General Integration Questions

1. What are different ways to integrate Salesforce with external systems?

Answer:

Salesforce integration can be done using:

- **API-based Integration:** Using REST, SOAP, or GraphQL APIs.
- **Middleware-based Integration:** Using tools like Mulesoft, Dell Boomi, or Informatica.
- **Event-based Integration:** Using Platform Events or Change Data Capture.
- **Data-based Integration:** Using Salesforce Connect, External Objects, or ETL tools.

Explanation:

The integration method depends on the use case—real-time, batch processing, or event-driven.

2. What is the difference between real-time and batch integrations?

Answer:

- **Real-time Integration:** Immediate response (e.g., REST API, SOAP API, Platform Events).
- **Batch Integration:** Processes data in bulk at scheduled intervals (e.g., Bulk API, ETL tools).

Explanation:

Real-time is useful for instant data updates, while batch is efficient for large-volume data processing.

3. Explain inbound and outbound integrations in Salesforce.

Answer:

- **Inbound Integration:** External system sends data to Salesforce using APIs like REST, SOAP, or Bulk API.
- **Outbound Integration:** Salesforce sends data to an external system using callouts, outbound messages, or event-driven mechanisms.

Explanation:

Inbound is used when Salesforce consumes external services, while outbound is used when Salesforce sends data to another system.

APIs & Web Services

4. What are the different types of APIs available in Salesforce?

Answer:

- **REST API** – Lightweight, used for web and mobile apps.
- **SOAP API** – Uses XML, suitable for enterprise applications.
- **Bulk API** – Processes large datasets asynchronously.
- **Streaming API** – Used for real-time event notifications.
- **Metadata API** – Used for deploying metadata.
- **Tooling API** – Used for development and debugging tools.

Explanation:

Each API serves different integration needs—REST for flexibility, SOAP for security, and Bulk for large data.

5. When would you use REST API vs. SOAP API in Salesforce?

Answer:

- **Use REST API:** When you need lightweight, simple integrations for web & mobile apps.
- **Use SOAP API:** When security, reliability, and structured transactions are required.

Explanation:

REST is stateless and easy to use, while SOAP provides robust security and structured transactions.

6. What is the difference between REST API and Bulk API?

Answer:

- **REST API:** Handles small real-time requests (e.g., retrieving a single record).
- **Bulk API:** Processes large amounts of data asynchronously.

Explanation:

Use REST for frequent small transactions and Bulk API for high-volume data migration.

7. How do you handle API rate limits in Salesforce?

Answer:

- Implement caching mechanisms.
- Use the Composite API to minimize API calls.
- Utilize Bulk API for data processing.
- Monitor API usage with `Limits` class in Apex.

Explanation:

Salesforce imposes API limits per org; optimizing API calls ensures smooth operations.

8. How do you authenticate REST API calls to Salesforce?

Answer:

Use **OAuth 2.0**, which supports:

- **Username-Password Flow**
- **JWT Bearer Token Flow**
- **Client Credentials Flow**

Explanation:

OAuth 2.0 ensures secure authentication and avoids exposing credentials directly.

9. What is the purpose of Named Credentials in Salesforce?

Answer:

Named Credentials store API endpoints and authentication details securely, removing the need to manage them in code.

Explanation:

They simplify authentication for external integrations by managing API endpoints and security in Salesforce.

Middleware & Integration Tools

10. What are some common middleware tools used for Salesforce integrations?

Answer:

- **Mulesoft** – Enterprise-level integration tool.
- **Dell Boomi** – Cloud-based integration.
- **Informatica** – Data integration platform.
- **Jitterbit** – API management tool.

Explanation:

Middleware helps integrate Salesforce with external systems without extensive coding.

11. How do you integrate Salesforce with Mulesoft?

Answer:

Use the **Salesforce Connector** in Mulesoft to call Salesforce APIs and process data between systems.

Explanation:

Mulesoft provides prebuilt connectors for easy Salesforce integration.

