Phase 7: Integration & External Access

Introduction

In the lifecycle of any Salesforce project, integration plays a critical role in ensuring that Salesforce does not operate in isolation but as part of a connected enterprise ecosystem. For ApexHub, Phase 7 is dedicated to enabling **data exchange, process synchronization, and secure external access**. This ensures that Salesforce can seamlessly interact with other business applications such as ERP systems, customer service portals, payment gateways, and third-party APIs.

By exposing Salesforce data and services through APIs and consuming external services using Apex callouts, ApexHub becomes a **central hub for both business logic and real-time collaboration**. Without proper integration, the system would risk becoming siloed, leading to inefficiencies, duplication of work, and inconsistent customer experiences.

This phase also establishes **security, governance, and error-handling practices** for integrations, ensuring that external access does not compromise the integrity or performance of Salesforce.

Objectives

The objectives of Phase 7 are as follows:

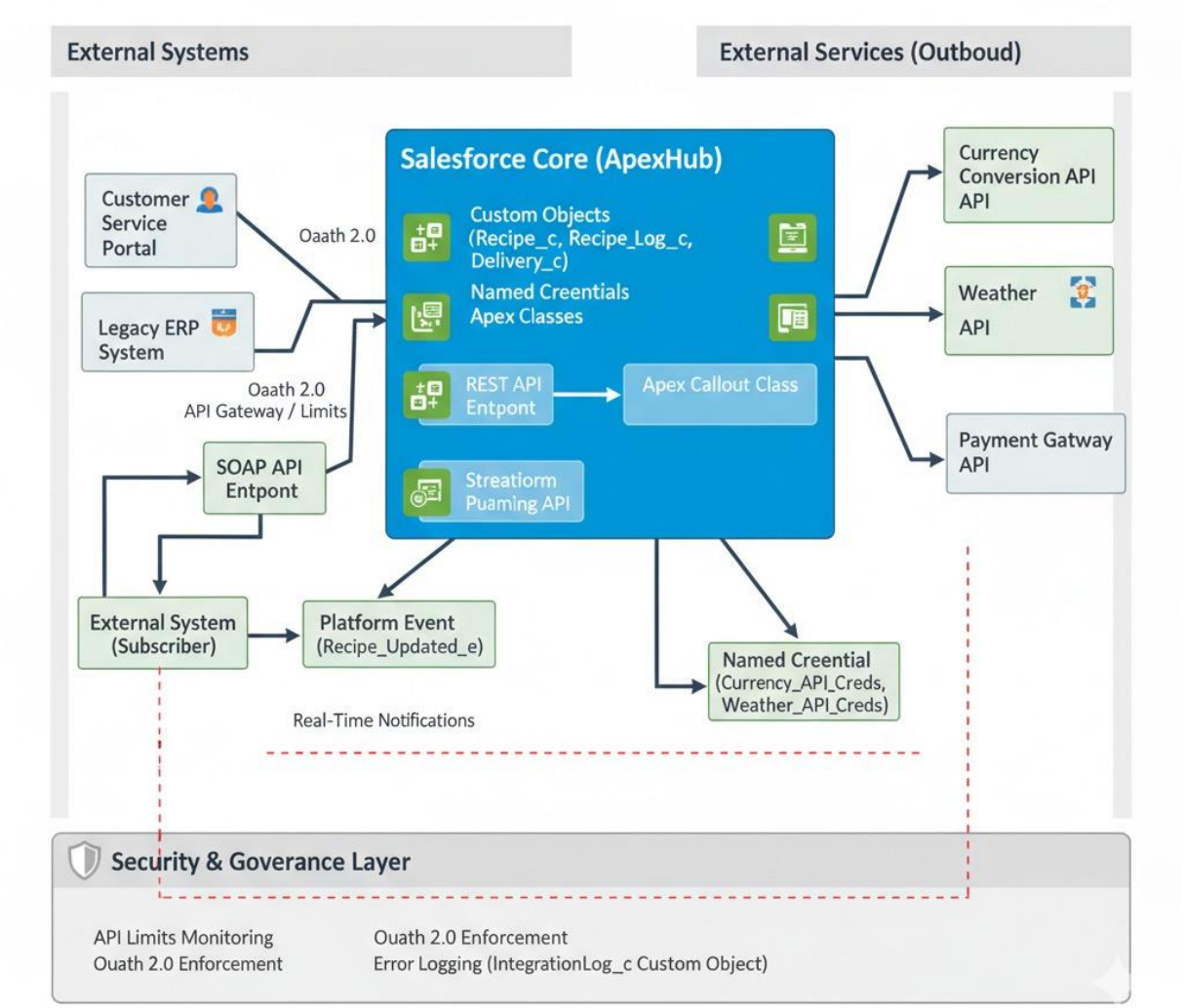
1. Configure Salesforce **REST and SOAP APIs** to allow secure external access to ApexHub data.
2. Implement **Apex callouts** to consume external services (e.g., currency conversion, payment validation, geolocation).
3. Set up **Named Credentials** to securely manage authentication details for third-party services.
4. Implement **Platform Events** and **Streaming APIs** for real-time, event-driven communication with external applications.
5. Define **error handling and monitoring practices** to track failed integrations and ensure system reliability.
6. Establish **security protocols** using OAuth 2.0, API limits, and governance best practices.
7. Validate integrations using test tools (e.g., Postman) and sandbox environments before production deployment.

