MuleSoft
from Salesforce

Mumbai MuleSoft Meetup (In Collaboration with Guwahati Meetup Group)

MuleSoft Training for Salesforce Developers and Beginners - Module 1

Date: 1st Nov 2025
Time: 11 AM to 1 PM

Safe Harbour Statement

MuleSoft from Salesforce

- Both the speaker and the host are organizing this meet-up in individual capacity only. We are not representing our companies here.

- This presentation is strictly for learning purposes only. Organizer/Presenter do not hold any responsibility that same solution will work for your business requirements.

- This presentation is not meant for any promotional activities.

Housekeeping

MuleSoft from Salesforce

A recording of this meetup will be uploaded to events page within 24 hours.

Questions can be submitted/asked at any time in the Chat/Questions & Answers Tab.

Make it more **Interactive!!!**

Give us feedback! Rate this meetup session by filling feedback form at the end of the day.

We Love Feedbacks!!! Its Bread & Butter for Meetup**.**

Organizers/Speakers

Jitendra Bafna

Senior Solution Architect
EPAM Systems

Moderators

Jitendra Bafna

Senior Solution Architect
EPAM Systems

Abhishek Bathwal
Technical Architect
NeuraFlash

What will we cover in Training?

MuleSoft from Salesforce

Module	Topic	What will we cover?	Date
Module 1	Integration & REST/HTTP Basics for Beginners & Salesforce	Integration, P2P, REST APIs, MuleSoft, Anypoint Platform	1st Nov 2025
Module 2	API Design with RAML for Beginners & Salesforce Developers	API Design with RAML, Publishing APIs to Exchange, Resources	2nd Nov 2025
Module 3	Anypoint Studio & Mule Basics for Beginners & Salesforce	API Implementation and Deploying API to CloudHub	8th Nov 2025
Module 4	Core Components, DataWeave & Error Handling Essentials	Dataweave, Error Handling, Core Components	9th Nov 2025
Module 5	Flow Control & Batch Processing for Scalable Integrations	Batch Processing, For Each, Parallel For Each	15th Nov 2025
Module 6	HTTP Connector – Listener, Requestor & Payload Handling	HTTP Connector, OAuth Module	16th Nov 2025
Module 7	Database Connector for CRUD Operations	Database connector to perform query and call store procedure	22nd Nov 2025

What will we cover in Training?

MuleSoft from Salesforce

Module	Topic	What will we cover?	Date
Module 8	Salesforce Connector for Seamless CRM Integration	Deep Dive into Salesforce Connector	23rd Nov 2025
Module 9	Hosting Options & Deploying Applications to CloudHub	ClodHub 1.0 and CloudHub 2.0	6th Dec 2025
Module 10	Managing & Securing APIs with API Manager & API Gateway	API Security, API Policies	7th Dec 2025
Module 11	MuleSoft Demo Project	Database and Salesforce related project	13th Dec 2025

Agenda

MuleSoft from Salesforce

What is Integration?	Understand the concept of system integration and how different applications communicate and share data.
What is Point-to-Point Integration?	Learn about traditional integration methods, their architecture, challenges, and limitations.
What are APIs?	Explore the fundamentals of APIs, their types, and how they enable seamless data exchange between systems.
What is MuleSoft?	Introduction to MuleSoft as an integration platform, its purpose, and how it simplifies API-led connectivity.
Overview of Anypoint Platform and Anypoint Studio	Get familiar with MuleSoft's key components — Anypoint Platform for API lifecycle management and Anypoint Studio for integration development.
Designing APIs using RAML	Learn the basics of RAML (RESTful API Modeling Language) and how to design well-structured, reusable APIs using MuleSoft tools.

MuleSoft
from Salesforce

Point-To-Point Integration

Point to Point Integration

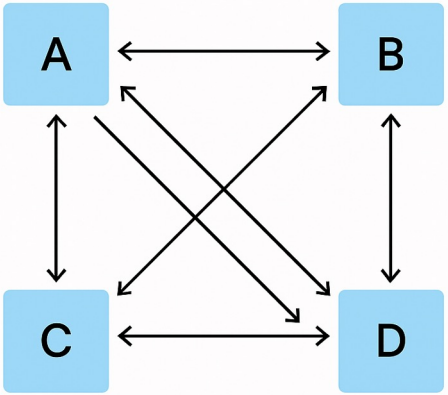
Point-to-point integration" is a method of connecting two systems, applications, or services directly to each other without using a central hub, middleware, or integration platform.

