information

1. Enter label : Hotel
2. Plural Name : Hotels
3. Data Type : (text)
4. Field Name : Hotel Name
5. Click Allow Reports
6. Allow Search ? Save

Purpose: Store data about hotels and update hotel information when new food options are added.

Custom Object Information

The singular and plural labels are used in tabs, page layouts, and reports.
Be careful when changing the name or label as it may affect existing integrations and merge templates.

Label	Hotel	Example: Account
Plural Label	Hotels	Example: Accounts
Starts with vowel sound	<input type="checkbox"/>	

The Object Name is used when referencing the object via the API.

Object Name	Hotel	Example: Account
-------------	-------	------------------

Description: [Empty Text Area]

Context-Sensitive Help Setting: Open the standard Salesforce.com Help & Training window
 Open a window using a Visualforce page

Content Name: [None]

Enter Record Name Label and Format

The Record Name appears in page layouts, key lists, related lists, lookups, and search results. For example, the Record Name for Account is "Account Name" and for Case it is "Case Number". Note that the Record Name field is always called "Name" when referenced via the API.

Record Name	Hotel Name	Example: Account Name
-------------	------------	-----------------------

Data Type: Text Warning: If you plan to insert a high volume of records in this object, via the API for example, use the Text data type.

Custom Object Information

The singular and plural labels are used in tabs, page layouts, and reports.
Be careful when changing the name or label as it may affect existing integrations and merge templates.

Label	Food Option	Example: Account
Plural Label	Food Options	Example: Accounts
Starts with vowel sound	<input type="checkbox"/>	

The Object Name is used when referencing the object via the API.

Object Name	Food_Option	Example: Account
-------------	-------------	------------------

Description: [Empty Text Area]

Context-Sensitive Help Setting: Open the standard Salesforce.com Help & Training window
 Open a window using a Visualforce page

Content Name: [None]

Enter Record Name Label and Format

The Record Name appears in page layouts, key lists, related lists, lookups, and search results. For example, the Record Name for Account is "Account Name" and for Case it is "Case Number". Note that the Record Name field is always called "Name" when referenced via the API.

Record Name	Food Option Name	Example: Account Name
-------------	------------------	-----------------------

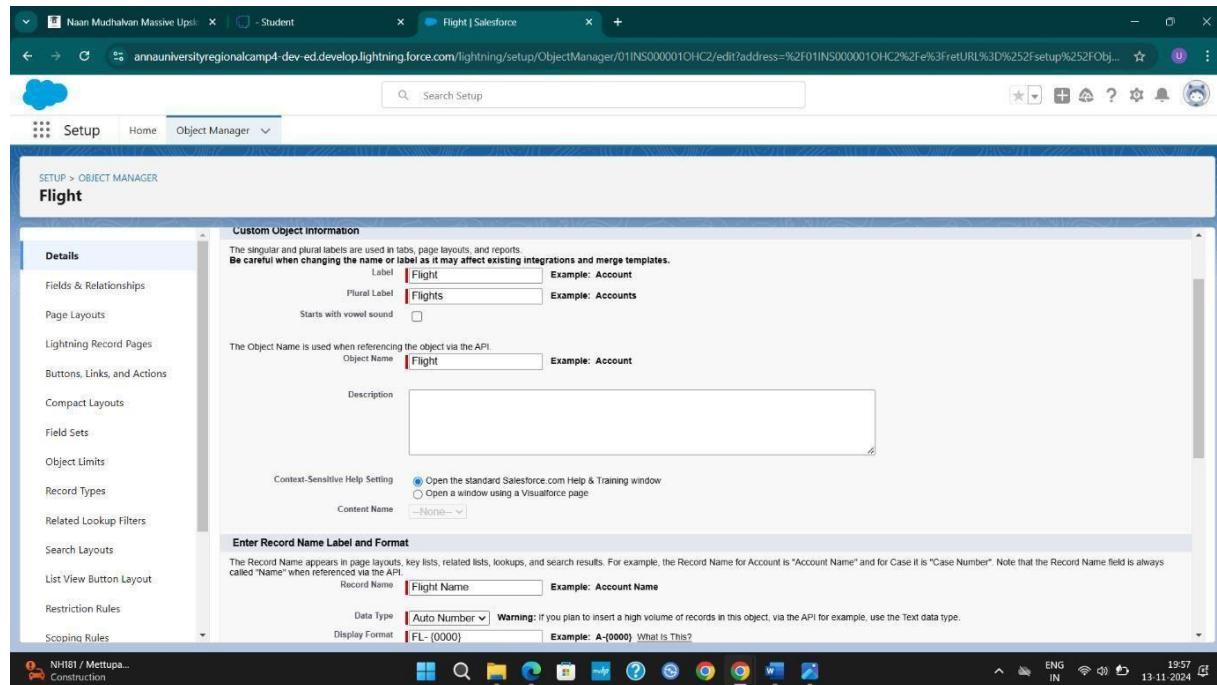
Data Type: Auto Number Warning: If you plan to insert a high volume of records in this object, via the API for example, use the Text data type.

Display Format: FO - (0000) Example: A-(0000) What Is This?

Flight Object:

Flight > Data Type > Auto Number > Format > FL- {0000}

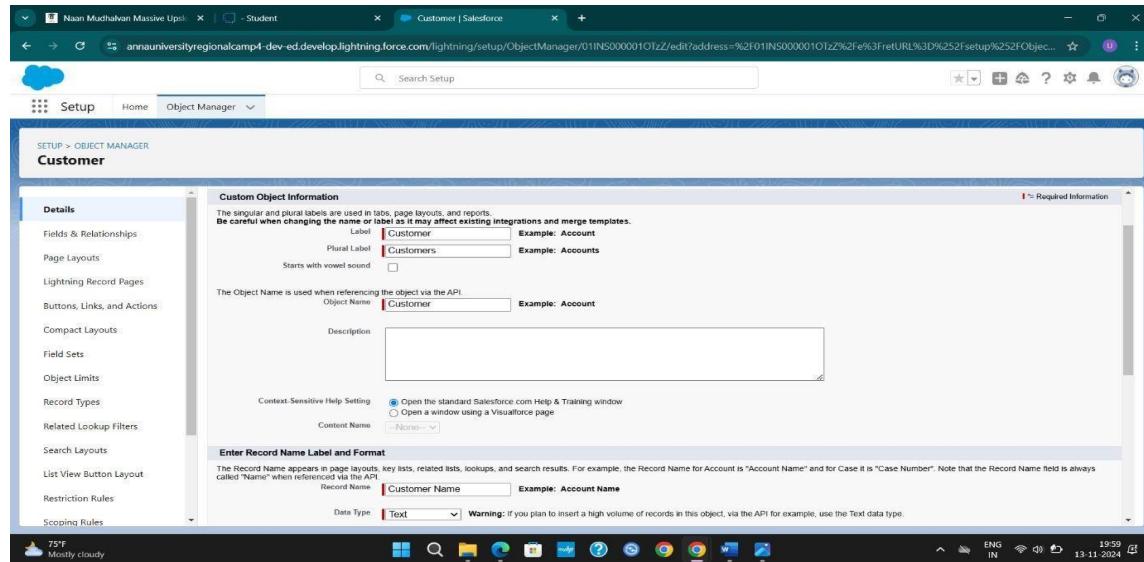
Purpose: Track flight bookings and manage customer notifications.



Customer Object:

Customer > Text > Field Name > Customer Name

Purpose: Manage customer information.

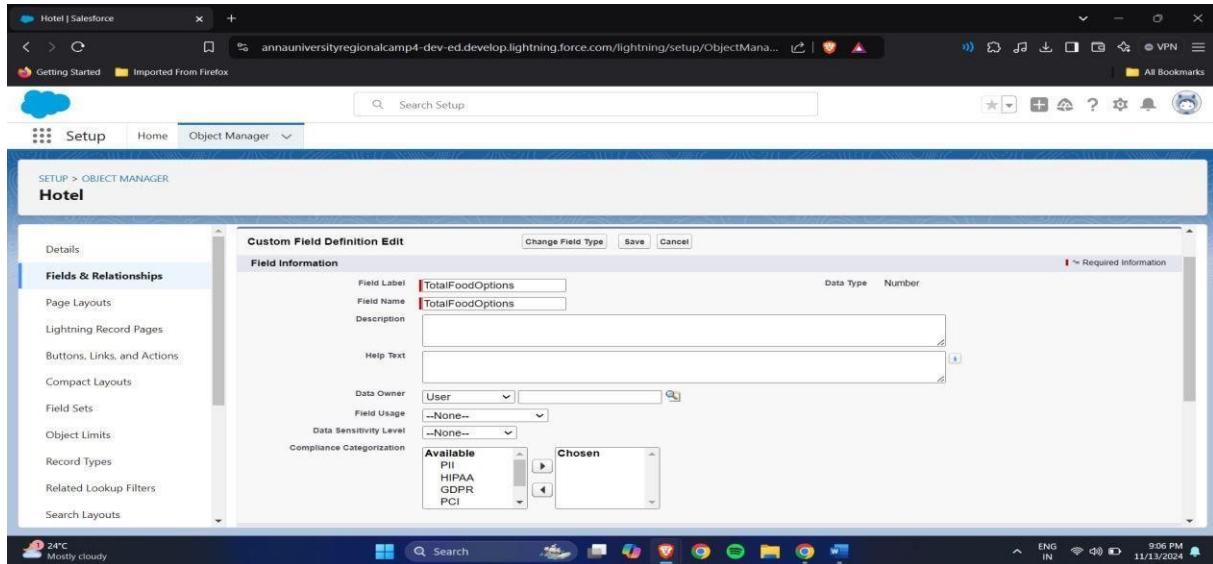


Fields

Create Fields for Hotel Object:

