TRIP ADVISOR OF E-MANAGEMENT

PROJECT DESCRIPTION :

The Garage Management System is a valuable tool for automotive repair facilities, helping them deliver top-notch service, increase operational efficiency, and build lasting customer relationships. With its user-friendly interface and powerful features, GMS empowers garages to thrive in a competitive market while ensuring a seamless and satisfying experience for both customers and staff.

PROJECT FLOW :

In this project you can do hands on practice the configuration as well as customization with the Data modelling, App building, User Adoption & Apex Code

Milestone 1: Acceptance Criteria & Solution

Milestone 2: Create Flow

Milestone 3 : Apex Tiggers

Milestone 4 : Apex Schedule

WHAT YOU`LL LEARN :

1. Real Time Salesforce Project
2. Object & Relationship in Salesforce
3. Formula fields and Validation rules.
4. Cross object formula fields.
5. Rollup summary fields.
6. Conditional formatting
7. Flows

INTRODUCTION :

Are you new to Salesforce? Not sure exactly what it is, or how to use it? Don’t know where you should start on your learning journey? If you’ve answered yes to any of these questions, then you’re in the right place. This module is for you.
Welcome to Salesforce! Salesforce is game-changing technology, with a host of productivity-boosting features, that will help you sell smarter and faster. As you work toward your badge for this module, we’ll take you through these features and answer the question, “What is Salesforce, anyway?”

Acceptance Criteria & Solution :

- As the Salesforce User we have to manage the data for the Hotels, Flights, and Food Options for this we have to create some automation for simplification.
- To ensure that when a new Food Option is added or updated, the corresponding Hotel's information is updated accordingly. For example, you might want to maintain a total count of food options for each hotel.
- Also there is automation for the customer benefits if the there buying amount is with respect to some amount then they will get some discounts on their bill
- For the flights there schedule process being involved where the customer who has booked the flight will get the reminder mail alert for knowing proper timing of the flight before 24 hrs it's important to manage the in a good way.
- The system should provide confirmation or notification to the user upon successful sending of the email.

Solution : For the Above requirements of TripAdvisor we have created the solutions by creating the custom objects and Fields the Custom Objects that are created are Hotels, Food Options, Customer & Flights. For the Automation we have used here a flow and triggers and for scheduling the email alerts we have created the Apex Schedulable class so email alerts will be created.

Create 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

With Above References Create following Object
Food Option ? Data Type ? Auto Number ? Format? FO - {0000}

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

Customer ? Text ? Field Name ? Customer Name

Create Fields for Hotel Object :

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

Create Fields For Food Option :

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

Create Fields in the Flight Object :

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

Create Fields in the Customer Object :

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

3	Discount Percent	Percentage

Create Fields in the Customer Object :

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

Create Flow:

Create the Flow for the discount for customer when the Amount is greater than 3000 some some Amount of Discounts will be there if the Amount is between 1500 to 3000 so Some Amount of Discount will be there for them

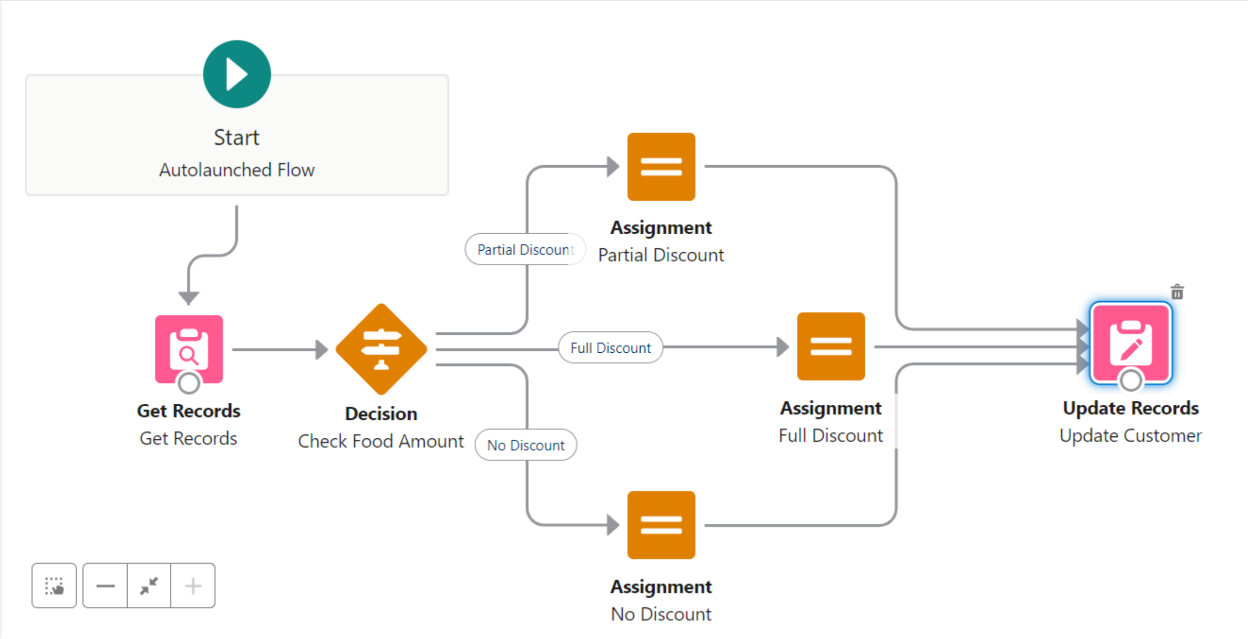
Flow Procedure :

