Project Title:

TRIPADVISOR E-MANAGEMENT

College Name: **MAILAM ENGINEERING COLLEGE**

Code: **4216**

Department of Computer Science and Business Systems

Team Leader:

Name: ABIRAMAN M	Reg No: 421622104001

Team Members:

Name: ARJUN M	Reg No: 421622104012
Name: BALAVENTHAN I	Reg No: 421622104018
Name: HARIHARAN S	Reg No: 421622104043

TripAdvisor E-Management

Hardware Required:

- System with advance configuration

Software Required:

- Salesforce Platform

System Required:

- Good Configuration
- The TripAdvisor E-Management app with the salesforce works with your all-in-one travel companion, empowering you to plan, book, and make the most of the trips. Discover millions of reviews and insights shared by fellow travelers, helping you make informed decisions for every aspect of your journey. Whether you're seeking the perfect hotel, top-rated restaurants, must-visit attractions, or the best travel deals, TripAdvisor has you covered.
- **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

- Enter label : Hotel

- Plural Name : Hotels
- Data Type : (text)
- Field Name : Hotel Name
- Click Allow Reports
- 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
• 1	• TotalFoodOptions	• Numb
• 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

Smart Internz

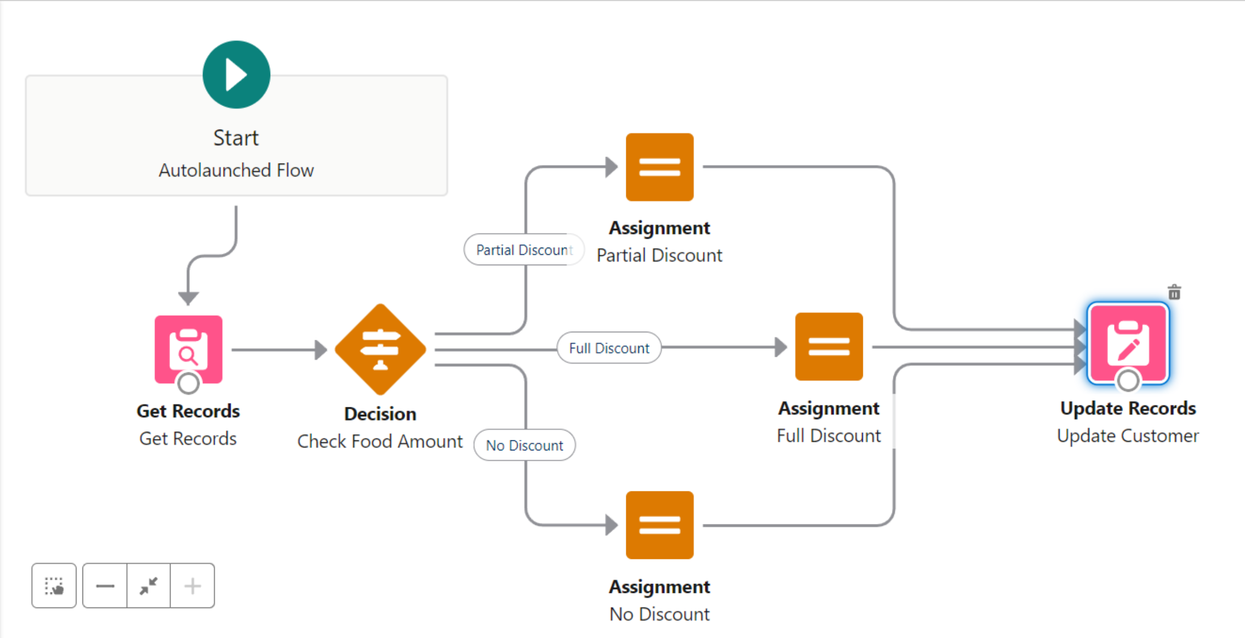
Create Flow

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

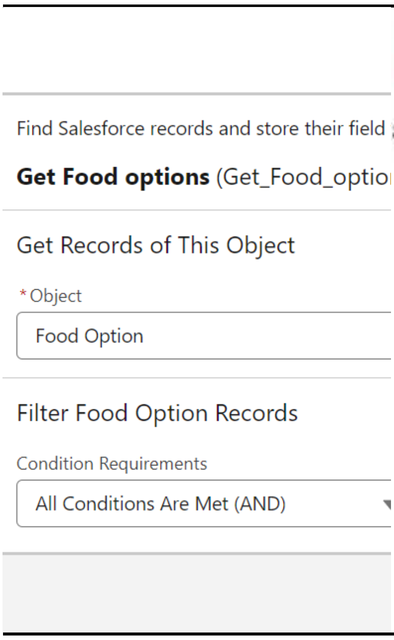
Flow Procedure

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



Find Salesforce records and store their field

Get Food options (Get_Food_optio

Get Records of This Object

*Object

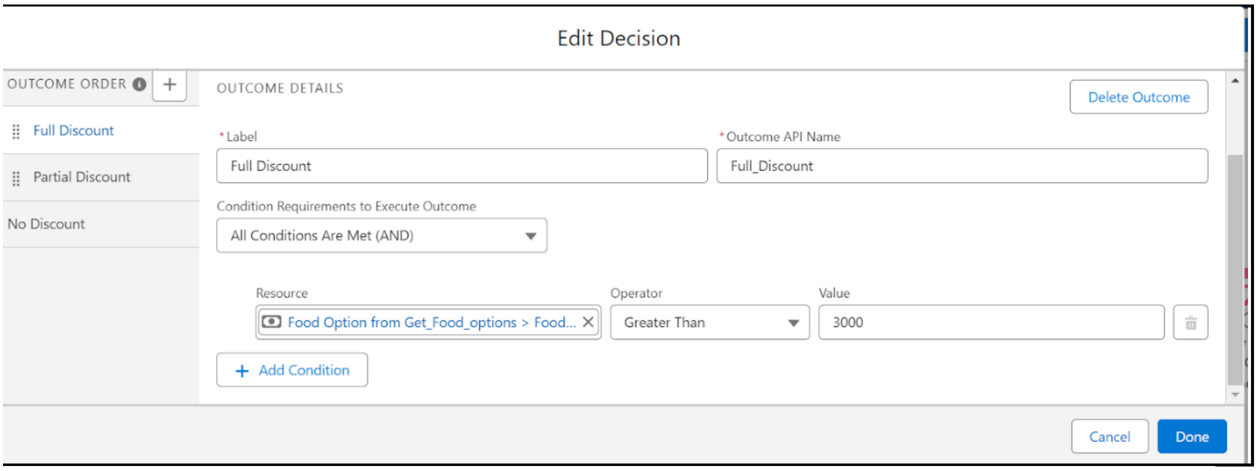
Food Option

Filter Food Option Records

Condition Requirements

All Conditions Are Met (AND)