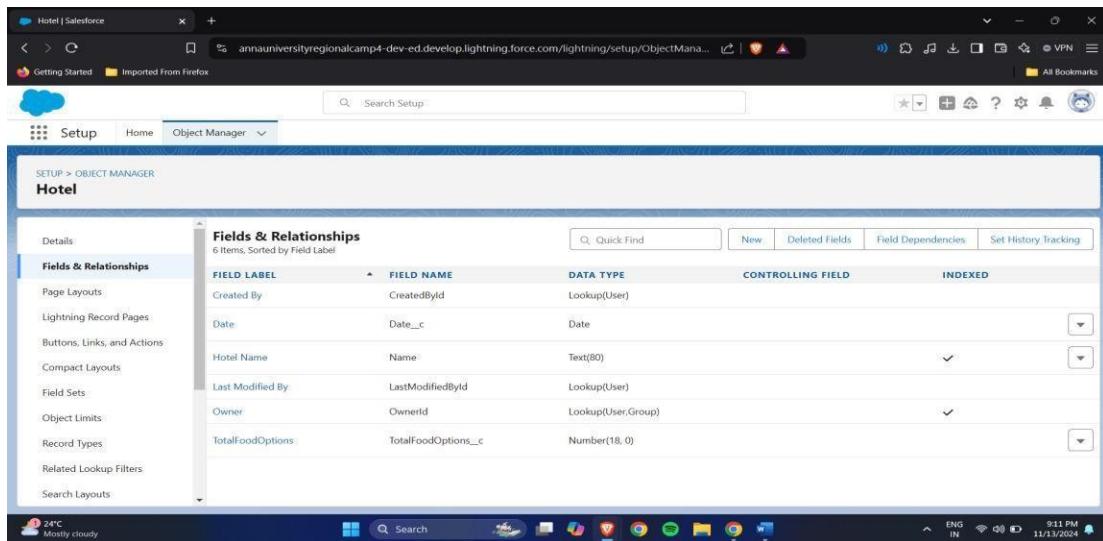
Creating fields for a "Hotel" object involves defining the data attributes that represent essential information about a hotel. These fields should capture the details needed to describe and manage the hotel within an application, database, or any system that tracks hotel information.

Sr. No.	Field Name	Data Type
1	TotalFoodOptions	Number
2	Date	Date



Fields & Relationships of all Hotel Fields:

In a system where you manage hotel data, creating fields and defining relationships for the "Hotel" object is crucial for organizing and retrieving information efficiently. Here's a detailed overview of the fields and relationships typically associated with a "Hotel" object.



Create Fields For Food Option:

Creating fields for a "Food Option" object is essential when building a system to manage food items, such as a restaurant menu or a hotel's food service options. These fields should cover all the necessary details that define each food item and make it easy for users to search, categorize, and manage food options.

Sr. No.	Field Name	Data Type
1	Name	Text
2	Hotel	Hotel(Lookup)
3	Food Amount	Currency

The screenshot shows the Salesforce Object Manager interface. The top navigation bar includes 'Food Option | Salesforce', a search bar, and various system icons. Below the header, the page title is 'SETUP > OBJECT MANAGER Food Option'. On the left, there's a sidebar with 'Fields & Relationships' selected, along with other options like 'Page Layouts', 'Lightning Record Pages', etc. The main content area displays a table titled 'Fields & Relationships' with 7 items. The columns are 'FIELD LABEL', 'FIELD NAME', 'DATA TYPE', 'CONTROLLING FIELD', and 'INDEXED'. The data includes:

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Food Amount	Food_Amount__c	Currency(18, 0)		
Food Option Name	Name	Auto Number		✓
Hotel	Hotel__c	Lookup(Hotel)		✓
Last Modified By	LastModifiedById	Lookup(User)		
Name	Name__c	Text(255)		
Owner	OwnerId	Lookup(User,Group)		✓

Create Fields in the Flight Object:

Creating fields in a "Flight" object involves defining essential attributes that represent information about a flight. These fields help to manage and organize flight details within a system for booking, tracking, or scheduling flights. Here's an example of typical fields for a "Flight" object

Sr. No.	Field Name	Data Type
1	Name	Date/Time
2	DepartureDateTime	Hotel(Lookup)

The screenshot shows the Salesforce Object Manager interface. The top navigation bar includes 'Flight | Salesforce', a search bar, and various system icons. Below the header, the page title is 'SETUP > OBJECT MANAGER Flight'. On the left, there's a sidebar with 'Fields & Relationships' selected, along with other options like 'Page Layouts', 'Lightning Record Pages', etc. The main content area displays a table titled 'Fields & Relationships' with 7 items. The columns are 'FIELD LABEL', 'FIELD NAME', 'DATA TYPE', 'CONTROLLING FIELD', and 'INDEXED'. The data includes:

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
ContactEmail	ContactEmail__c	Email		
Created By	CreatedById	Lookup(User)		
DepartureDateTime	DepartureDateTime__c	Date/Time		
Flight Name	Name	Auto Number		✓
Last Modified By	LastModifiedById	Lookup(User)		
Name	Name__c	Lookup(Hotel)		✓
Owner	OwnerId	Lookup(User,Group)		✓

Create Fields in the Customer Object:

Creating fields for a "Customer" object involves defining essential details to identify and understand each customer in the system. Here are common fields typically included

Sr. No.	Field Name	Data Type
1	Customer Name	Name
2	Discount Amount	Formula (Currency)
3	Discount Percent	Percentage

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Customer Name	Customer_Name__c	Text(255)		
Customer Name	Name	Text(80)		✓
Discount Amount	Discount_Amount__c	Currency(18, 0)		
Discount Percent	Discount_Percent__c	Percent(18, 0)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓

Flow for Customer Discount Automation:

A Salesforce Flow was created to apply discounts based on the customer's purchase amount. Discounts are granted if the amount exceeds certain thresholds:

Create a new flow variable “TripAdviser”.

Flow Variables

Create 3 variable :

Variable > Api name > foId > text > Available for Input

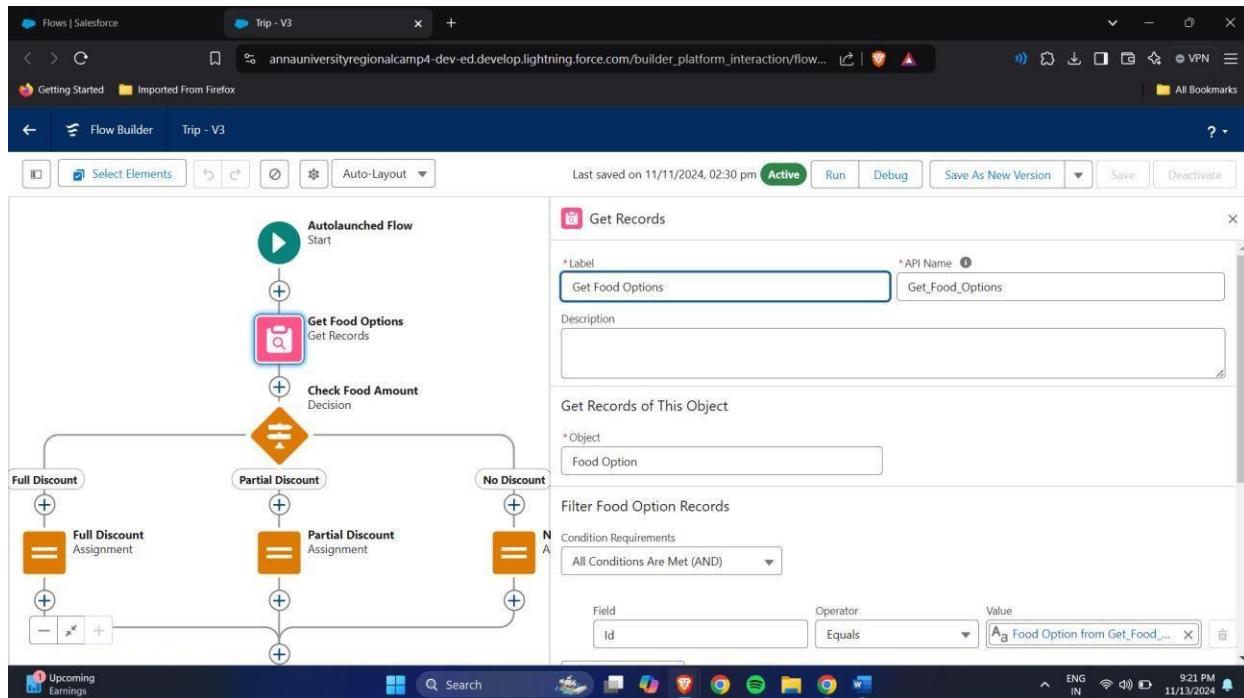
Variable > Api name > csId > text > Available for Input

Variable > Api name > discount > Number

Flow Logic:

Flow Variables are temporary placeholders used within a process or workflow to store and manipulate data as it moves through different stages of execution. These variables enable dynamic data handling, allowing information to be passed from one step to another within a flow.

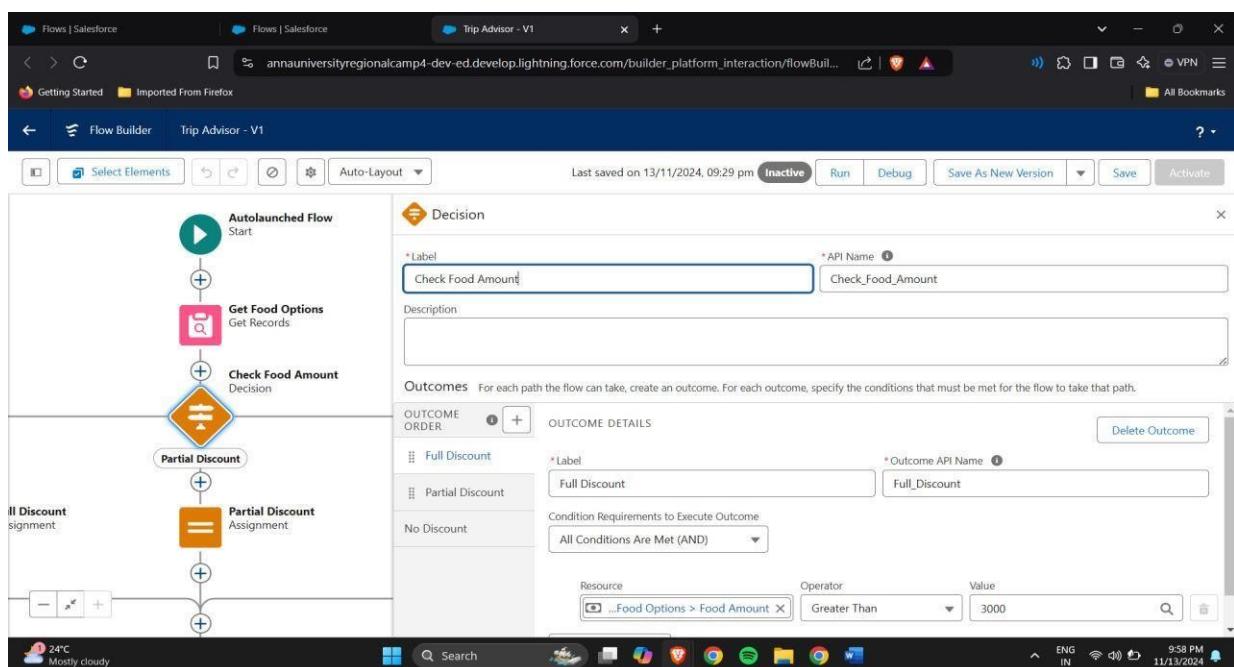
Get Records: Retrieve the necessary customer records.