Create 3 variable :

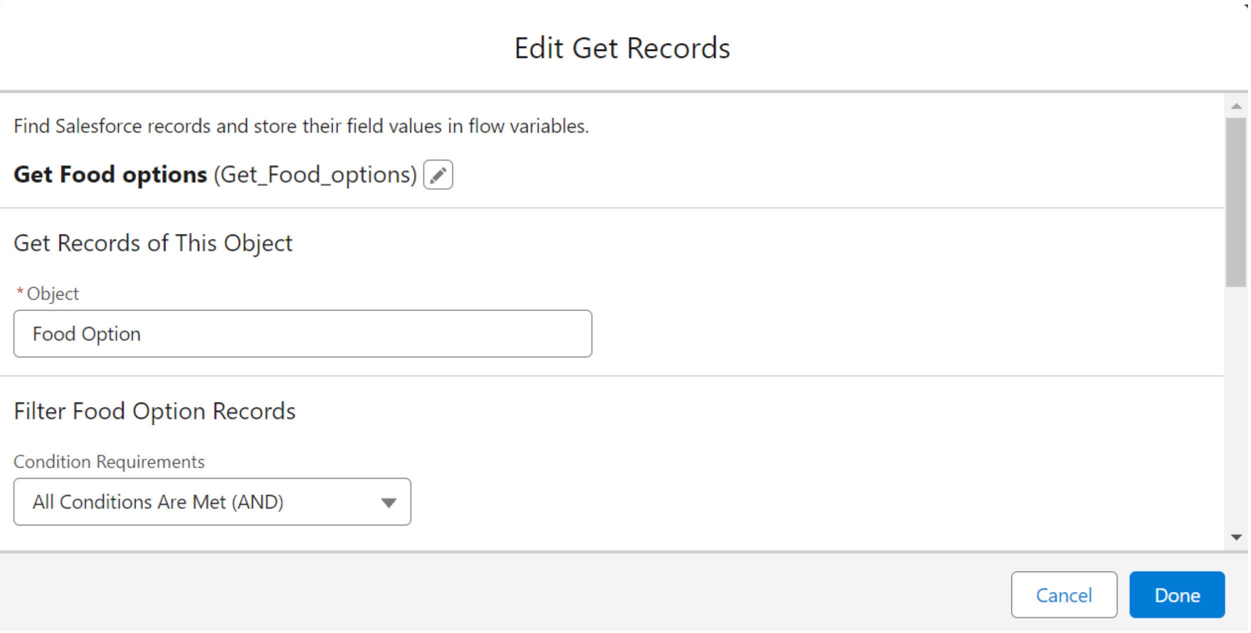
Variable > Api name > foId > text > Available for Input

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

Variable >Api name > discount > Number



Flow Steps : Get Records



Edit Get Records

Find Salesforce records and store their field values in flow variables.

Get Food options (Get_Food_options)

Get Records of This Object

*Object

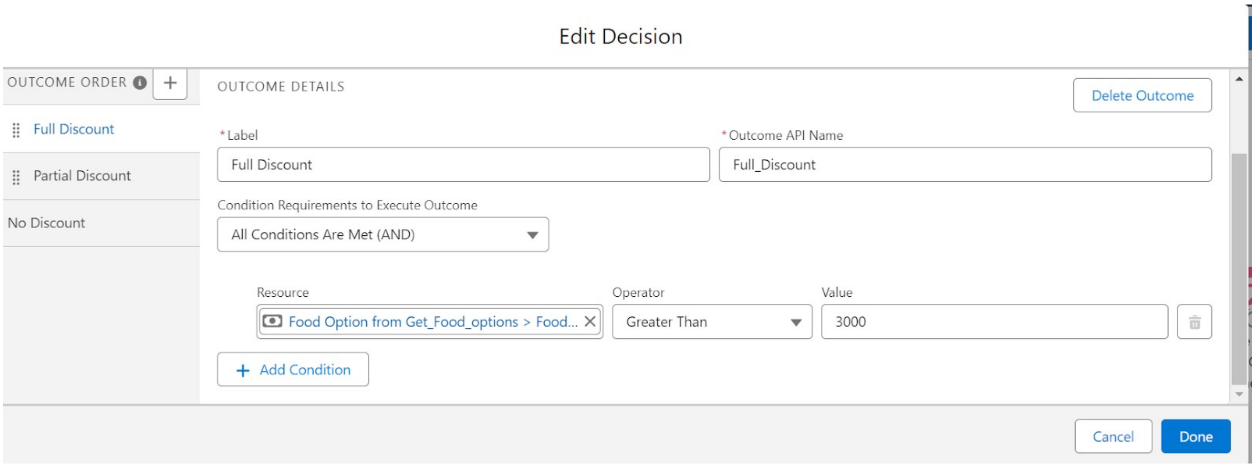
Food Option

Filter Food Option Records

Condition Requirements

All Conditions Are Met (AND)