12. What is Salesforce Connect, and how does it work?

Answer:

Salesforce Connect allows accessing external data in Salesforce without storing it. It uses:

- **OData protocol**
- **External Objects**

Explanation:

Salesforce Connect provides a real-time view of external system data without data duplication.

Apex Integration (Callouts & Web Services)

13. How do you make an HTTP callout in Apex?

Answer:

Use `Http` and `HttpRequest` classes:

```
Http http = new Http();
HttpRequest request = new HttpRequest();
```

```
request.setEndpoint('https://api.example.com');
request.setMethod('GET');
HttpResponse response = http.send(request);
```

Explanation:

Apex callouts allow Salesforce to communicate with external services using REST or SOAP.

14. What is the purpose of the `@future(callout=true)` annotation?

Answer:

It enables asynchronous callouts to avoid Apex transaction limits.

Explanation:

Salesforce restricts callouts in synchronous transactions; `@future(callout=true)` helps avoid these limits.

Integration Patterns & Best Practices

15. What are the five Salesforce integration patterns?

Answer:

1. **Remote Process Invocation - Request & Reply** (Real-time API call).
2. **Remote Process Invocation - Fire & Forget** (Asynchronous API call).
3. **Batch Data Synchronization** (Scheduled data transfer).
4. **Remote Call-In** (External system calls Salesforce API).
5. **UI Update Based on Data Changes** (Streaming API & Platform Events).

Explanation:

Each pattern serves a specific use case—real-time, batch, or event-driven.

16. What is the difference between an ESB (Enterprise Service Bus) and Point-to-Point integration?

Answer:

- **ESB (e.g., Mulesoft)**: Centralized integration hub.
- **Point-to-Point**: Direct connection between two systems.

Explanation:

ESB is scalable and reusable, while point-to-point is simple but hard to maintain at scale.

Security & Authentication

17. How does OAuth 2.0 work in Salesforce?

Answer:

OAuth 2.0 allows secure access via **Access Tokens** without sharing credentials.

Explanation:

OAuth is used for API authentication and external app integrations.

18. What is JWT (JSON Web Token) authentication in Salesforce?

Answer:

JWT provides stateless, secure authentication using signed tokens.

Explanation:

JWT is useful for server-to-server authentication in Salesforce integrations.

19. What is the difference between a Connected App and an API user?

Answer:

- **Connected App**: Defines how external apps authenticate with Salesforce.
- **API User**: A dedicated user with API permissions.

Explanation:

A Connected App manages authentication, while an API User provides controlled access.

20. How do you secure Salesforce APIs from unauthorized access?

Answer:

- Use **OAuth 2.0** for authentication.
- Set **IP Whitelisting & Profile-based access**.
- Use **Named Credentials** to store sensitive information securely.

Explanation:

Following security best practices ensures API access control and data protection.

Data Integration & ETL

21. What are the different ways to import/export data in Salesforce?

Answer:

- **Data Loader** – Manual import/export via CSV.
- **Data Import Wizard** – UI-based tool for simple imports.
- **Bulk API** – Efficient for large data sets.
- **ETL Tools (Mulesoft, Informatica, Boomi, etc.)** – For complex integrations.

Explanation:

Data Loader is best for one-time loads, while ETL tools handle complex data transformations and migrations.

22. What is the role of Data Loader and its limitations?

Answer:

Role: Data Loader is a Salesforce-provided tool for bulk importing, updating, and exporting data.

Limitations:

- Cannot handle large datasets efficiently (use Bulk API instead).
- Requires manual operation unless automated via CLI.
- Does not support complex data transformations.

Explanation:

Data Loader is ideal for admin-driven imports, but it lacks real-time automation.

23. How do you integrate Salesforce with an external database?

Answer:

- **Salesforce Connect (OData/External Objects)** – Access external data in real-time.
- **Custom API Callouts (Apex REST/SOAP callouts)** – Fetch data from external databases.
- **Middleware (Mulesoft, Boomi, Informatica, etc.)** – Synchronize data between systems.
- **Direct Database Connection (JDBC/ODBC via Heroku Connect)** – Sync Salesforce with PostgreSQL.