Purpose:

The "**Get Records**" element in a flow (such as in Salesforce Flow or similar automation platforms) is to retrieve specific records from a database based on defined criteria. This action allows you to fetch data that can be used later in the flow for various purposes, such as updating records, making decisions, or displaying information.

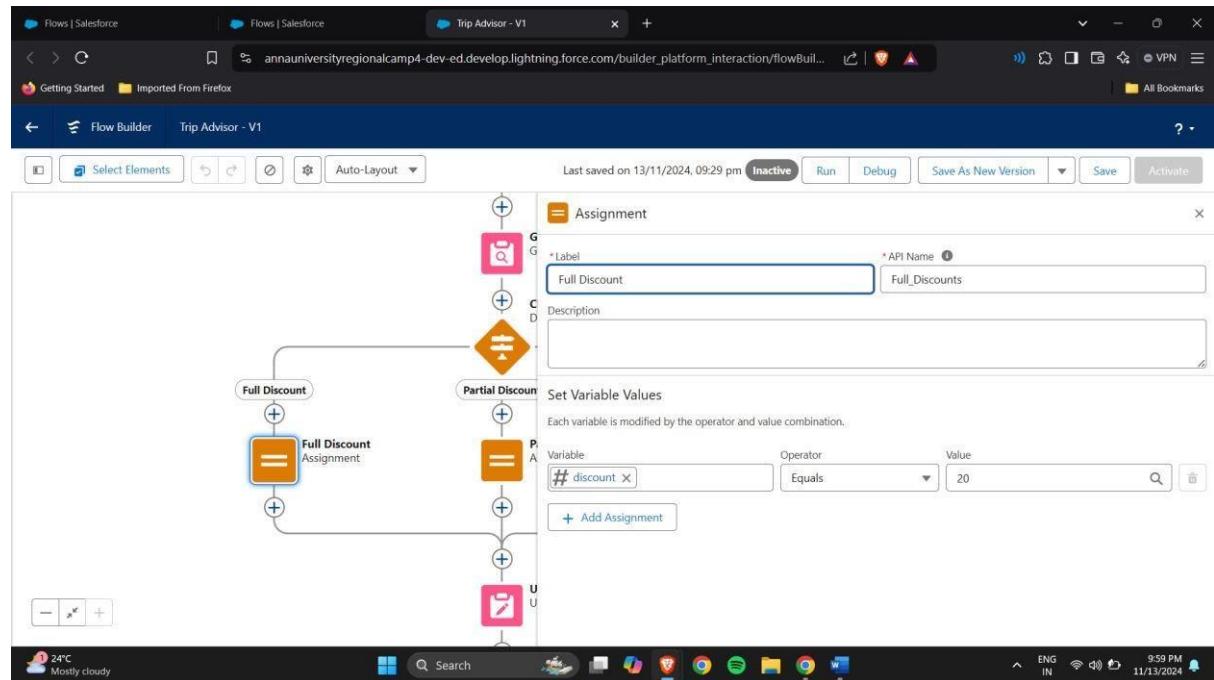
Decision Element: Determine the discount rate based on the purchase amount:



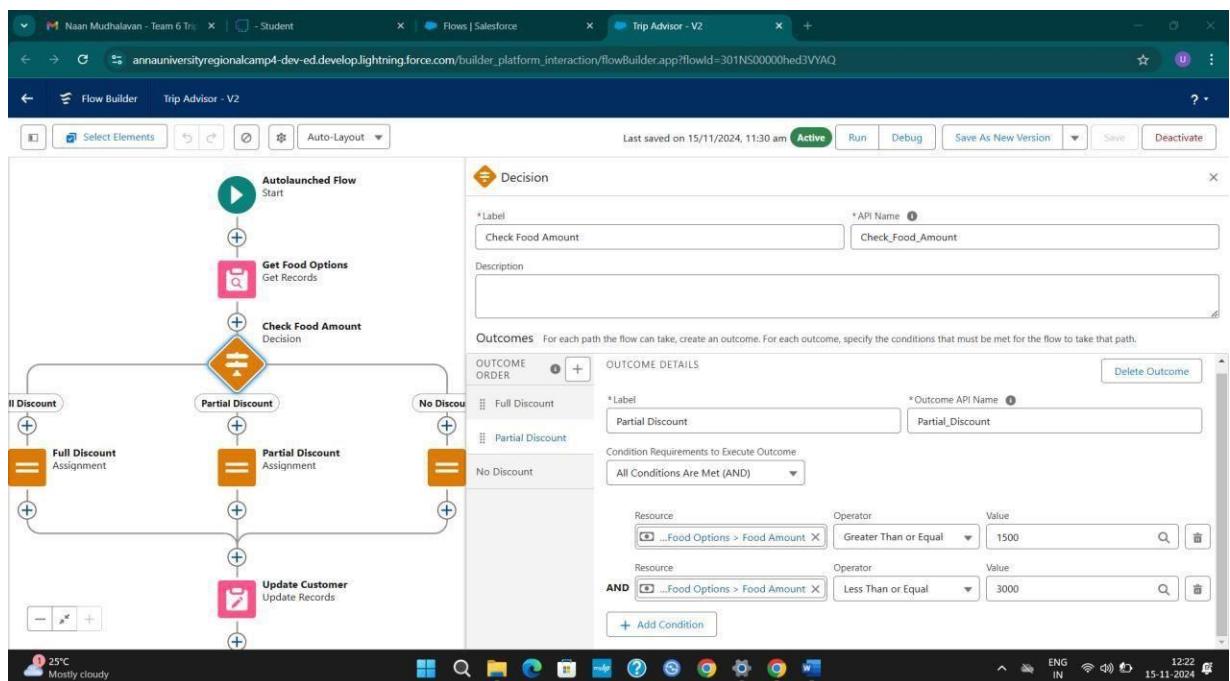
Purpose:

The Decision Element in a flow is used to control the flow's path based on specified conditions. This is particularly useful for creating dynamic, condition-based workflows

Full Discount: Amount exceeds 3000.

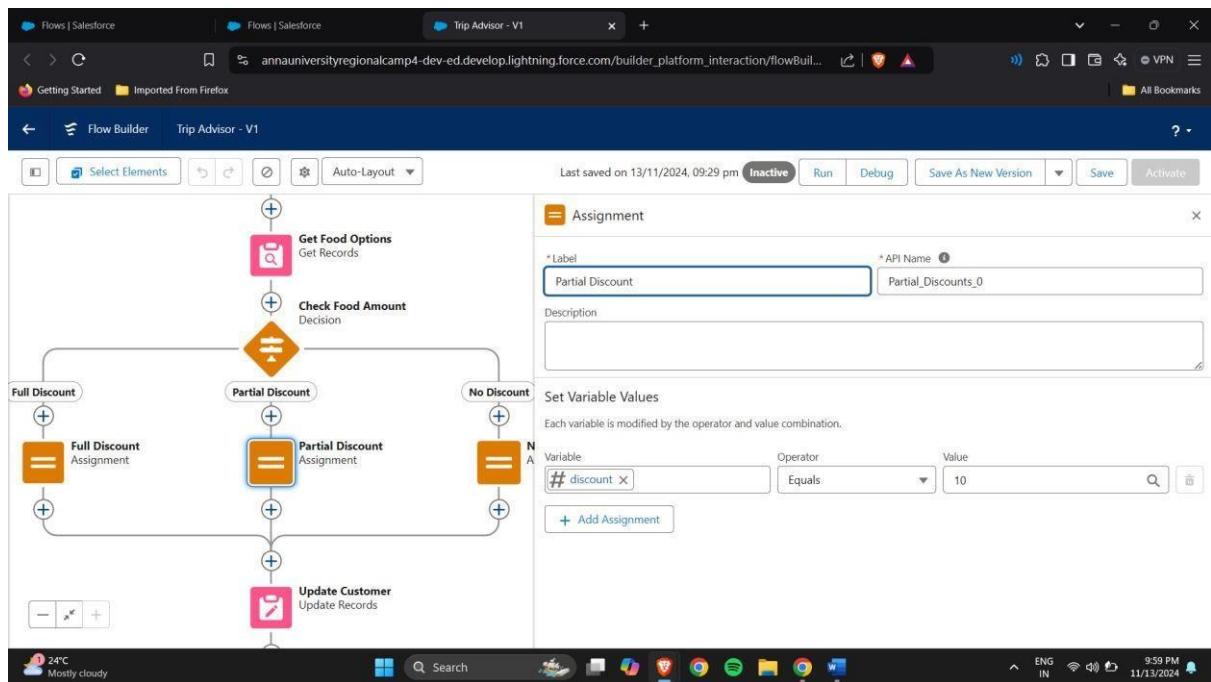


Partial Discount: Amount In-between 1500 to 3000.