2. Decision Element: Create 2 Outcomes



Edit Decision

OUTCOME ORDER

Full Discount

Partial Discount

No Discount

OUTCOME DETAILS

*Label: Full Discount

*Outcome API Name: Full_Discount

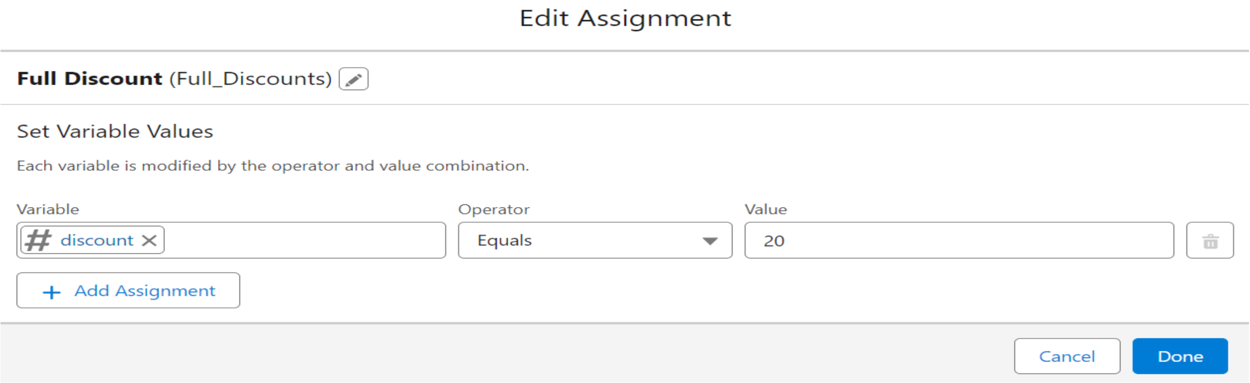
Condition Requirements to Execute Outcome: All Conditions Are Met (AND)

Resource: Food Option from Get_Food_options > Food...

Operator: Greater Than

Value: 3000

Take the 3 Assignments > Full Discount, Partial Discount & No Discount



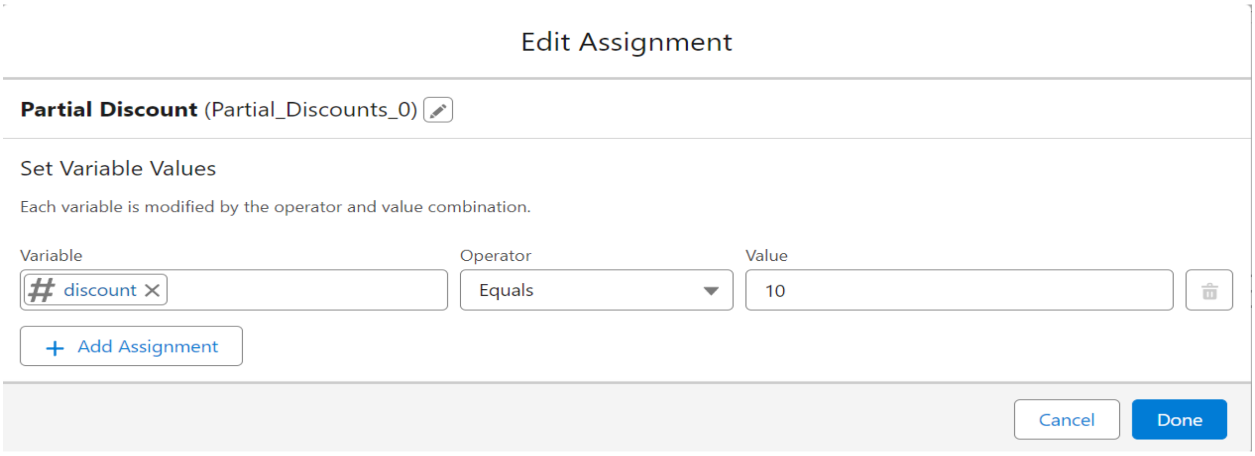
Edit Assignment

Full Discount (Full_Discounts)

Set Variable Values

Each variable is modified by the operator and value combination.