Explanation:

The choice depends on whether real-time or batch synchronization is required.

24. What are the best practices for bulk data processing in integrations?

Answer:

- Use **Bulk API** instead of REST API for large datasets.
- Implement **batch processing** to avoid hitting limits.
- Use **Asynchronous Apex (Batch Apex, Queueable Apex)** for scalable processing.
- Monitor API limits and optimize queries.

Explanation:

Optimizing bulk operations ensures performance efficiency and avoids API limits.

25. How do you use the Bulk API for data migration?

Answer:

1. **Create a Bulk API Job** – Define insert, update, upsert, or delete.
2. **Submit Batches** – Upload large datasets in chunks.
3. **Monitor Job Status** – Check processing status.
4. **Retrieve Results** – Handle success and error responses.

Explanation:

Bulk API is optimized for large-scale data migration with minimal API consumption.

Integration Patterns & Best Practices

26. What are the five Salesforce integration patterns?

Answer:

1. **Remote Process Invocation - Request & Reply** – Real-time API request (e.g., REST API).
2. **Remote Process Invocation - Fire & Forget** – Asynchronous request (e.g., Future Methods).

3. **Batch Data Synchronization** – Periodic data transfer (e.g., ETL tools).
4. **Remote Call-In** – External system calls Salesforce API (e.g., Salesforce REST API).
5. **UI Update Based on Data Changes** – Streaming API for real-time UI updates.

Explanation:

Each pattern is suited for different business scenarios (real-time, batch, event-driven).

27. When would you use the Remote Process Invocation - Request and Reply pattern?

Answer:

- When **real-time** data exchange is needed.
- When Salesforce needs an **immediate response** from an external system (e.g., validating payment before order creation).

Explanation:

This pattern ensures that Salesforce receives instant responses but increases dependency on external system availability.

28. What is the difference between an ESB (Enterprise Service Bus) and Point-to-Point integration?

Answer:

- **ESB (Mulesoft, Boomi, etc.)** – Centralized hub managing multiple integrations.
- **Point-to-Point** – Direct connection between two systems.

Explanation:

ESB is scalable but complex, while point-to-point is simple but difficult to maintain in large environments.

29. How do you ensure data consistency across multiple systems in an integration?

Answer:

- Use **transactional integration** (e.g., rollback on failure).
- Implement **idempotency** (avoiding duplicate processing).
- Utilize **event-driven architecture** (Platform Events, Change Data Capture).
- Regularly sync data using ETL tools.

Explanation:

Ensuring consistency prevents data mismatches and improves system reliability.

30. What are some common integration challenges and how do you solve them?

Answer:

1. **API Rate Limits** – Use Bulk API, Composite API, or cache data.
2. **Authentication Issues** – Implement OAuth 2.0, JWT, or Named Credentials.
3. **Data Latency** – Use Platform Events for real-time updates.
4. **Error Handling** – Implement retry mechanisms and error logs.
5. **Security Risks** – Follow best practices like encryption and OAuth scopes.

Explanation:

Overcoming these challenges ensures seamless and secure integrations.

Security & Authentication

31. How does OAuth 2.0 work in Salesforce?

Answer:

OAuth 2.0 provides secure authentication using **Access Tokens** via flows like:

- **Authorization Code Flow** (Web apps)

- **JWT Bearer Token Flow** (Server-to-server)
- **Client Credentials Flow** (Machine-to-machine)

Explanation:

OAuth 2.0 enhances security by eliminating the need to store credentials.

32. What is JWT (JSON Web Token) authentication in Salesforce?

Answer:

JWT authentication allows **stateless, secure** API authentication by sending **signed tokens** instead of credentials.

Explanation:

JWT is useful for server-to-server integrations where no user interaction is required.