2. Decision Element: Create 2 Outcomes



Edit Decision

Full Discount

Partial Discount

No Discount

OUTCOME DETAILS

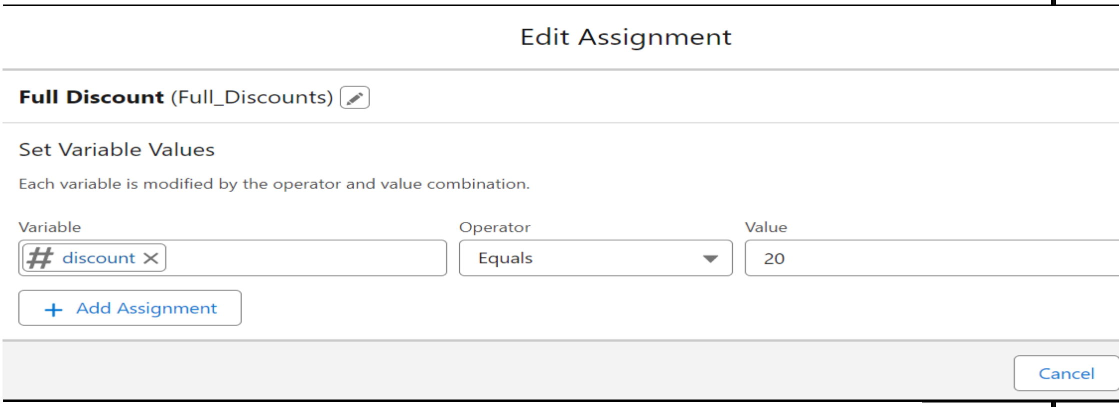
*Label: Full Discount

*Outcome API Name: Full_Discount

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

Resource	Operator	Value
Food Option from Get_Food_options > Food...	Greater Than	3000

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



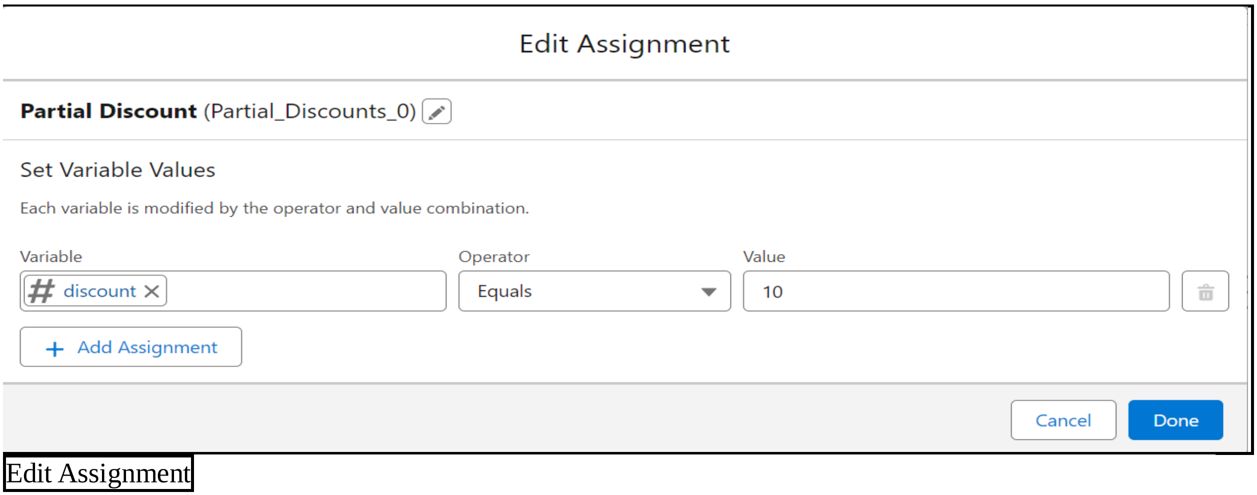
Edit Assignment

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

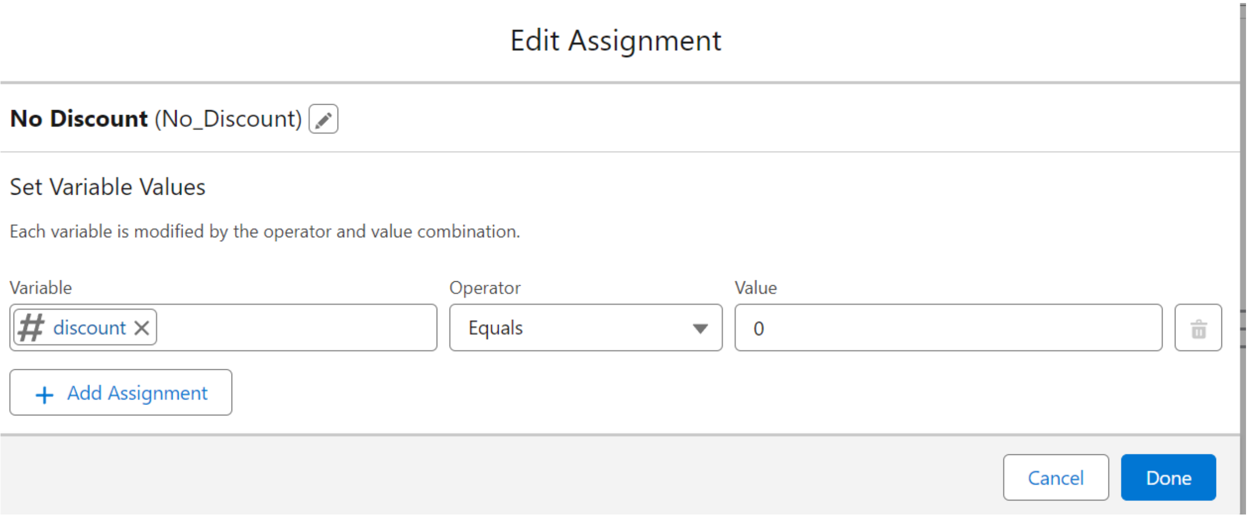
Partial Discount (Partial_Discounts_0)