Variable	Operator	Value
discount	Equals	20



Edit Assignment

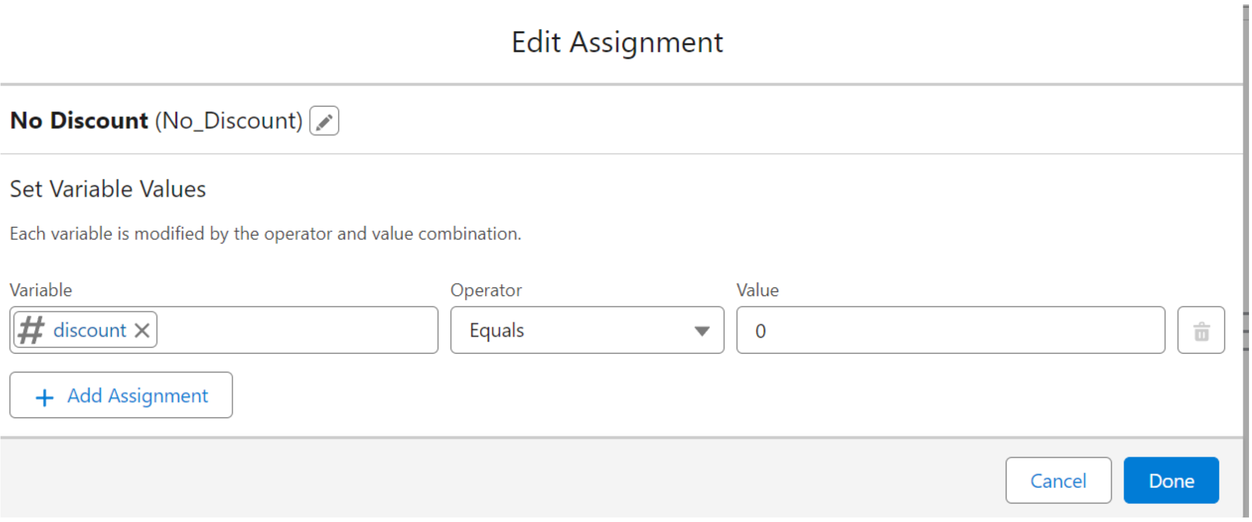
Partial Discount (Partial_Discounts_0)

Set Variable Values

Each variable is modified by the operator and value combination.

Variable	Operator	Value
discount	Equals	10

Edit Assignment



Edit Assignment

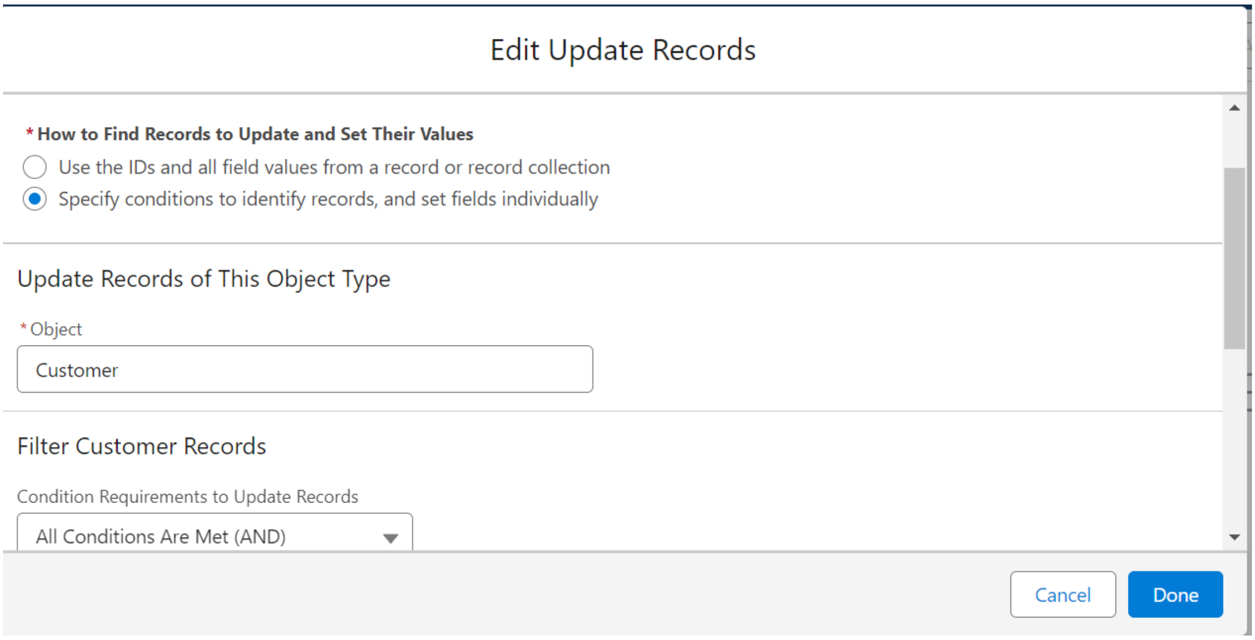
No Discount (No_Discount)

Set Variable Values

Each variable is modified by the operator and value combination.

Variable	Operator	Value
# discount	Equals	0

Update Record Element :