Each system needs its **own direct connection** to every other system it communicates with.

For example:

- If you have **System A** and **System B**, you make one connection: **A** ↔ **B**.
- If you add **System C**, you need connections: **A** ↔ **C** and **B** ↔ **C**.
- As more systems are added, the connections grow very quickly.

Point-to-Point Integration



Example: 2 systems, 1 connection

A → B

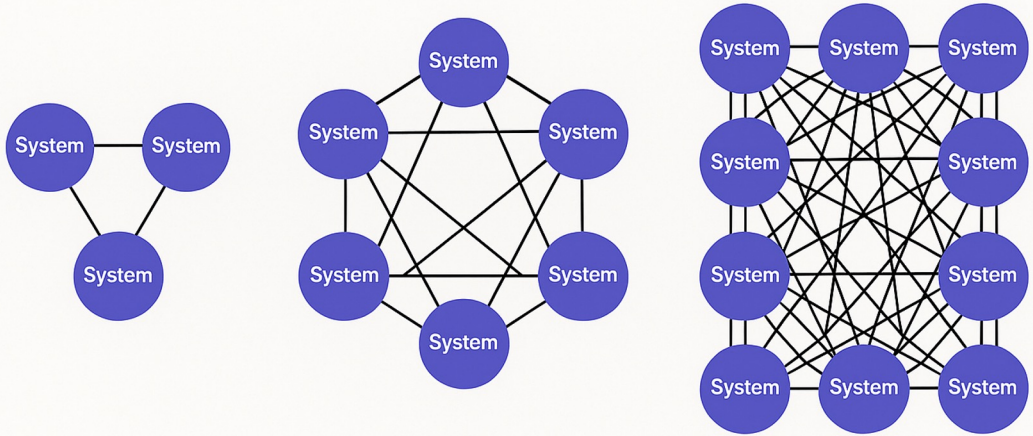
Example: 3 systems, 3 connections

A → B, A → C, B → C

Example: 4 systems, 6 connections

A → B, A → C, A → D, B → C, B → D, C–D

Point-to-Point Integration



Step 1

Number of connections: 3

Step 2

Number of connections: 10

Step 3

Number of connections: 45

Advantages

✅ **Simple to start** – quick and easy when you only have 2–3 systems.

✅ **Direct communication** – data moves straight from one system to another.

✅ **Low initial cost** – no fancy middleware or platform needed.

Disadvantages

- **Scalability issues**: Each new system requires multiple new connections. For *n* systems, you may need *n(n-1)/2* connections.
- **Maintenance burden**: Any change in one system's API or format can break the integration.
- **Spaghetti architecture**: With many systems, the network of connections becomes messy and hard to manage.
- **Limited flexibility**: Hard to add business logic or transformations compared to using middleware.

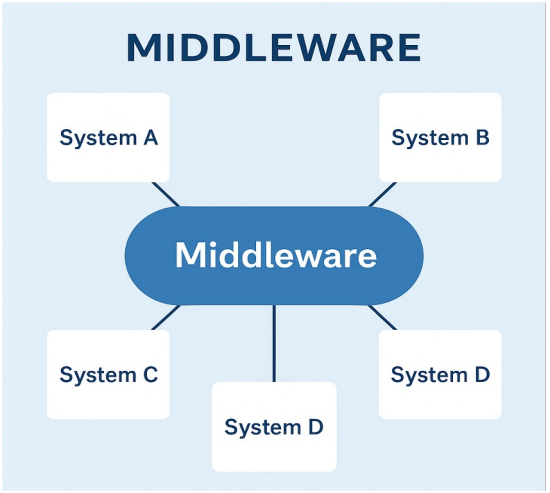
Common Use Cases

- Small businesses with just a few systems.
- Simple integrations (e.g., syncing contacts between two apps).
- One-off data exchanges.