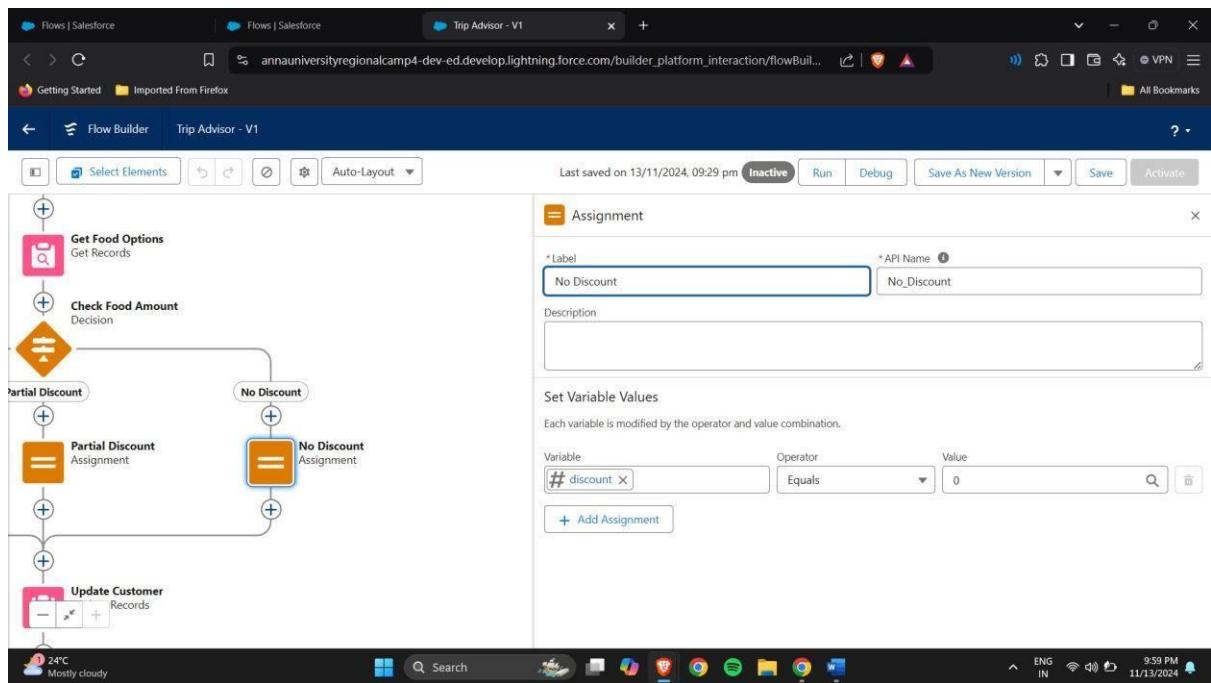


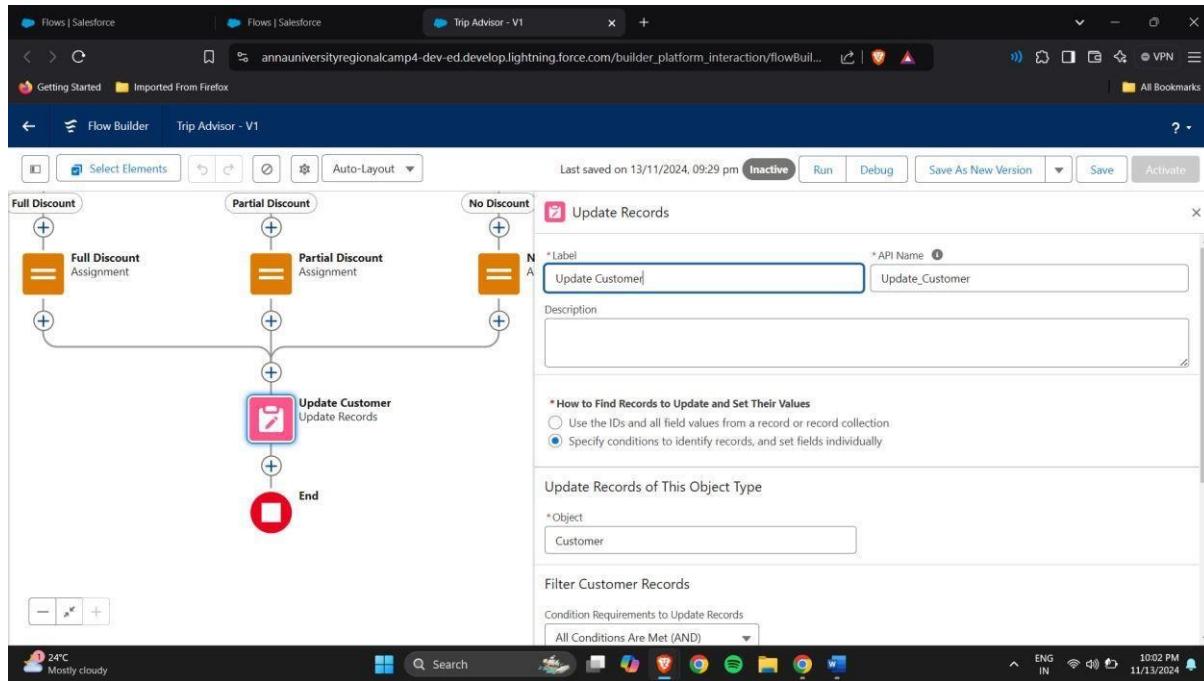
The Full Discount in a flow (such as a sales or customer service process) is to Provide Complete Financial Relief, Streamline Issue Resolution, Enhance Customer Loyalty and Support Promotional Strategies.

Partial Discount: Amount between 1500 and 3000.



No Discount: Amount below 1500.

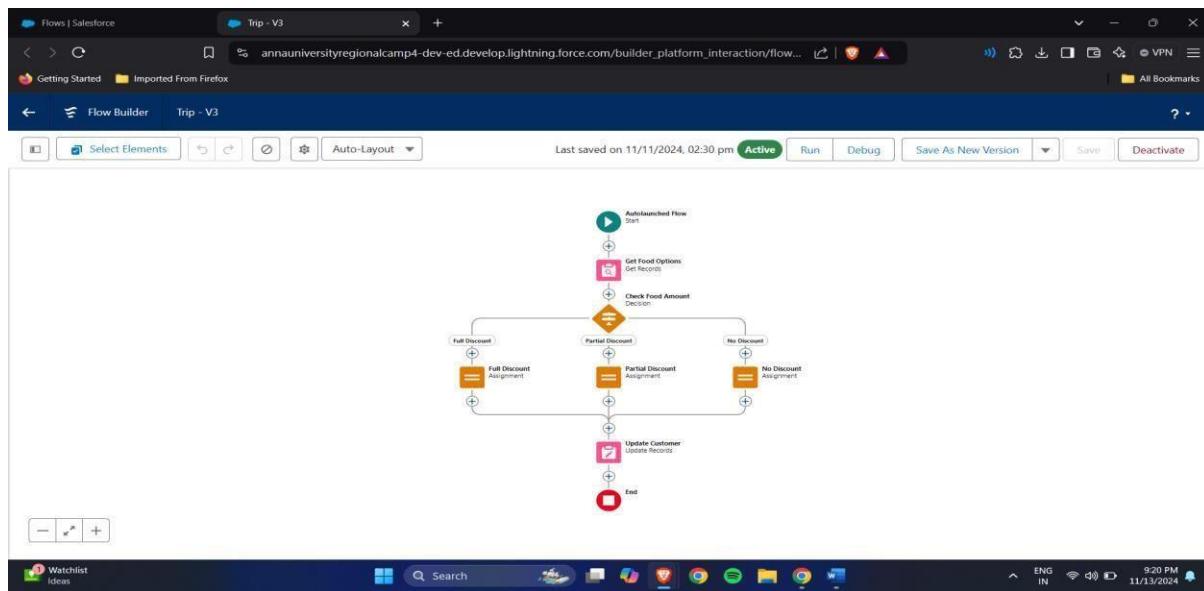




Purpose:

A "Update Record" element in a flow (commonly in CRM platforms like Salesforce) is to modify existing records in the database based on specified criteria and conditions. This element allows automation of record updates within workflows, saving time and ensuring data consistency.

Final Output of the Flow activate:



Tabs in Salesforce provide a user interface for managing and viewing records.