Edit Update Records

* **How to Find Records to Update and Set Their Values**

○ Use the IDs and all field values from a record or record collection

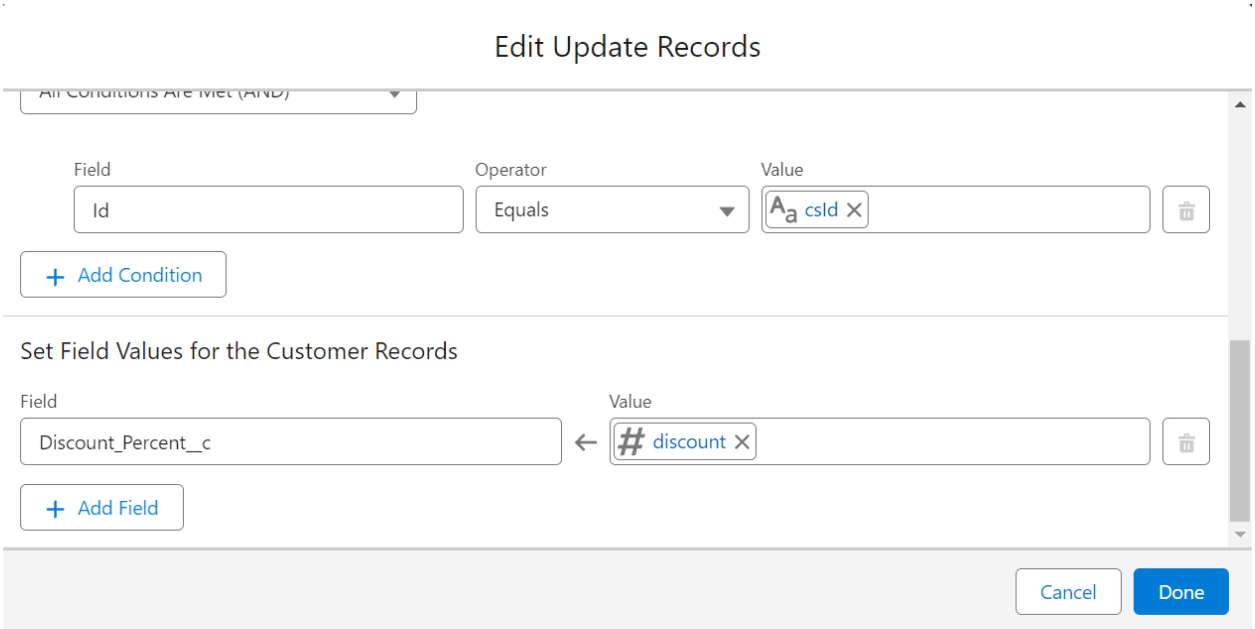
● Specify conditions to identify records, and set fields individually

Update Records of This Object Type

* Object: Customer

Filter Customer Records

Condition Requirements to Update Records: All Conditions Are Met (AND)



Edit Update Records

All Conditions Are Met (AND)

Field	Operator	Value
Id	Equals	csId

Set Field Values for the Customer Records

Field		Value
Discount_Percent_c	←	discount

Apex Triggers:

Scenario: In the Hotel you have to ensure that when a new Food Option is added or updated, the corresponding Hotel's information is updated accordingly. For example, you might want to maintain a total count of food options for each hotel. To manage the things properly with perspective to the Hotel things should be clearly manageable for making the food options available with respect to hotels

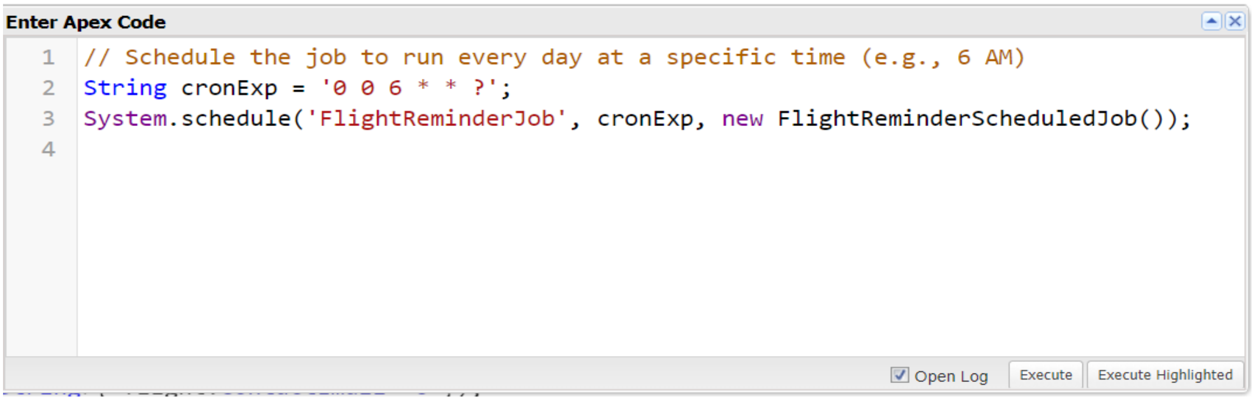
Apex trigger With Handler :

Apex Trigger With Handler

```
public class FoodOptionTriggerHandler {
   // Method to update hotel information based on food options
   public static void updateHotelInformation(List<Food_Option__c> newFoodOptions,
List<Food_Option__c> oldFoodOptions, TriggerOperation operation) {
      Set<Id> hotelIdsToUpdate = new Set<Id>();

      // Collect unique Hotel Ids affected by food options changes
```

