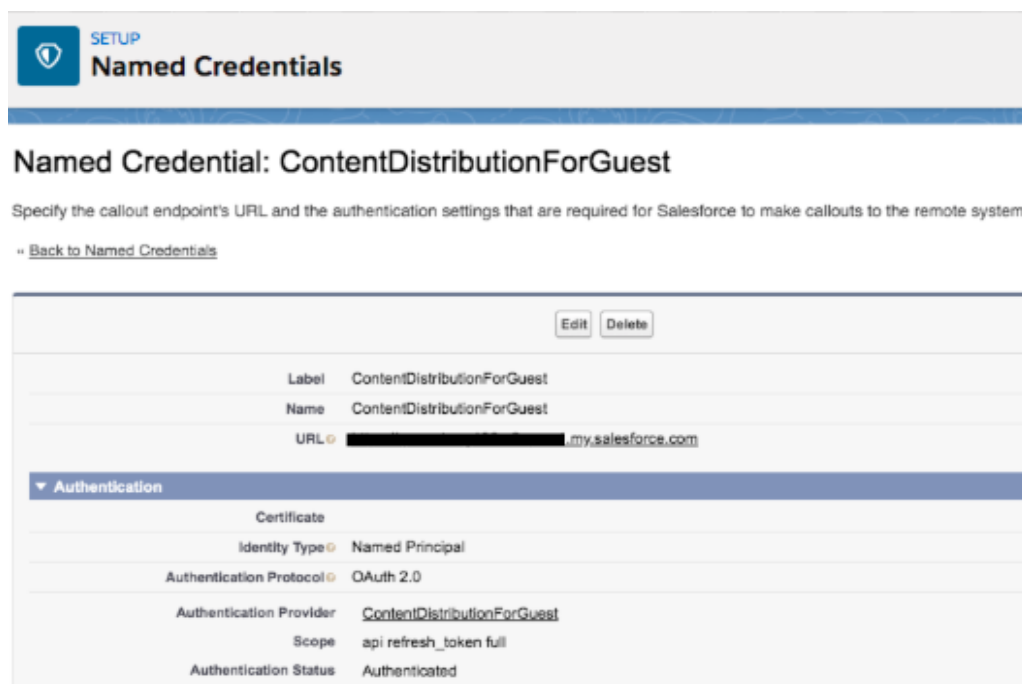# Corporate Travel & Expense Management System Project

Phase 6: Integration & External Access

🔸 **Goal**: Connect Salesforce with external systems for reimbursements, notifications, and future scalability.

1. Named Credentials

o Create **Finance_API_Credential** (**NamedCredential.DeveloperName**) for secure ERP/Finance API authentication.

2. External Services

o Import OpenAPI schema → generates actions like **ProcessReimbursement__c** for **Expense_Claim__c**.

3. Web Services (REST/SOAP)

- o Expose REST API:

  - Endpoint: /services/apexrest/TravelRequestAPI/

  - Allows external booking app to create **Travel_Request__c** (fields: **Purpose__c**, **Start_Date__c**, **End_Date__c**, **Estimated_Cost__c**).

- o Consume REST API:

  - Apex callout from **Expense_Claim__c** when **Status__c** = 'Approved'.

  - Sends JSON payload: {"claimId": **Expense_Claim__c.Id**, "amount": **Expense_Claim__c.Amount__c**}.

```apex
@RestResource(urlMapping='/TravelRequestAPI/*')
global with sharing class TravelRequestAPI {

    @HttpPost
    global static String createTravelRequest(String body) {
        RestRequest req = RestContext.request;
        RestResponse res = RestContext.response;

        // Parse incoming JSON
        Map<String, Object> json = (Map<String, Object>) JSON.deserializeUntyped(body);

        Travel_Request__c tr = new Travel_Request__c();
        tr.Purpose__c = (String) json.get('Purpose__c');
        tr.Start_Date__c = Date.valueOf((String) json.get('Start_Date__c'));
        tr.End_Date__c = Date.valueOf((String) json.get('End_Date__c'));
        tr.Estimated_Cost__c = Decimal.valueOf(json.get('Estimated_Cost__c') + '');

        insert tr;

        return tr.Id;
    }
}
```

☐ Open Log | Execute | Execute Highlighted

4. Callouts

- o Trigger: **ExpenseClaimTrigger** → After Update.

- o If **Status__c** == 'Approved' → call external Finance API using HttpRequest + NamedCredential**.**

```apex
public class ExpenseCallout {
    @future(callout=true)
    public static void sendExpense(String expenseId) {
        Expense_Claim__c claim = [SELECT Id, Amount__c FROM Expense_Claim__c WHERE Id = :expenseId LIMIT 1];

        HttpRequest req = new HttpRequest();
        req.setEndpoint('callout:ExpenseService/api/submitClaim');
        req.setMethod('POST');
        req.setHeader('Content-Type', 'application/json');

        Map<String, Object> payload = new Map<String, Object>{
            'claimId' => claim.Id,
            'amount' => claim.Amount__c
        };

        req.setBody(JSON.serialize(payload));

        Http http = new Http();
        HttpResponse res = http.send(req);

        // Optional: handle response
        System.debug('Response: ' + res.getBody());
    }
}
```

☐ Open Log   Execute   Execute Highlighted

5. Platform Events

- o Event: **TravelRequestSubmitted__e**.