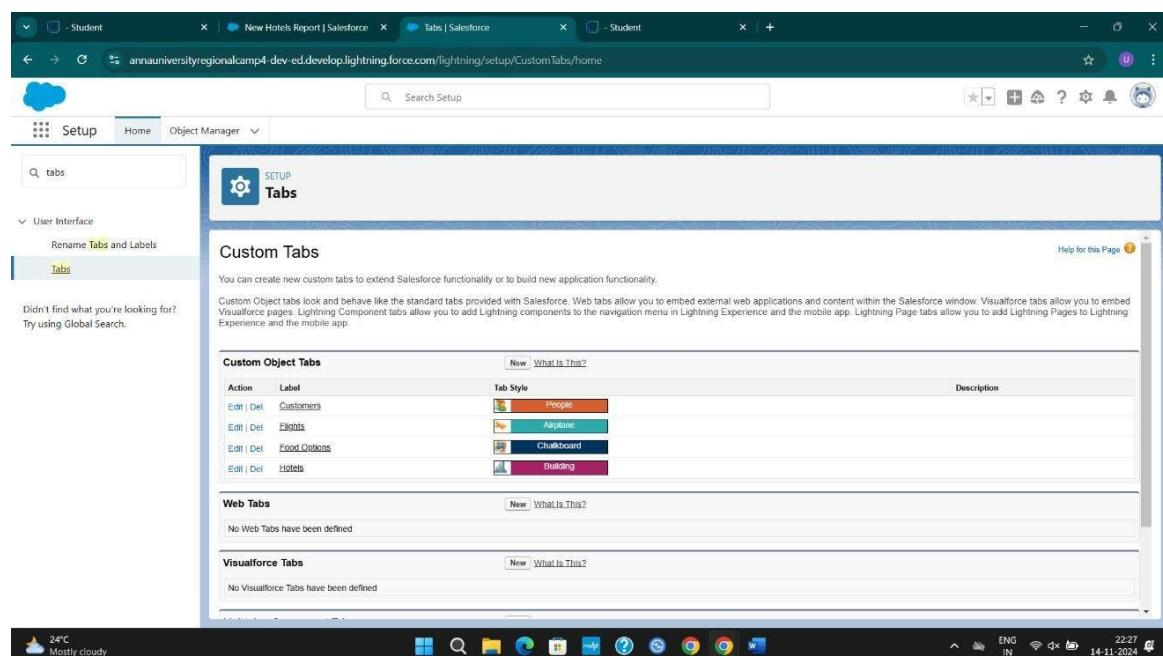
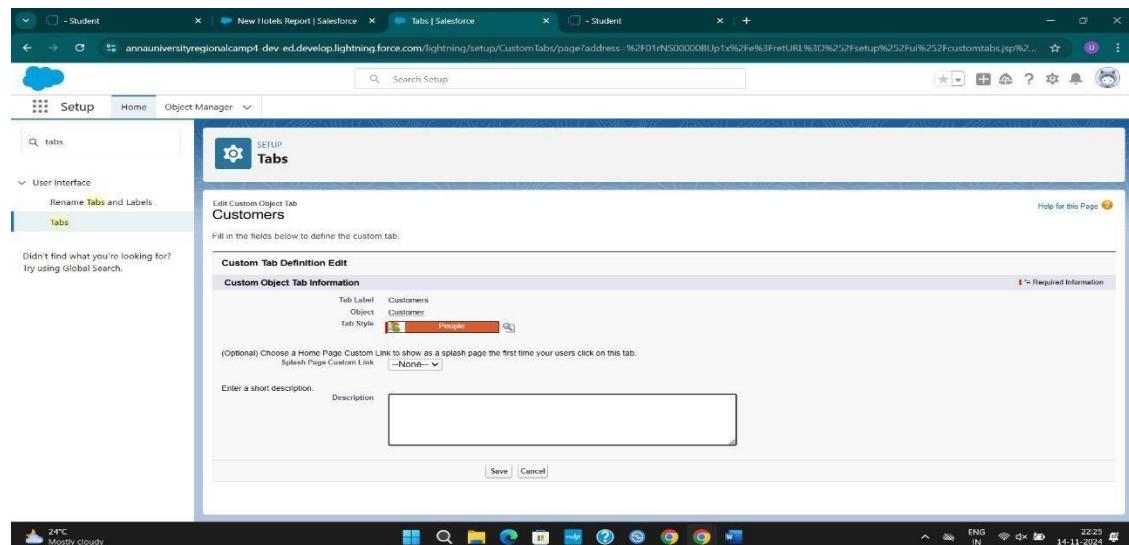
1. Types of Tabs:

- Custom Tabs: Specific to custom objects.
- Web Tabs: Display web content.
- Visualforce Tabs: Display Visualforce pages.
- Lightning Component Tabs: Add Lightning components to the navigation.

Use Case:

Creating objects and storing TripAdvisor E-management data is the first step to meet their requirements. To enable employees to access stored data efficiently, the admin needs to create dedicated tabs. By designing specific tabs, the organization can enhance the user experience, streamline navigation features. This approach helps employees find and manage data efficiently, supporting better service and operational effectiveness within TripAdvisor E-management.

Creating a Custom Tab



- From Setup, search Tabs and select New (Custom Object Tab).
- Choose Opportunity Automobile and complete the setup.

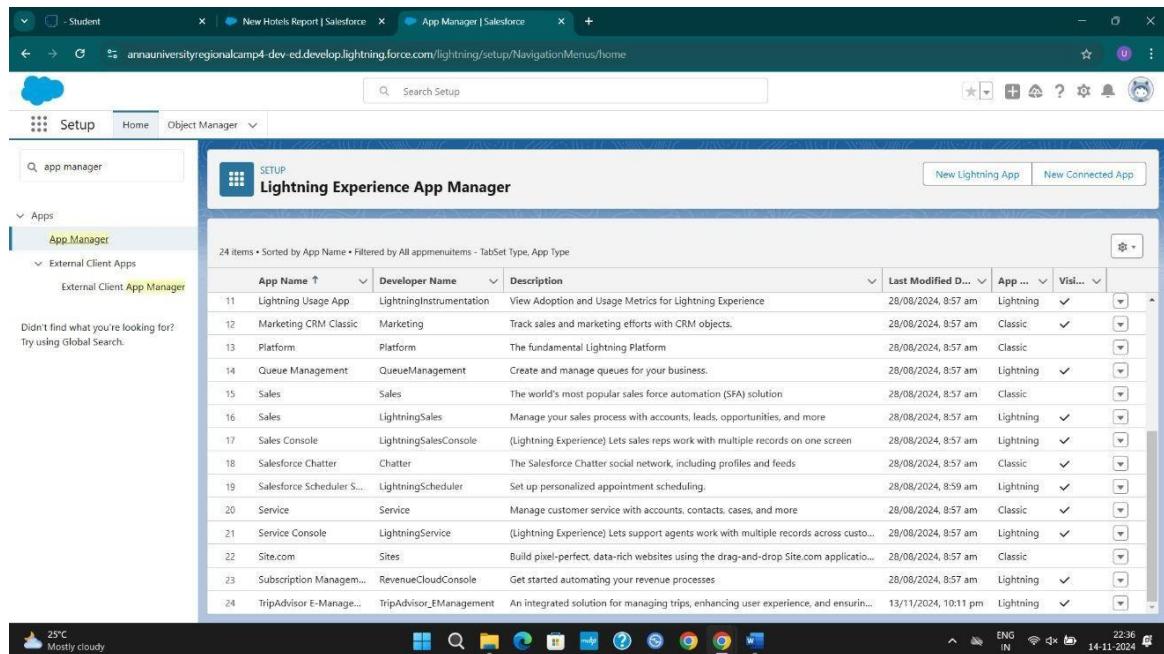
Use case:

Well done! You're close to meeting the requirements of TripAdvisor E-Management by creating objects to store the organization's data effectively. However, building a database alone is not enough to fully meet organizational needs. The real challenge lies in ensuring that users within TripAdvisor E-Management can easily access and interact with the objects you've created for them.

As the Admin for TripAdvisor E-Management, it's your responsibility to ensure that every user in the organization has appropriate access to the data modeling structure, enabling them to retrieve, update, and manage the data they need seamlessly. This will help TripAdvisor E-Management operate efficiently and provide users with a smooth experience as they engage with the system.

Create a Lightning App To create a lightning app page:

- 1.Go to setup page → search “app manager” in quick find → select “app manager” → click on New lightning App.

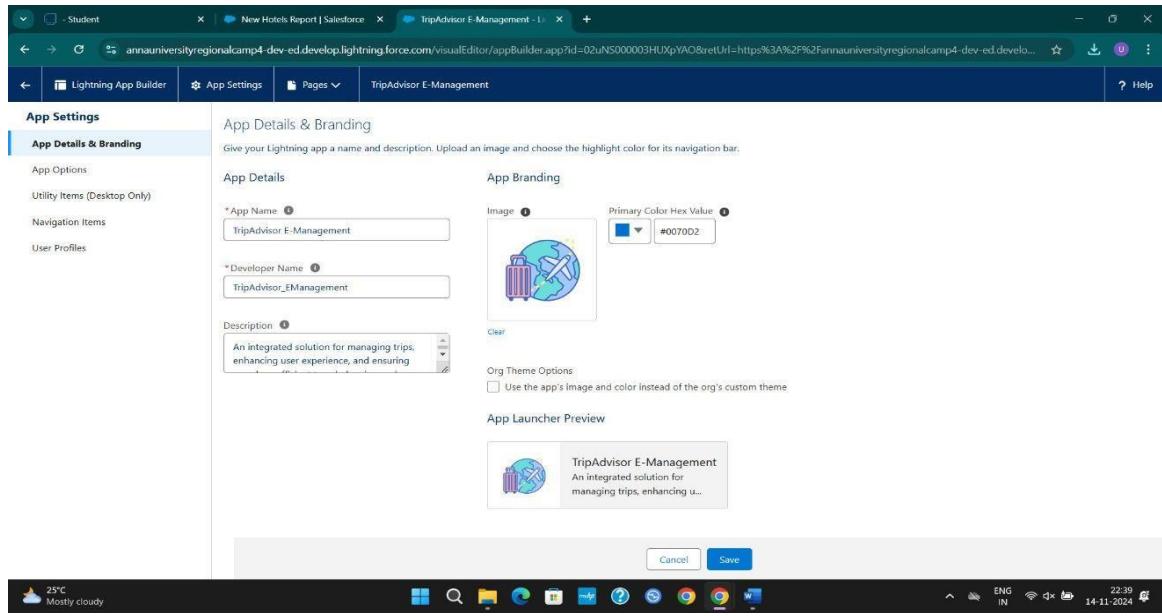


- 2.Fill the app name in app details and branding as follow App Name : TripAdvisor E-Management.

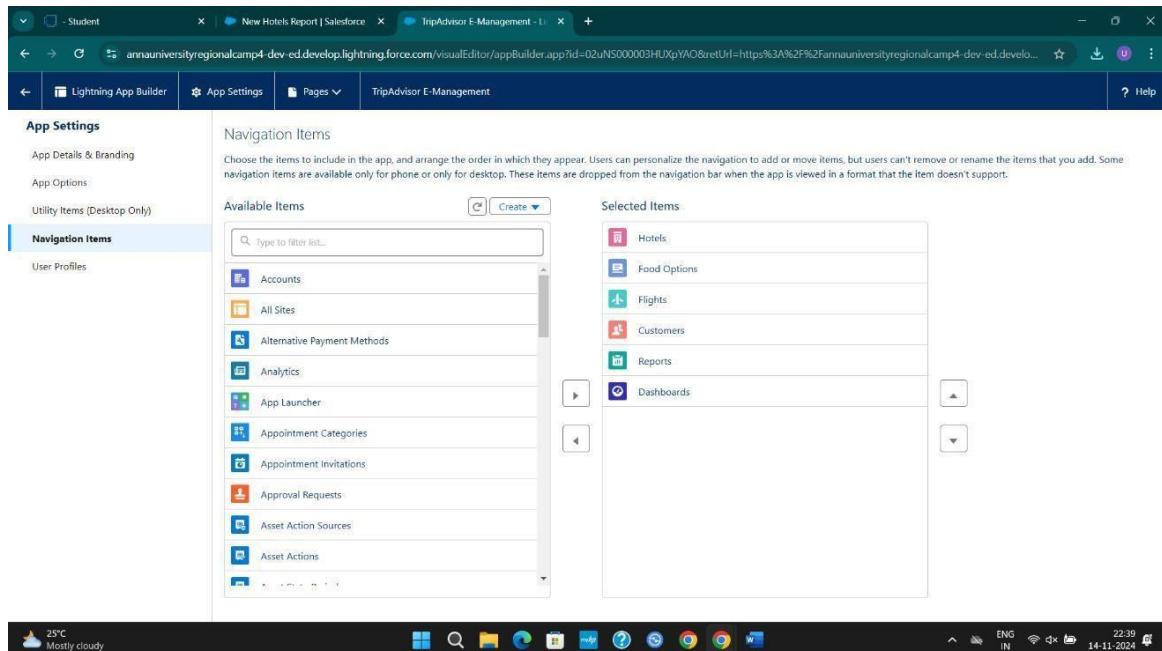
Developer Name : this will auto populated