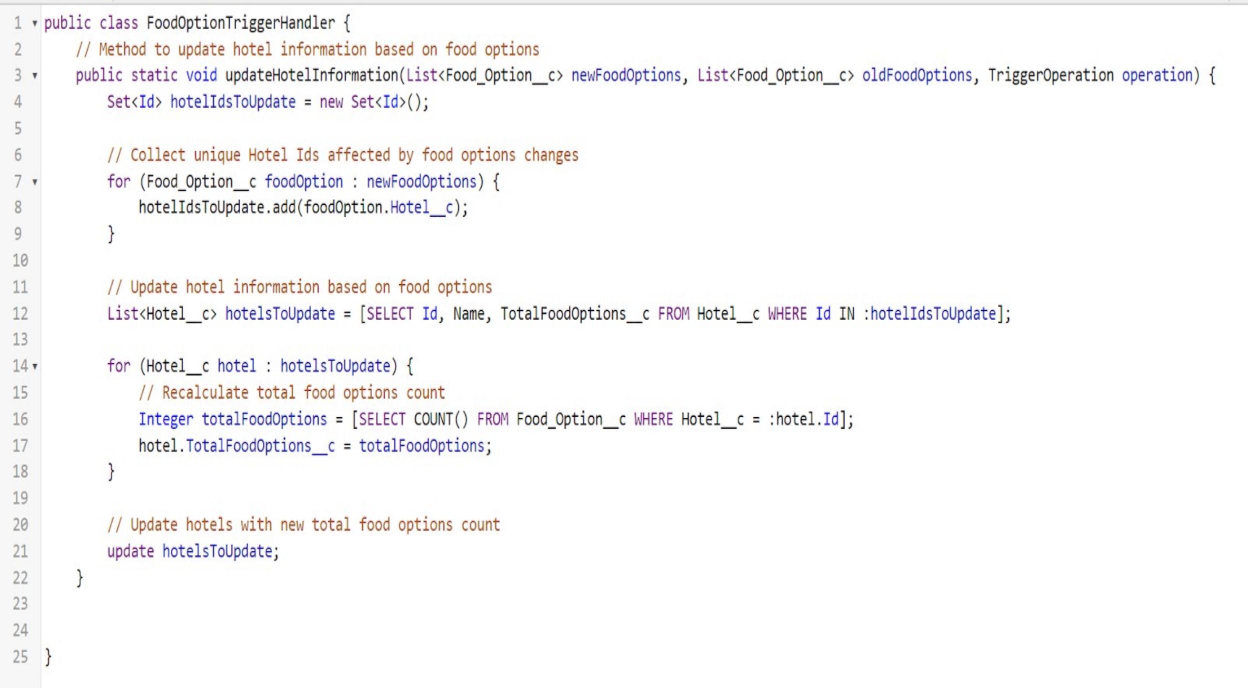
```
        for (Food_Option__c foodOption : newFoodOptions) {
            hotelIdsToUpdate.add(foodOption.Hotel__c);
        }

        // Update hotel information based on food options
        List<Hotel__c> hotelsToUpdate = [SELECT Id, Name, TotalFoodOptions__c FROM Hotel__c
WHERE Id IN :hotelIdsToUpdate];

        for (Hotel__c hotel : hotelsToUpdate) {
            // Recalculate total food options count
            Integer totalFoodOptions = [SELECT COUNT() FROM Food_Option__c WHERE
Hotel__c = :hotel.Id];
            hotel.TotalFoodOptions__c = totalFoodOptions;
        }

        // Update hotels with new total food options count
        update hotelsToUpdate;
    }

}
```

System.schedule('FlightReminderJob', cronExp, new FlightReminderScheduledJob());



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

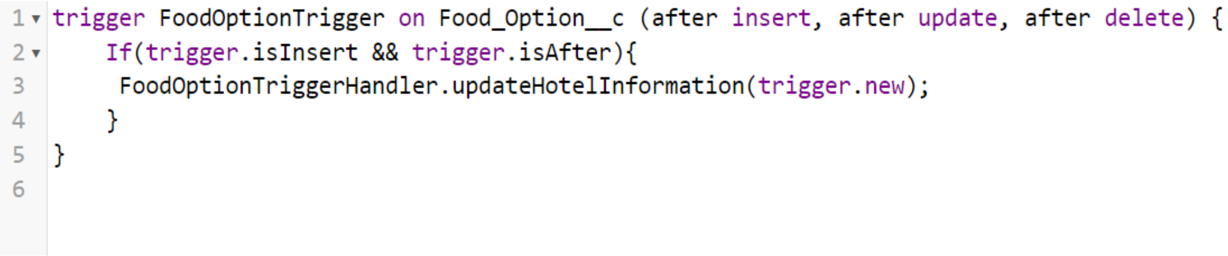
Conclusion : We have Created this Customization process for the proper flow of the business if TripAdvisor where they can easily access the Hotel requirement then food options and also the ease for the customers with the preferable discount with there Amount limits this process helps to save time from multiple manual processes.



```
public class FoodOptionTriggerHandler {
    // Method to update hotel information based on food options
    public static void updateHotelInformation(List<Food_Option__c> newFoodOptions, List<Food_Option__c> oldFoodOptions, TriggerOperation operation) {
        Set<Id> hotelIdsToUpdate = new Set<Id>();

        // Collect unique Hotel Ids affected by food options changes
        for (Food_Option__c foodOption : newFoodOptions) {
            hotelIdsToUpdate.add(foodOption.Hotel__c);
        }

        // Update hotel information based on food options
        List<Hotel__c> hotelsToUpdate = [SELECT Id, Name, TotalFoodOptions__c FROM Hotel__c WHERE Id IN :hotelIdsToUpdate];

        for (Hotel__c hotel : hotelsToUpdate) {
            // Recalculate total food options count
            Integer totalFoodOptions = [SELECT COUNT() FROM Food_Option__c WHERE Hotel__c = :hotel.Id];
            hotel.TotalFoodOptions__c = totalFoodOptions;
        }

        // Update hotels with new total food options count
        update hotelsToUpdate;
    }


}
```