Activities

1. REST & SOAP API Enablement

- **REST API:** Enabled to allow external applications to access and manipulate Salesforce data.
    - Example: /services/data/v60.0/sobjects/Recipe__c/ retrieves recipe data in JSON format.
- **SOAP API:** Configured for compatibility with legacy systems requiring XML-based communication.

- Testing performed using **Postman** and external client tools to confirm endpoints were functional and secure.



2. Apex Callouts to External Services

Apex classes were developed to perform **HTTP callouts** to third-party APIs. Examples include:

- **Currency Conversion API:** Used to calculate dynamic invoice amounts in multiple currencies.
- **Weather API:** Logs local weather conditions against a Delivery__c object for analytics.
- **Payment Gateway API:** (future scope) To validate transactions directly from Salesforce.

All callouts were secured using **Named Credentials**, ensuring no sensitive credentials were hard-coded.

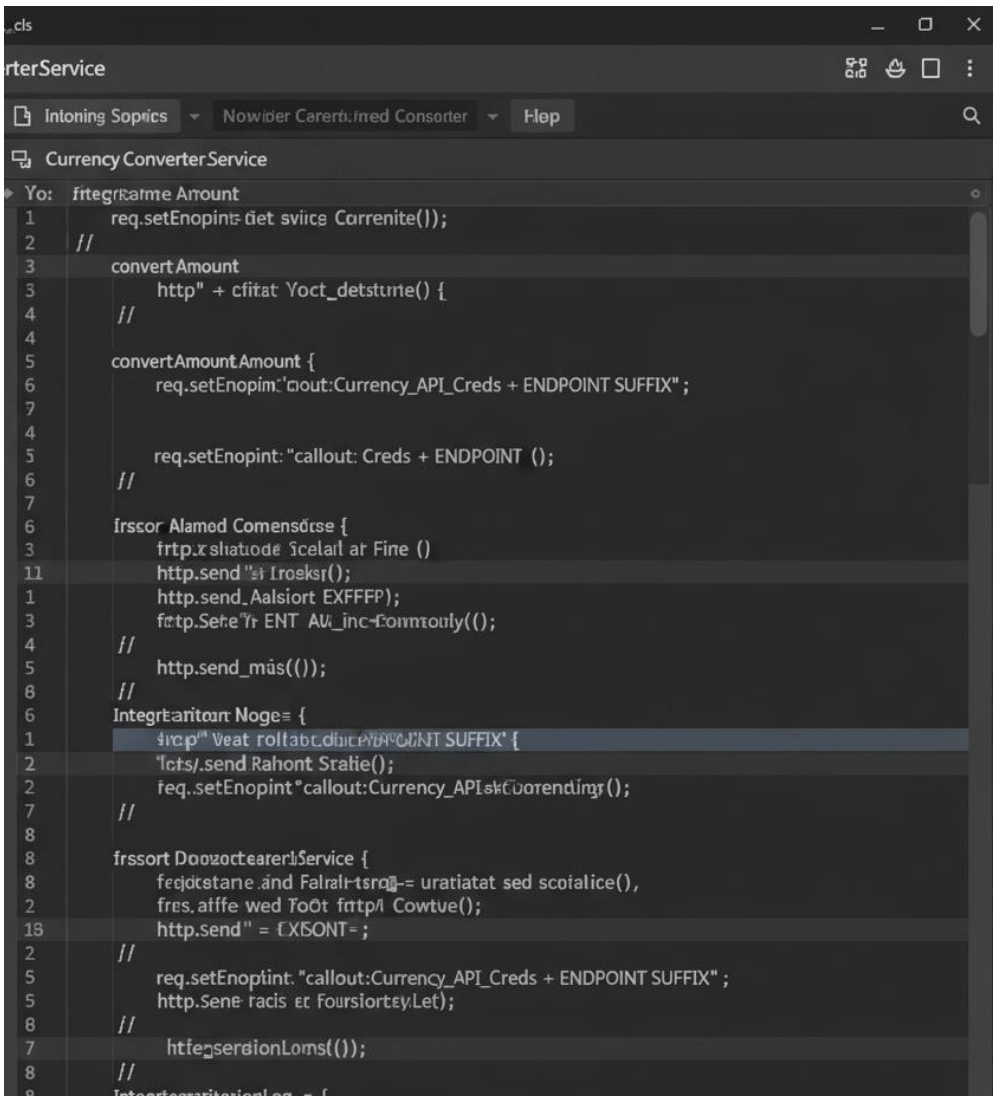
3. Named Credentials Setup

- Named Credentials were configured to store external service endpoints and authentication keys.
- Example: Currency_API_Creds was created to handle all currency conversion API requests.
- This simplified authentication and centralized external access management.

4. Real-Time Integrations with Platform Events & Streaming API

- **Platform Event (Recipe_Updated__e):** Fired whenever a recipe record was updated, notifying external systems instantly.
- **Streaming API (PushTopic):** Configured to notify client systems when new RecipeLog__c records were created.
- Example Use Case: A connected ERP system updates its project planning module whenever a recipe execution log is completed.

This **event-driven model** ensures that ApexHub data is always synchronized with external systems in near real time.