33. How do you configure Single Sign-On (SSO) with Salesforce?

Answer:

1. **Enable SSO** in Salesforce.
2. **Set up Identity Provider (IdP)** – Okta, Azure AD, or PingIdentity.
3. **Configure SAML or OAuth** for authentication.
4. **Test SSO Login** and troubleshoot.

Explanation:

SSO allows users to log in to Salesforce using corporate credentials without re-entering passwords.

34. What is the difference between a Connected App and an API user?

Answer:

- **Connected App** – Defines OAuth flows and permissions for external applications.
- **API User** – A dedicated user with API access, often with restricted permissions.

Explanation:

A Connected App manages authentication, while an API User provides dedicated access with specific security controls.

35. How do you secure Salesforce APIs from unauthorized access?

Answer:

- **Use OAuth 2.0** for authentication.
- **Enable IP restrictions** to prevent unauthorized logins.
- **Limit API Access** with Profiles & Permission Sets.
- **Use Named Credentials** to manage external API endpoints securely.

Explanation:

Implementing strong security controls ensures API integrity and prevents unauthorized access.

Real-World Salesforce integration scenarios

Scenario 1: Payment Gateway Integration (REST API)

Business Case:

A company wants to integrate **Salesforce with PayPal/Stripe** so that when an opportunity is closed-won, the payment link is automatically generated and sent to the customer.

Solution:

- Use **Apex HTTP Callout** to PayPal/Stripe REST API to create a payment request.

- Store the **Payment Link** in a custom field on the Opportunity record.
- Send an **automated email** to the customer with the payment link.
- Once the payment is made, use a **Webhook (Inbound API)** to update the payment status in Salesforce.

Implementation Example (Outbound Callout to Stripe API):

```
HttpRequest req = new HttpRequest();
req.setEndpoint('https://api.stripe.com/v1/payment_intents');
req.setMethod('POST');
req.setHeader('Authorization', 'Bearer sk_test_xxxxxxx');
req.setBody('amount=5000&currency=usd&payment_method_types[]=card');

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

Key Takeaways:

- **Real-time payment status updates.**
- **Webhook listener for inbound updates.**
- **Secure API authentication using OAuth 2.0.**

Scenario 2: Salesforce - ERP Integration (SAP/Microsoft Dynamics)

Business Case:

A company wants to sync **Salesforce Orders** with their **ERP system (SAP/Microsoft Dynamics)** for inventory and invoice management.

Solution:

- Use **Mulesoft or Boomi** as middleware for data transformation.
- When an **order is created in Salesforce**, trigger an **Outbound Message or Platform Event**.
- The ERP system receives order details, processes it, and updates inventory.
- ERP then sends the **invoice details** back to Salesforce using REST API.

Key Takeaways:

- **Batch integration (using Bulk API) for high data volume.**

- **Event-driven (Platform Events) for real-time updates.**
- **Error handling via middleware (retry logic, logging).**

Scenario 3: Real-Time Lead Enrichment Using External API

Business Case:

A sales team wants to **enrich lead data** (email, phone, company size) in real-time by calling an external **Data Enrichment API (e.g., Clearbit, ZoomInfo, or LinkedIn APIs)**.

Solution:

- When a **new Lead** is created in Salesforce, make an **Apex Callout** to fetch additional details.
- Update the Lead record with the **company size, industry, revenue, LinkedIn profile, etc.**
- Use **Named Credentials** for API authentication.

Example Apex Callout to External API:

```
HttpRequest req = new HttpRequest();
req.setEndpoint('https://api.clearbit.com/v2/companies/find?domain=exa
mple.com');
req.setMethod('GET');
req.setHeader('Authorization', 'Bearer your_api_key');

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

Key Takeaways:

- **Real-time lead enrichment improves sales efficiency.**
- **Named Credentials for secure API authentication.**
- **Governor limits managed using Future/Queueable Apex.**

Scenario 4: Salesforce & External Database Sync (SQL Server/PostgreSQL)

Business Case:

A company needs to keep **customer records in sync** between Salesforce and an **external SQL Server/PostgreSQL database** for reporting.