Trigger

```
trigger FoodOptionTrigger on Food_Option__c (after insert, after update, after delete) {
    If(trigger.isInsert && trigger.isAfter){
     FoodOptionTriggerHandler.updateHotelInformation(trigger.new);
    }
}
```



```
trigger FoodOptionTrigger on Food_Option__c (after insert, after update, after delete) {
    If(trigger.isInsert && trigger.isAfter){
     FoodOptionTriggerHandler.updateHotelInformation(trigger.new);
    }
}
```

Apex Schedule :

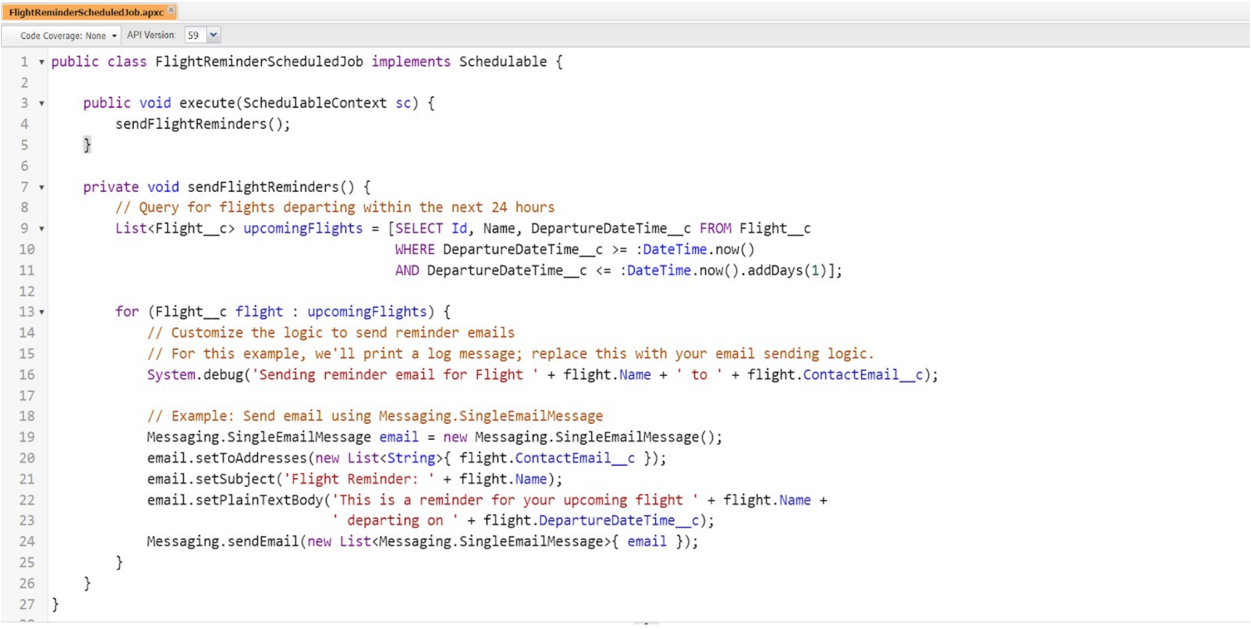
Create the Reminder mail for the customer who has booked the flight according to that booking set the Apex schedule so mail will be sent prior to 24hrs.

Note: Please create the required field for Scheduled Apex Code

```
}
```

Apex Schedule Class Solution :

```
public class FlightReminderScheduledJob implements Schedulable {

    public void execute(SchedulableContext sc) {
        sendFlightReminders();
    }

    private void sendFlightReminders() {
        // Query for flights departing within the next 24 hours
        List<Flight__c> upcomingFlights = [SELECT Id, Name, DepartureDateTime__c FROM Flight__c
                                           WHERE DepartureDateTime__c >= :DateTime.now()
                                           AND DepartureDateTime__c <= :DateTime.now().addDays(1)];

        for (Flight__c flight : upcomingFlights) {
            // Customize the logic to send reminder emails
            // For this example, we'll print a log message; replace this with your email sending logic.
            System.debug('Sending reminder email for Flight ' + flight.Name + ' to ' + flight.ContactEmail__c);

            // Example: Send email using Messaging.SingleEmailMessage
            Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
            email.setToAddresses(new List<String>{ flight.ContactEmail__c });
            email.setSubject('Flight Reminder: ' + flight.Name);
            email.setPlainTextBody('This is a reminder for your upcoming flight ' + flight.Name +
                                   ' departing on ' + flight.DepartureDateTime__c);
            Messaging.sendEmail(new List<Messaging.SingleEmailMessage>{ email });
        }
    }
}
```