Description : Give a meaningful description

Image : optional (if you want to give any image you can otherwise not mandatory) Primary color hex value : keep this default



3. Then click Next → (App option page) keep it as default → Next → (Utility Items) keep it as default → Next.



4. To Add Navigation Items:

5. Search profiles (System administrator) in the search bar → click on the arrow button → save & finish.

OWD Setting:

Use case:

TripAdviser E-Management, **Organization-Wide Defaults (OWDs)** are the foundational security settings that determine access to data across the system. OWDs are used to control who can access specific information within the platform. You can extend or restrict access through additional methods such as sharing rules, role hierarchies, team structures, and account groups, as well as manual sharing options.

Create OWD Setting

1. Go to Set Up → in the Quick Find box type Sharing Settings → click on it.
2. Click Edit in the Organization-Wide Defaults area.

Object	Default Internal Access	Default External Access	Grant Access Using Hierarchies
Lead	Public Read/Write/Transfer	Private	✓
Account and Contract	Public Read/Write	Private	✓
Contact	Controlled by Parent	Controlled by Parent	✓
Order	Controlled by Parent	Controlled by Parent	✓
Asset	Controlled by Parent	Controlled by Parent	✓
Opportunity	Public Read/Write	Private	✓
Case	Public Read/Write/Transfer	Private	✓
Campaign	Public Full Access	Private	✓
Campaign Member	Controlled by Campaign	Controlled by Campaign	✓
User	Public Read Only	Private	✓

3. Search for the Employee object.

4. Under default internal access and default external access change the options to “Private” and under grant access using hierarchies select the check box.
5. Click on save.
6. This Setting is for all the Users Which have been Created.

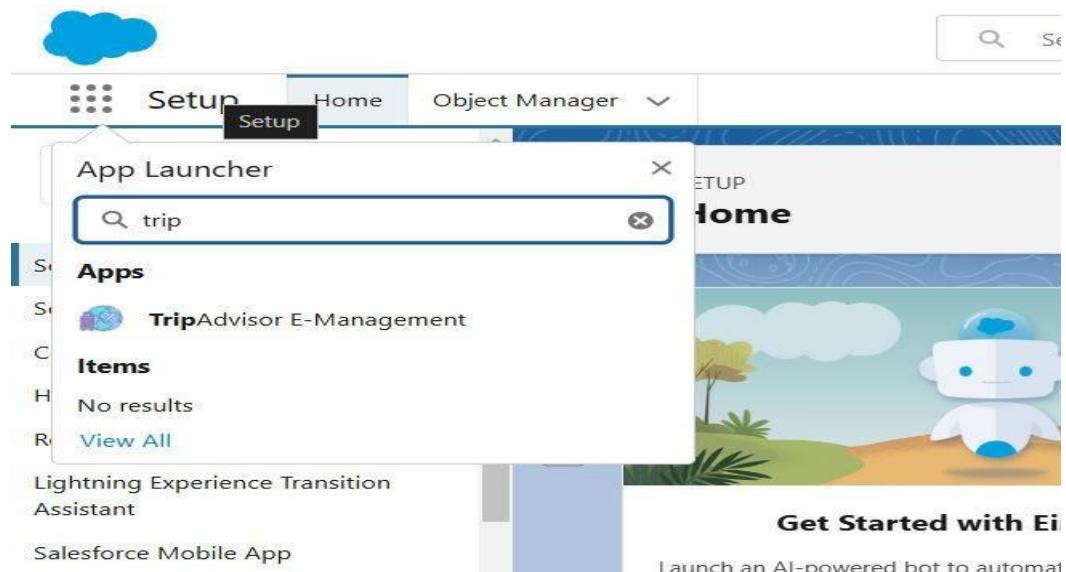
User Adoption:

Use Case:

As a new Administrator in the TripAdvisor E-management system, you handle user management tasks such as creating and editing user accounts, resetting passwords, assigning permissions, configuring access to travel data, and more. In this unit, you will learn about managing users and adding them to your TripAdvisor E-management platform.

Create a Record (Employee)

1. Click on App Launcher on the left side of the screen.
2. Search Employee Management System & click on it.



3. Click on the Employee tab.
4. Click New.
5. Fill the Details and click on Save.

The screenshot shows the 'Edit Jothi Hotel' page. At the top, it says 'Edit Jothi Hotel'. Below that, there's a note: '* = Required Information'. The form fields include:

- * Hotel Name: Jothi Hotel
- Owner: Umar Ahmed Khan A
- TotalFoodOptions: 8
- Date: 14/11/2024
- Created By: Umar Ahmed Khan A, 14/11/2024, 11:02 am
- Last Modified By: Umar Ahmed Khan A, 14/11/2024, 11:24 am

 At the bottom, there are three buttons: 'Cancel', 'Save & New', and a blue 'Save' button.

Create at least 10 records for each of the objects: Hotel, Flight, Customers, Food Options.

- Student Recently Viewed | Hotels | Sales +

annauniversityregionalcamp4-dev-ed.develop.lightning.force.com/lightning/o/Hotel_c/list?filterName=_Recent

TripAdvisor E-Mana... Hotels Flights Customers Food Options Reports Dashboards

Hotels Recently Viewed

5 items • Updated a minute ago

Hotel Name

1 Jothi Hotel

2 PK illam

3 Arand Hotel

4 Pacha Elai

5 Maria Lodge

- Student Recently Viewed | Flights | Sales +

annauniversityregionalcamp4-dev-ed.develop.lightning.force.com/lightning/o/Flight_c/list?filterName=_Recent

TripAdvisor E-Mana... Hotels Flights Customers Food Options Reports Dashboards

Flights Recently Viewed

5 items • Updated a few seconds ago

Flight Name

1 FL- 0007

2 FL- 0003

3 FL- 0005

4 FL- 0004

5 FL- 0001

- Student Recently Viewed | Customers | +

annauniversityregionalcamp4-dev-ed.develop.lightning.force.com/lightning/o/Customer_c/list?filterName=_Recent

TripAdvisor E-Mana... Hotels Flights Customers Food Options Reports Dashboards

Customers Recently Viewed

5 items • Updated a few seconds ago

Customer Name

1 Siva

2 Tamil

3 Prasanth

4 Subash

5 elayabarathi

Apex Trigger

An Apex Trigger was implemented to update hotel information whenever a new food option is added or updated. This ensures the hotel's total food options count reflects all associated food options.

Apex Trigger Handler:

```

1  public class FoodOptionTriggerHandler {
2      public static void updateHotelInformation(List<Food_Option__c> newFoodOptions, List<Food_Option__c> oldFoodOptions, String operation) {
3          Set<Id> hotelIdsToUpdate = new Set<Id>();
4          // Collect unique Hotel IDs from new Food Options (insert or update)
5          if (newFoodOptions != null) {
6              for (Food_Option__c foodOption : newFoodOptions) {
7                  if (foodOption.Hotel__c != null) {
8                      hotelIdsToUpdate.add(foodOption.Hotel__c);
9                  }
10             }
11         }
12         // Collect unique Hotel IDs from old Food Options (update or delete)
13         if (oldFoodOptions != null) {
14             for (Food_Option__c foodOption : oldFoodOptions) {
15                 if (foodOption.Hotel__c != null) {
16                     hotelIdsToUpdate.add(foodOption.Hotel__c);
17                 }
18             }
19         }
20         if (hotelIdsToUpdate.isEmpty()) {
21             return;
22         }
23         // Query the affected Hotel records
24         List<Hotel__c> hotelsToUpdate = [SELECT Id, TotalFoodOptions__c FROM Hotel__c WHERE Id IN :hotelIdsToUpdate];
25         // Recalculate the total food options count for each hotel
26         for (Hotel__c hotel : hotelsToUpdate) {
27             Integer totalFoodOptions = [SELECT COUNT() FROM Food_Option__c WHERE Hotel__c = :hotel.Id];
28             hotel.TotalFoodOptions__c = totalFoodOptions;
29         }
30         // Update the Hotel records with the new total count
31         if (!hotelsToUpdate.isEmpty()) {
32             update hotelsToUpdate;
33         }
34     }
35 }

```

An **Apex Trigger Handler** is a design pattern used to organize and manage the logic of an Apex trigger. It helps in maintaining clean, reusable, and easily maintainable code. Instead of placing the logic directly within the trigger, it delegates it to a handler class

```
1 trigger FoodOptionTrigger on Food_Option__c (after insert, after update, after delete) {
2
3     // After Insert
4     if (Trigger.isInsert & Trigger.isAfter) {
5         FoodOptionTriggerHandler.updateHotelInformation(Trigger.new, null, 'insert');
6     }
7
8     // After Update
9     if (Trigger.isUpdate & Trigger.isAfter) {
10        FoodOptionTriggerHandler.updateHotelInformation(Trigger.new, Trigger.old, 'update');
11    }
12
13    // After Delete
14    if (Trigger.isDelete & Trigger.isAfter) {
15        FoodOptionTriggerHandler.updateHotelInformation(null, Trigger.old, 'delete');
16    }
17 }
```

Logs Tools Checkpoints Query Editor View Status Progress Problems

File Edit Debug Test Workspaces Help

FoodOptionTriggerHandler.apex | FoodOptionTriggerTest.apex

Code Coverage: None API Version: 62

Run Test Go To

Logs Tools Checkpoints Query Editor View Status Progress Problems

File Edit Debug Test Workspaces Help

FoodOptionTriggerHandler.apex | FoodOptionTriggerTest.apex

23°C Mostly cloudy

An **Trigger** in Salesforce is a piece of code that automatically executes (or "fires") when a specific event occurs on a record in Salesforce, such as creating, updating, or deleting a record. It allows developers to add custom logic to standard operations, providing more control over data and business processes.