Set Variable Values

Each variable is modified by the operator and value combination.

Variable	Operator	Value
discount	Equals	10

Edit Assignment



Edit Assignment

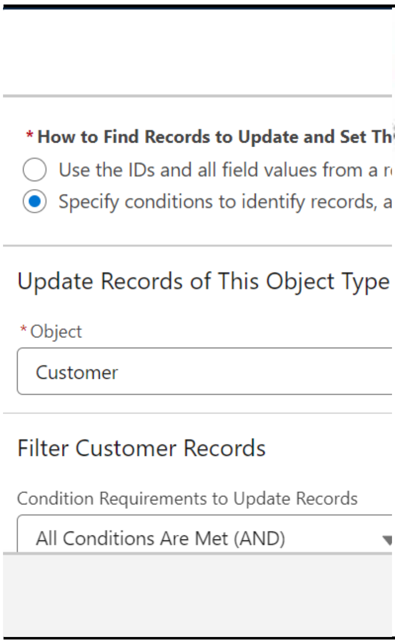
No Discount (No_Discount)

Set Variable Values

Each variable is modified by the operator and value combination.

Variable	Operator	Value
discount	Equals	0

Update Record Element



* How to Find Records to Update and Set Th

○ Use the IDs and all field values from a r

● Specify conditions to identify records, a

Update Records of This Object Type

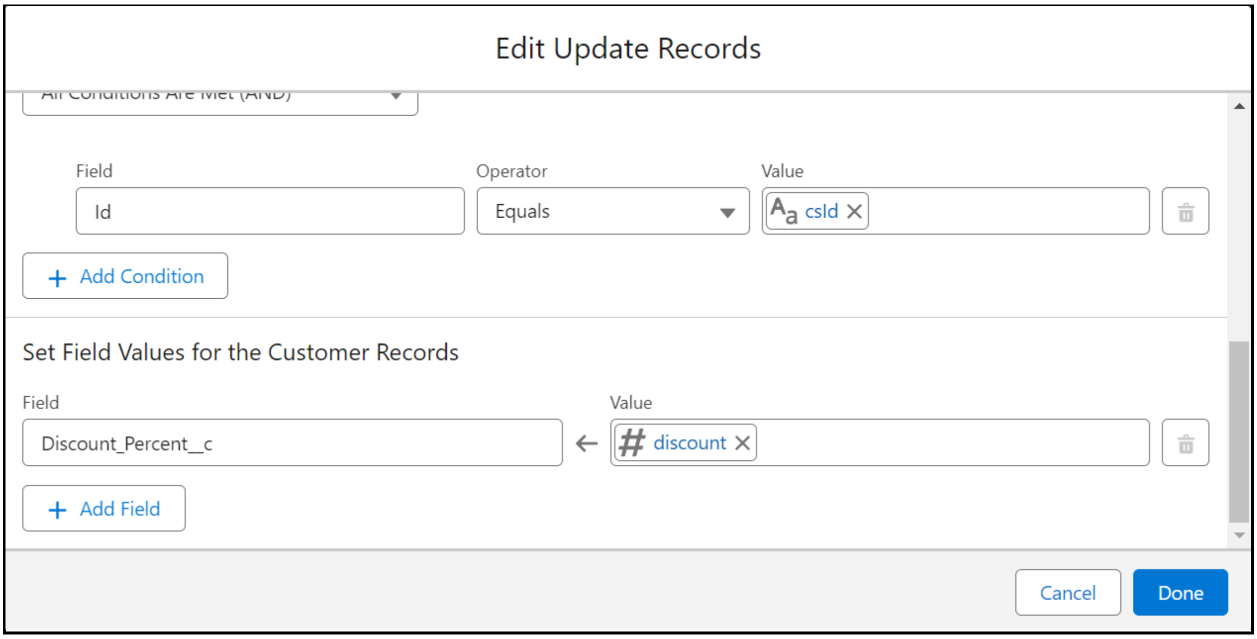
* Object

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

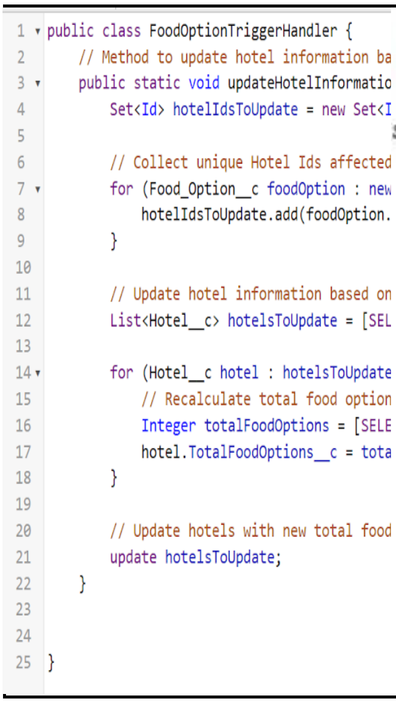
Apex Triggers

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

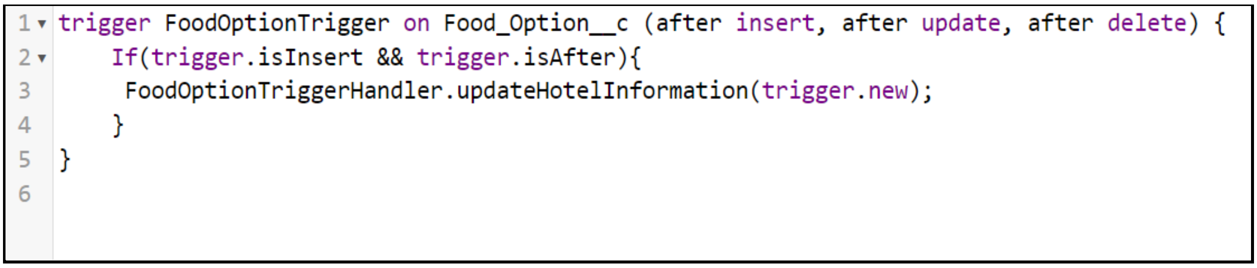
```
public class FoodOptionTriggerHandler {
    // Method to update hotel information based on food options