Solution:

- Use **Salesforce Connect & OData** to access external database tables **without storing data in Salesforce**.
- If data must be stored in Salesforce, use **MuleSoft or ETL tools (Informatica, Talend)** to sync data periodically.
- Implement **Change Data Capture (CDC)** in Salesforce to detect and push data updates to the external database.

Key Takeaways:

- **Salesforce Connect avoids data duplication.**
- **ETL tools handle batch processing for large datasets.**
- **Change Data Capture ensures real-time sync.**

Scenario 5: Customer Support System Integration (Zendesk/ServiceNow)

Business Case:

A company uses **Zendesk for ticketing** and **Salesforce for customer management**. They want to:

1. **Create a new case in Salesforce** when a ticket is created in Zendesk.
2. **Update the ticket in Zendesk** when the case is resolved in Salesforce.

Solution:

- Use **Zapier or MuleSoft** for **low-code integration**.
- Configure **Outbound Messages** to send case updates to Zendesk.
- Use **REST API** to fetch ticket details from Zendesk when needed.

Key Takeaways:

- **API integration ensures real-time updates.**
- **Middleware simplifies automation.**
- **Webhook-based updates avoid polling.**

Scenario 6: Social Media Integration (Twitter/Facebook)

Business Case:

A company wants to track customer complaints from **Twitter** in Salesforce and create cases automatically.

Solution:

- Use **Twitter API + Salesforce Apex Callouts** to fetch mentions of their brand.
- Use **Einstein Sentiment Analysis** to detect negative tweets.
- If a complaint is detected, **automatically create a case** in Salesforce.

Key Takeaways:

- **AI-driven customer service (Sentiment Analysis).**
- **API-based social media monitoring.**
- **Automated case creation & assignment.**

Scenario 7: IoT Integration with Salesforce (Smart Devices & Sensors)

Business Case:

A **smart home device company** wants to track device status in Salesforce and notify customers when maintenance is required.

Solution:

- Devices send **sensor data (temperature, battery, errors)** to **AWS IoT/MQTT**.
- AWS processes the data and triggers **Platform Events in Salesforce**.
- Salesforce automatically **creates a maintenance request** if an issue is detected.

Key Takeaways:

- **IoT-driven proactive service.**
- **Event-based architecture using Platform Events.**

- **Automated maintenance scheduling.**

Scenario 8: Salesforce & WhatsApp Integration (Twilio API)

Business Case:

A company wants to send **WhatsApp notifications** to customers when an order is shipped.

Solution:

- Use **Twilio WhatsApp API** for message sending.
- Configure **Apex Callout** to send WhatsApp messages based on **Order Status updates**.

Example Apex Code for Twilio WhatsApp API Callout:

```
HttpRequest req = new HttpRequest();
req.setEndpoint('https://api.twilio.com/2010-04-
01/Accounts/{AccountSID}/Messages.json');
req.setMethod('POST');
req.setHeader('Authorization', 'Basic {Base64EncodedCredentials}');
req.setBody('To=whatsapp:+123456789&From=whatsapp:+14155238886&Body=Yo
ur order has been shipped!');

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

Key Takeaways:

- **Real-time messaging improves customer experience.**
- **Twilio API simplifies WhatsApp integration.**
- **Apex callout automates message sending.**

Summary of Real-World Use Cases

Use Case	Integration Method
Payment Gateway	REST API, Webhooks

ERP Sync (SAP/MS Dynamics)	MuleSoft, Outbound Messages
Lead Enrichment	REST API, Named Credentials
Database Sync	Salesforce Connect, ETL
Ticketing (Zendesk)	Middleware, API Calls
Social Media Tracking	REST API, Einstein Sentiment
IoT Integration	Platform Events, AWS IoT
WhatsApp Messaging	Twilio API, Apex Callout