Test Trigger:

```
1 @isTest
2 public class FoodOptionTriggerTest {
3
4     @testSetup
5     static void setupTestData() {
6         // Create a test Hotel record
7         Hotel__c testHotel1 = new Hotel__c(Name = 'Hotel A');
8         insert testHotel1;
9
10        // Create Food Option records for testHotel1
11        List<Food_Option__c> foodOptions = new List<Food_Option__c>{
12            new Food_Option__c(Name__c = 'Pizza', Hotel__c = testHotel1.Id),
13            new Food_Option__c(Name__c = 'Burger', Hotel__c = testHotel1.Id)
14        };
15        insert foodOptions;
16    }
17
18    @isTest
19    static void testAfterInsert() {
20        // Fetch the Hotel record before insert
21        Hotel__c testHotel = [SELECT Id, TotalFoodOptions__c FROM Hotel__c WHERE Name = 'Hotel A' LIMIT 1];
22        System.assertEquals(2, testHotel.TotalFoodOptions__c, 'Initial count should be 2');
23
24        // Insert a new Food Option
25        Food_Option__c newFoodOption = new Food_Option__c(Name__c = 'Pasta', Hotel__c = testHotel.Id);
26        insert newFoodOption;
27
28        // Verify the count after insert
29        testHotel = [SELECT Id, TotalFoodOptions__c FROM Hotel__c WHERE Id = :testHotel.Id];
30        System.assertEquals(3, testHotel.TotalFoodOptions__c, 'Count should increase to 3 after insert');
31    }
32 }
```

Logs Tools Checkpoints Query Editor View Status Progress Problems

File Edit Debug Test Workspaces Help

FoodOptionTriggerHandler.apex | FoodOptionTriggerTest.apex

Code Coverage: None API Version: 62

Run Test Go To

Logs Tools Checkpoints Query Editor View Status Progress Problems

File Edit Debug Test Workspaces Help

FoodOptionTriggerHandler.apex | FoodOptionTriggerTest.apex

23°C Mostly cloudy

A **Test Trigger** in Salesforce is used to validate that the trigger behaves as expected under different conditions. It is written using Apex test methods to simulate various scenarios, ensuring that triggers perform the correct operations, like inserting, updating, or deleting records.

The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is annauniversityregionalcamp4-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCIPage. The tabs at the top are 'FoodOptionTriggerHandler.apxc' and 'FoodOptionTriggerTest.apxc'. The code editor contains an Apex class named 'FoodOptionTriggerTest' with two test methods: 'setupTestData()' and 'testAfterInsert()'. Below the code editor is a 'Logs' tab showing a 'Test Run' with two entries. The first entry is a success (0 failures, 1 total) from Nov 14 2024 at 22:14:15 GM.. The second entry is another success (0 failures, 1 total) from Nov 14 2024 at 22:16:44 GM.. The status bar at the bottom shows the date as 14-11-2024 and the time as 22:16.

```

1  @isTest
2  public class FoodOptionTriggerTest {
3
4      @testSetup
5      static void setupTestData() {
6          // Create a test Hotel record
7          Hotel__c testHotel1 = new Hotel__c(Name = 'Hotel A');
8          insert testHotel1;
9
10         // Create Food Option records for testHotel1
11         List<Food_Option__c> foodOptions = new List<Food_Option__c>{
12             new Food_Option__c(Name__c = 'Pizza', Hotel__c = testHotel1.Id),
13             new Food_Option__c(Name__c = 'Burger', Hotel__c = testHotel1.Id)
14         };
15         insert foodOptions;
16     }
17
18     @isTest
19     static void testAfterInsert() {
20         // Fetch the Hotel record before insert
21     }
}

```

A "Test Trigger case run successfully" means that a trigger (an automated process or function) was executed, and it completed without errors or failures.

- **Trigger:** A piece of code that runs automatically in response to specific events (like creating, updating, or deleting a record).
- **Test Case:** A scenario designed to verify that the trigger works as expected under certain conditions.
- **Successful Run:** The trigger was executed correctly, and the desired results were achieved, with no issues encountered during testing (e.g., data was updated correctly, no errors occurred).

Apex Scheduled

Apex Scheduled Class for Flight Reminders

The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is annauniversityregionalcamp4-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCIPage. The tabs at the top are 'FlightReminderScheduledJob.apxc' and 'FlightReminderScheduledJob.apxc'. The code editor contains an Apex class named 'FlightReminderScheduledJob' with a single method 'execute'. Below the code editor is a 'Logs' tab showing a log entry for 'Umar Ahmed Khan A' at 11/14/2024, 7:12:54 PM. The operation was '/services/data/v52.0/tooling/executeAnonymous'. The status was 'Success'. The status bar at the bottom shows the date as 14-11-2024 and the time as 19:12.

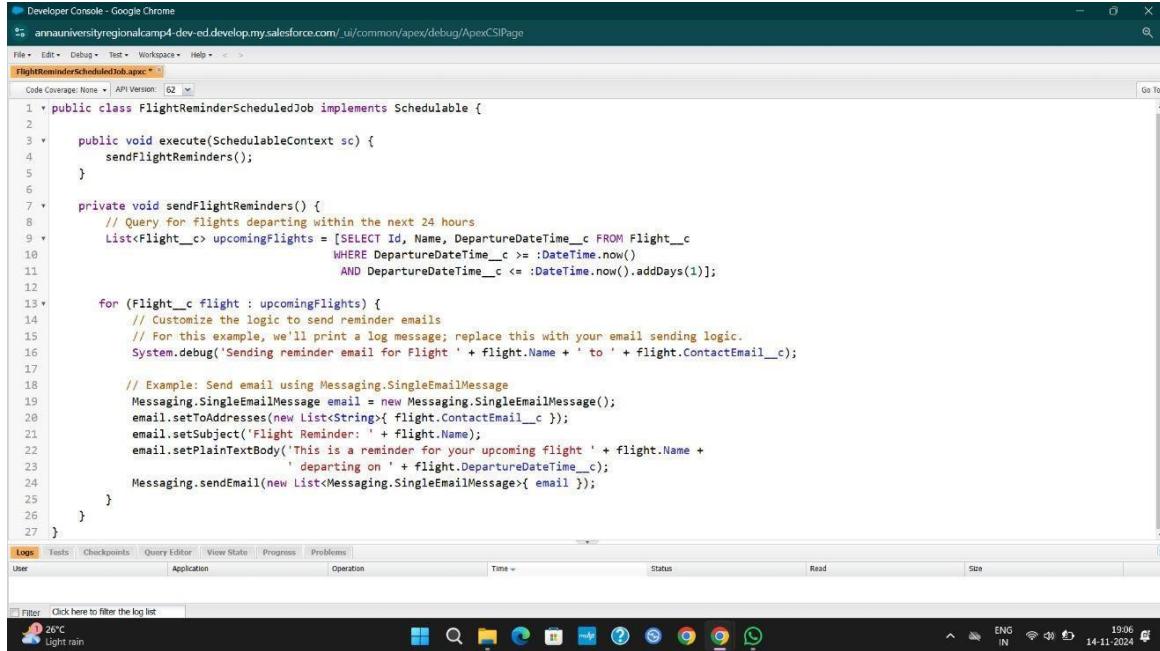
```

1  public class FlightReminderScheduledJob implements Schedulable {
2
3      public void execute(SchedulableContext sc) {
4          sendFlightReminders();
5      }
6
7      private void sendFlightReminders() {
8          // Query for flights departing within the next 24 hours
9          List<Flight__c> upcomingFlights = [SELECT Id, Name, DepartureDateTime__c FROM Flight__c
10                                         WHERE DepartureDateTime__c >= :DateTime.now()
11                                           AND DepartureDateTime__c <= :DateTime.now().addDays(1)];
12
13      for (Flight__c flight : upcomingFlights) {
14          // Customize the logic to send reminder emails
15          // For this example, we'll print a log message; replace this with your email sending logic.
16          System.debug('Sending reminder email for Flight ' + flight.Name + ' to ' + flight.ContactEmail__c);
17
18          // Example: Send email using Messaging.SingleEmailMessage
19          Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
20          email.setToAddresses(new List<String>{ flight.ContactEmail__c });
21          email.setSubject('Flight Reminder: ' + flight.Name);
22          email.setPlainTextBody('This is a reminder for your upcoming flight ' + flight.Name +
23                                ' departing on ' + flight.DepartureDateTime__c);
24      }
}

```

A **Scheduled Class** in platforms like Salesforce allows you to automate and schedule the execution of Apex classes at specified times or intervals. It is particularly useful for recurring

tasks, such as sending emails, updating records, or integrating external systems, without manual intervention.