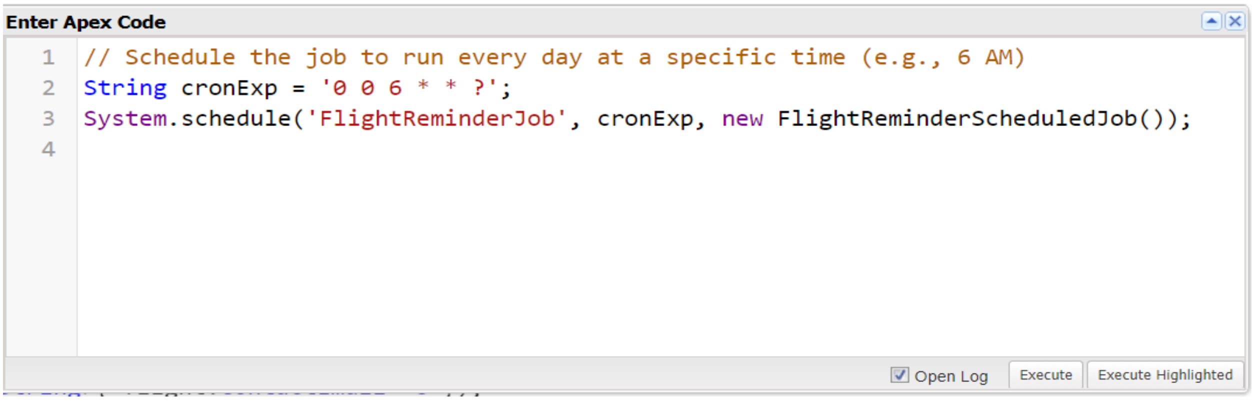


```
public class FlightReminderScheduledJob implements Schedulable {

    public void execute(SchedulableContext sc) {
        sendFlightReminders();
    }

    private void sendFlightReminders() {
        // Query for flights departing within the next 24 hours
        List<Flight__c> upcomingFlights = [SELECT Id, Name, DepartureDateTime__c FROM Flight__c
                                           WHERE DepartureDateTime__c >= :DateTime.now()
                                           AND DepartureDateTime__c <= :DateTime.now().addDays(1)];

        for (Flight__c flight : upcomingFlights) {
            // Customize the logic to send reminder emails
            // For this example, we'll print a log message; replace this with your email sending logic.
            System.debug('Sending reminder email for Flight ' + flight.Name + ' to ' + flight.ContactEmail__c);

            // Example: Send email using Messaging.SingleEmailMessage
            Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
            email.setToAddresses(new List<String>{ flight.ContactEmail__c });
            email.setSubject('Flight Reminder: ' + flight.Name);
            email.setPlainTextBody('This is a reminder for your upcoming flight ' + flight.Name +
                                   ' departing on ' + flight.DepartureDateTime__c);
            Messaging.sendEmail(new List<Messaging.SingleEmailMessage>{ email });
        }
    }
}
```

The FlightReminderScheduledJob class implements the Schedulable interface, and the execute method is where you put the logic to send reminder emails.

The sendFlightReminders method queries for flights departing within the next 24 hours. You can customize the query based on your specific requirements.

Create the Apex code in an anonymous Window to execute the Apex Code
// Schedule the job to run every day at a specific time (e.g., 6 AM)
String cronExp = '0 0 6 * * ?';
System.schedule('FlightReminderJob', cronExp, new FlightReminderScheduledJob());
System.schedule('FlightReminderJob', cronExp, new FlightReminderScheduledJob());



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

Conclusion: We have Created this Customization process for the proper flow of the business if TripAdvisor where they can easily access the Hotel requirement then food options and also the ease for the customers with the preferable discount with there Amount limits this process helps to save time from multiple manual processes.

```
public class FlightReminderScheduledJob implements Schedulable {
public void execute(SchedulableContext sc) {
sendFlightReminders();
}
private void sendFlightReminders() {
// Query for flights departing within the next 24 hours
List<Flight__c> upcomingFlights = [SELECT Id, Name,
DepartureDateTime__c FROM Flight__c
WHERE DepartureDateTime__c >= :DateTime.now()
AND DepartureDateTime__c <=
:DateTime.now().addDays(1)];
for (Flight__c flight : upcomingFlights) {
// Customize the logic to send reminder emails
// For this example, we'll print a log message; replace this with your email
sending logic.
System.debug('Sending reminder email for Flight ' + flight.Name + ' to ' +
flight.ContactEmail__c);
// Example: Send email using Messaging.SingleEmailMessage
Messaging.SingleEmailMessage email = new
Messaging.SingleEmailMessage();
email.setToAddresses(new List<String>{ flight.ContactEmail__c });
email.setSubject('Flight Reminder: ' + flight.Name);
email.setPlainTextBody('This is a reminder for your upcoming flight ' +
flight.Name +
' departing on ' + flight.DepartureDateTime__c);
Messaging.sendEmail(new List<Messaging.SingleEmailMessage>{ email
});
}
}
}
```

The FlightReminderScheduledJob class implements the Schedulable interface, and the execute method is where you put the logic to send reminder emails.

The sendFlightReminders method queries for flights departing within the next 24 hours. You can customize the query based on your specific requirements.

Create the Apex code in an anonymous Window to execute the Apex Code.

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

Implementation Project.

This section gives clarity on addressing various use cases or situations that Salesforce can handle during the implementation. Use Case 1: Automated feedback collection and analysis. Use Case 2: Real-time booking management and customer notifications. Use Case 3: Enhanced customer service response through ticket prioritization.

CONCLUSION :

Summary of Achievements: Successfully implemented an E-Management system for TripAdvisor that improves operational efficiency, automates customer interactions, and enhances user experience.