    public static void updateHotelInformation(List<Food_Option__c>newFoodOptions,
List<Food_Option__c>oldFoodOptions, TriggerOperation operation) {
        Set<Id>hotelIdsToUpdate = new Set<Id>();


        // Collect unique Hotel Ids affected by food options changes
        for (Food_Option__cfoodOption :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__chotel :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;
    }



}
```



```
public class FoodOptionTriggerHandler {
    // Method to update hotel information ba
    public static void updateHotelInformatio
        Set<Id> hotelIdsToUpdate = new Set<I

        // Collect unique Hotel Ids affected
        for (Food_Option__c foodOption : new
            hotelIdsToUpdate.add(foodOption.
        }

        // Update hotel information based on
        List<Hotel__c> hotelsToUpdate = [SEL

        for (Hotel__c hotel : hotelsToUpdate
            // Recalculate total food option
            Integer totalFoodOptions = [SELE
            hotel.TotalFoodOptions__c = tota
        }

        // Update hotels with new total food
        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.

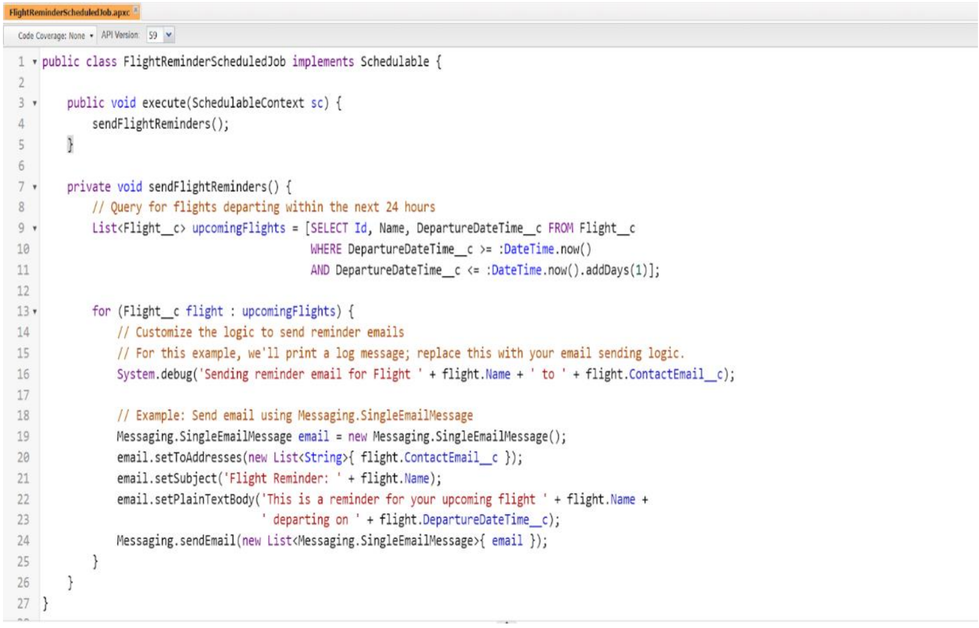
- Note: Please create the required field for Scheduled Apex Code

Apex Schedule Class Solution

- public class FlightReminderScheduledJob implements Schedulable {
- public void execute(SchedulableContextsc) {
- 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__cflight :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 });
- }
- }
- }

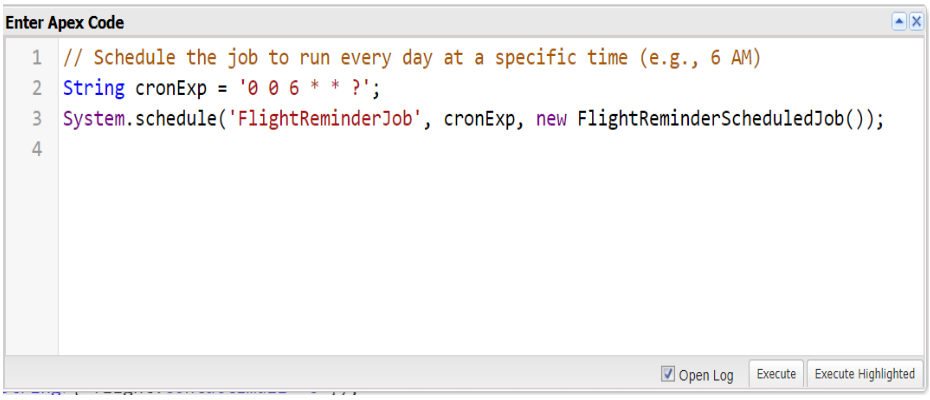


```
public class FlightReminderScheduledJob implements Schedulable {

    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());
-



```
// 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.