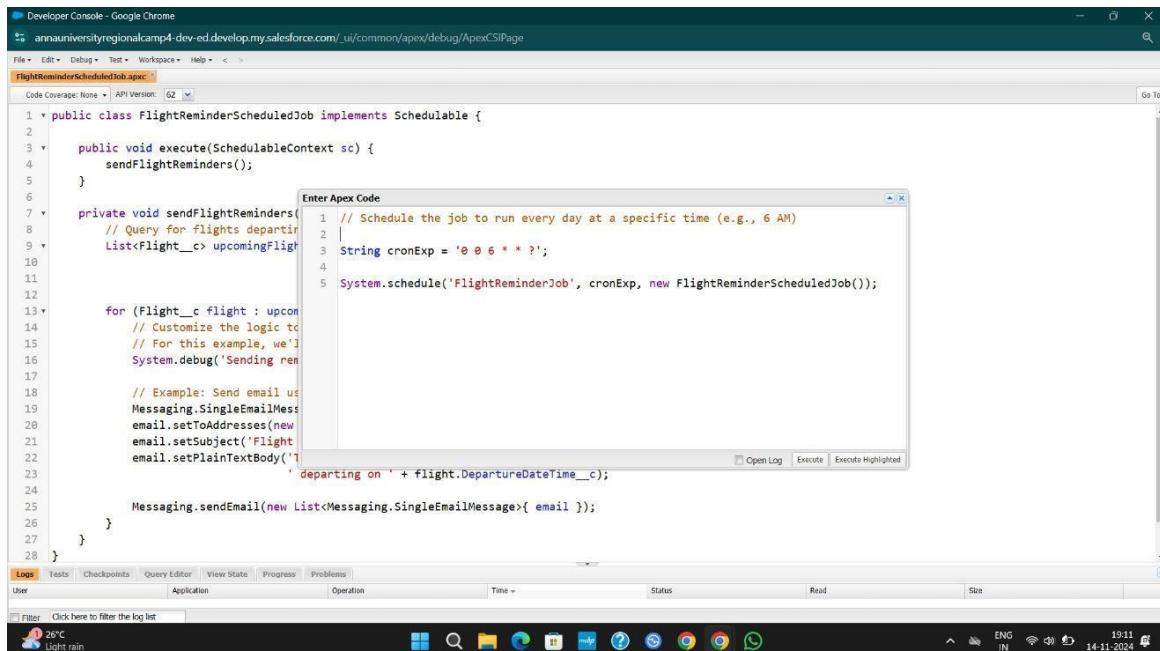


The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is https://annaundergraduatecamp4-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexConsolePage. The code editor displays the `FlightReminderScheduledJob` class:

```
1 * public class FlightReminderScheduledJob implements Schedulable {
2
3     public void execute(SchedulableContext sc) {
4         sendFlightReminders();
5     }
6
7     private void sendFlightReminders() {
8         // Query for flights departing within the next 24 hours
9         List<Flight__c> upcomingFlights = [SELECT Id, Name, DepartureDateTime__c FROM Flight__c
10                                         WHERE DepartureDateTime__c >= :DateTime.now()
11                                           AND DepartureDateTime__c <= :DateTime.now().addDays(1)];
12
13     for (Flight__c flight : upcomingFlights) {
14         // Customize the logic to send reminder emails
15         // For this example, we'll print a log message; replace this with your email sending logic.
16         System.debug('Sending reminder email for Flight ' + flight.Name + ' to ' + flight.ContactEmail__c);
17
18         // Example: Send email using Messaging.SingleEmailMessage
19         Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
20         email.setToAddresses(new List<String>{ flight.ContactEmail__c });
21         email.setSubject('Flight Reminder: ' + flight.Name);
22         email.setPlainTextBody('This is a reminder for your upcoming flight ' + flight.Name +
23                               ' departing on ' + flight.DepartureDateTime__c);
24         Messaging.sendEmail(new List<Messaging.SingleEmailMessage>{ email });
25     }
26 }
27 }
```

The console interface includes tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. Below the code editor is a log viewer with a filter bar and a status bar at the bottom showing system information.

Scheduling a Job Code refers to setting up an automated process to run at a specific time or interval without manual intervention. This is commonly used in systems like CRM platforms, databases, or job scheduling tools. The purpose is to perform repetitive tasks (e.g., data updates, reports generation) at predefined times.



The screenshot shows the Salesforce Developer Console with the `FlightReminderScheduledJob` code. A modal dialog titled "Enter Apex Code" is open over the code editor, containing the following cron expression:

```
1 // Schedule the job to run every day at a specific time (e.g., 6 AM)
2 |
3 String cronExp = '0 0 6 * * ?';
4
5 System.schedule('FlightReminderJob', cronExp, new FlightReminderScheduledJob());
```

The main code editor below the dialog contains the original `FlightReminderScheduledJob` code. The status bar at the bottom indicates the date and time as 14-11-2024 19:11.

A Scheduled Job Code is a task or process that runs automatically at predefined times or intervals. When it runs successfully, it means the scheduled job has executed without errors, completing the intended task as per the schedule.

The screenshot shows the Salesforce 'Scheduled Jobs' page. The left sidebar has a 'Scheduled Jobs' section under 'Jobs'. The main area displays a table titled 'All Scheduled Jobs' with columns: Action, Job Name, Submitted By, Submitted, Started, Next Scheduled Run, Type, and Cron Trigger ID. The table lists three jobs: 'FlightReminderJob', 'Metalytics Data Loader Job for Org ...', and 'Program Milestone Computation Cron Job'. A status message at the top says 'Percentage of Scheduled Jobs Used: 1%'.

"All jobs scheduled have been updated" means that the tasks or processes that were planned or set to run at specific times have been modified or refreshed. This could involve changing the timing, parameters, or details of the scheduled jobs to ensure they align with new requirements, improve efficiency, or reflect updated information.

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

1. Tabular
2. Summary
3. Matrix
4. Joined Reports

The screenshot shows the 'Create Report' page. On the left, there's a sidebar with a 'Category' section listing various report types like Accounts & Contacts, Opportunities, etc. The main area has a 'Select a Report Type' search bar with 'hotel' typed in. Below it, a table shows 'Report Type Name' and 'Category' for 'Hotels' and 'Food Options with Hotel'. To the right, a 'Details' panel shows 'Hotels' as the 'Standard Report Type' with a 'Start Report' button. It also includes sections for 'Created By You' (New Hotels Report), 'Created By Others' (No Reports Yet), and 'Objects Used in Report Type' (Owner).

4. Customize your report and add fields from left pane as shown below

5. Save or run it.

Reports for Hotel, Flight, Customer, and Food Option have been created using a standardized method, streamlining data management and analysis for improved decision-making and business operations.

A new hotel report has been created, streamlining data management and providing valuable insights for decision-making.

A new flight report has been created, providing updated and relevant data for better analysis and decision-making.

	Customer: Customer Name	Discount Amount	Discount Percent
1	Elayabharathi	₹10	9%
2	Subash	₹30	30%
3	Prasanth	₹35	20%
4	Tamil	₹69	9%
5	Sivakumaran	₹50	14%
6		₹194	82%

A new customer report has been created to manage and analyze customer data efficiently.

	Food Option: Food Option Name	Hotel	Name
1	FO - 0003	PK illam	Idly
2	FO - 0004	Maria Lodge	Idly
3	FO - 0009	Maria Lodge	Dosa
4	FO - 0001	Jothi Hotel	EB Dosa
5	FO - 0026	Jothi Hotel	Tandoori
6	FO - 0011	Pacha Elai	poori
7	FO - 0027	Jothi Hotel	Pasta
8	FO - 0022	Maria Lodge	Parotta
9	FO - 0017	PK illam	Biryani
10	FO - 0019	Anand Hotel	Biryani
11	FO - 0028	Jothi Hotel	Noodles
12	FO - 0023	PK illam	Parotta
13	FO - 0002	Pacha Elai	Idly
14	FO - 0014	Anand Hotel	poori

A new FoodOption report has been created to streamline the management and analysis of food-related data.

1. Create a report with report type: “TripAdviser E-Management and Projects”.

Dashboards:

Dashboards provide a visual summary of real-time data, enabling users to quickly understand business trends, monitor performance, and make informed decisions. They allow easy access to report data through visual components.

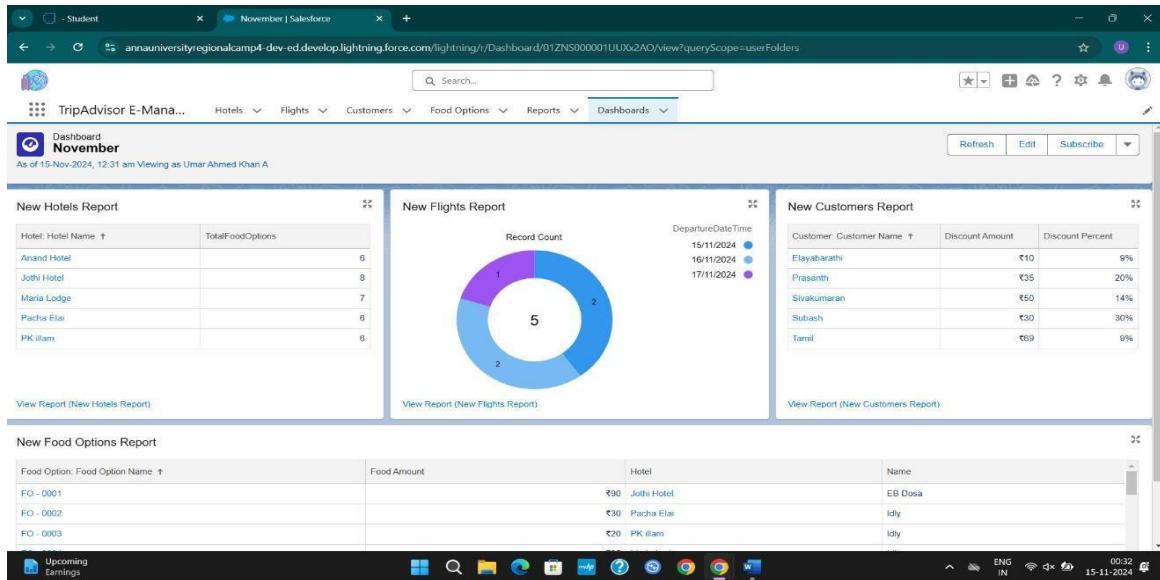
Create Dashboard

1. Go to the app → click on the Dashboards tabs.

2. Give a Name and click on Create.

3. Select add component.

4. Select a Report and click on select.



Conclusion

Summary of Achievements

The *TripAdvisor E-Management* Salesforce project successfully streamlined the management of travel-related services by implementing a comprehensive solution that integrates automation, custom user interfaces, and real-time notifications. Key achievements include:

- Developed a user-friendly CRM tailored to manage hotel, flight, and food option data efficiently.
- Automated discount calculations and flight reminder notifications, significantly reducing manual processes and improving customer satisfaction.
- Provided actionable insights into hotel occupancy, food option availability, and flight booking trends, supporting better business decisions.

This solution is scalable and adaptable, providing a robust foundation for future enhancements, such as advanced customer personalization or integration with third-party travel platforms.