5. Security & Governance Practices

- **OAuth 2.0:** Implemented for secure external authentication.
- **API Limits:** Configured monitoring dashboards to track daily API calls and prevent overuse.
- **Error Logging:** A custom object IntegrationLog__c was created to record API failures, error codes, and timestamps for troubleshooting.

6. Sandbox & Testing

- All integrations were first validated in **Scratch Orgs and Sandboxes**.
- Postman collections were created for REST API testing.
- Simulated failure scenarios to confirm retry and error-handling mechanisms worked as expected.

Deliverables

At the end of Phase 7, the following deliverables were achieved:

1. REST and SOAP APIs enabled and validated for ApexHub objects (Recipe__c, RecipeLog__c).
2. Apex callout classes implemented for external service consumption (currency and weather APIs).
3. Named Credentials configured for secure API authentication.
4. Platform Event (Recipe_Updated__e) created and tested with external subscribers.
5. Streaming API configured to broadcast changes in RecipeLog__c.
6. Security policies (OAuth 2.0, API limits, error logging) documented and enforced.
7. Integration testing completed in sandbox environments with Postman validation.

Expected Outcomes

The completion of Phase 7 ensures that:

- **ApexHub is no longer a siloed system** but part of a broader enterprise ecosystem.
- Salesforce data is securely exposed to partner systems while respecting access limits.
- External services enrich Salesforce functionality (e.g., real-time currency rates).
- Event-driven architecture reduces latency and ensures data synchronization across platforms.
- Integration practices follow Salesforce's **security and governance standards**, minimizing risks.
- The system is future-ready for middleware adoption (e.g., MuleSoft, Boomi) if enterprise integration needs grow.