- o Fields: **Travel_Request__c.Id**, **Purpose__c**, **Estimated_Cost__c**.

- o Published when a new Travel Request is submitted.

- o Subscribed by Manager Dashboard / Finance middleware.

```
import { LightningElement, wire } from 'lwc';
import { subscribe, MessageContext } from 'lightning/empApi';

export default class TravelRequestListener extends LightningElement {
    channelName = '/event/TravelRequestSubmitted__e';
    subscription = {};

    @wire(MessageContext)
    messageContext;

    connectedCallback() {
        this.handleSubscribe();
    }

    handleSubscribe() {
        subscribe(this.messageContext, this.channelName, (event) => {
            console.log('New Travel Request:', event.data.payload);
            // Update UI, show toast, etc.
        }).then(response => {
            this.subscription = response;
        });
    }
}
```

6. Change Data Capture

o   Enable Change data capture on:

- ✓ **Travel_Request__c** (tracks **Status__c** changes).

- ✓ **Expense_Claim__c** (tracks **Amount__c** + **Status__c** changes).

```
import { LightningElement, wire } from 'lwc';
import { subscribe, MessageContext } from 'lightning/empApi';

export default class TravelRequestStatusListener extends LightningElement {
    channelName = '/data/Travel_Request__cChangeEvent';
    subscription = {};

    @wire(MessageContext)
    messageContext;

    connectedCallback() {
        subscribe(this.messageContext, this.channelName, (message) => {
            const changed = message.data.payload.ChangeEventHeader.changedFields;
            if (changed.includes('Status__c')) {
                console.log('Status__c changed:', message.data.payload.Status__c);
                // Handle update (e.g., notify manager, refresh dashboard, etc.)
            }
        }).then(response => {
            this.subscription = response;
        });
    }
}
```

7. Salesforce Connect

o External Object: **Finance_Reimbursement__x**.

o Maps reimbursement data from ERP → visible in related list on **Expense_Claim__c**.

8. API Limits

o Dev Org = 15,000 API calls/day (**Organization.ApiRequests**).

o Batch callouts → group multiple **Expense_Claim__c** in one request.

```apex
public class ExpenseCalloutService {
    @future(callout=true)
    public static void sendBatchClaims(List<Id> claimIds) {
        List<Expense_Claim__c> claims = [
            SELECT Id, Amount__c
            FROM Expense_Claim__c
            WHERE Id IN :claimIds
        ];

        List<Map<String, Object>> payloadList = new List<Map<String, Object>>();

        for (Expense_Claim__c ec : claims) {
            payloadList.add(new Map<String, Object>{
                'claimId' => ec.Id,
                'amount' => ec.Amount__c
            });
        }

        Map<String, Object> payloadWrapper = new Map<String, Object>{
            'claims' => payloadList
        };

        HttpRequest req = new HttpRequest();
        req.setEndpoint('callout:ExpenseService/api/submitClaimsBatch');
        req.setMethod('POST');
        req.setHeader('Content-Type', 'application/json');
        req.setBody(JSON.serialize(payloadWrapper));
```

```
            Http http = new Http();
            HttpResponse res = http.send(req);
            System.debug('Response: ' + res.getBody());
        }
}
```

9. OAuth & Authentication

o External Finance API → connect via OAuth 2.0 using **AuthProvider**.

o Managers approve in Salesforce → approval flows propagate to ERP**.**

```
public class ERPApprovalSync {
    @future(callout=true)
    public static void sendApproval(Id claimId) {
        Expense_Claim__c ec = [
            SELECT Id, Amount__c, Status__c
            FROM Expense_Claim__c
            WHERE Id = :claimId
        ];

        if (ec.Status__c != 'Approved') return;

        HttpRequest req = new HttpRequest();
        req.setEndpoint('callout:ERP_Finance_API/claims/approve');
        req.setMethod('POST');
        req.setHeader('Content-Type', 'application/json');

        Map<String, Object> payload = new Map<String, Object>{
            'claimId' => ec.Id,
            'amount' => ec.Amount__c,
            'approvedBy' => UserInfo.getUserId(),
            'approvalDate' => Date.today()
        };

        req.setBody(JSON.serialize(payload));

        Http http = new Http();
        HttpResponse res = http.send(req);
```

```
    HttpResponse res = http.send(req);

    System.debug('ERP Approval Response: ' + res.getBody());
    }
}
```

10. Remote Site Settings

o   Add Finance API base URL:

▪   RemoteSiteSetting.DeveloperName = **FinanceAPI**

▪   Also add **RemoteSiteSetting.Url** to base URL.

```
HttpRequest req = new HttpRequest();
req.setEndpoint('https://ogfarm-63a245672-corporate_travel_and_expense_system.com/api/test');
req.setMethod('GET');

Http http = new Http();
HttpResponse res = http.send(req);

System.debug('Response: ' + res.getBody());
```