What is Integration?

What is Integration?

- **Integration** means connecting different systems, applications, or software so they can **work together** and **share data** smoothly.
- Instead of each system working in isolation, integration creates a **unified flow of information**.



Point-To-Point Integration Problem

The Problem with Point-to-Point

In **point-to-point integration**, every system connects **directly** to every other system.

- Works fine for 2–3 systems.
- But with 5, 10, or 20 systems → the number of connections grows super fast → becomes a **spaghetti mess**.
- Hard to maintain, costly to scale, error-prone.

How integration helps?

To solve these problems, businesses use **centralized integration approaches** like:

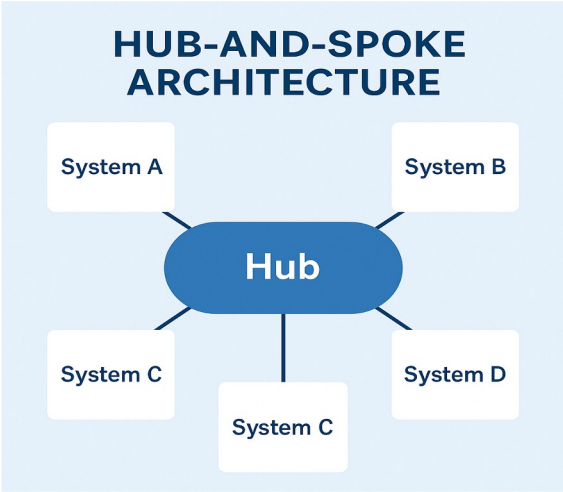
1. Hub-and-Spoke Integration

- All systems connect to a **central hub** (middleware, ESB, or iPaaS).
- The hub manages data flow, transformations, and rules.
- Instead of many connections, each system needs just **one connection to the hub**.

Example:

- System A ↔ Hub
- System B ↔ Hub
- System C ↔ Hub

  Now, A can talk to B or C through the hub → much cleaner!



Event Driven Architecture

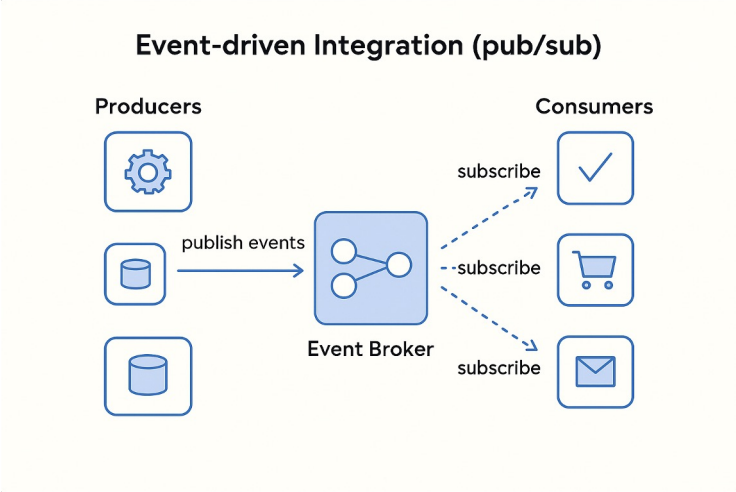
Event-driven integration is like a **newspaper subscription model**:

- A **publisher** (system that generates an event) posts news.
- A **broker** (like a post office) distributes that news.
- **Subscribers** (systems that care about that news) get it automatically.

👉 **Example:**

- **E-commerce app** publishes an event: *"Order Placed"*.
- **Inventory system** subscribes → reduces stock.
- **Shipping system** subscribes → creates shipment.
- **Email system** subscribes → sends confirmation email.

The publisher doesn't know who is listening — it just posts events.



🔷 **Why It's Useful**

- **Loosely coupled**: Systems don't depend directly on each other.
- **Scalable**: Many subscribers can consume the same event.
- **Real-time**: Changes spread instantly.
- **Flexible**: Easy to add new subscribers later without touching the publisher.

What is REST APIs?

What is REST APIs?

REST API integration is a way for two or more applications to communicate and share data over the internet using **REST (Representational State Transfer)** principles.

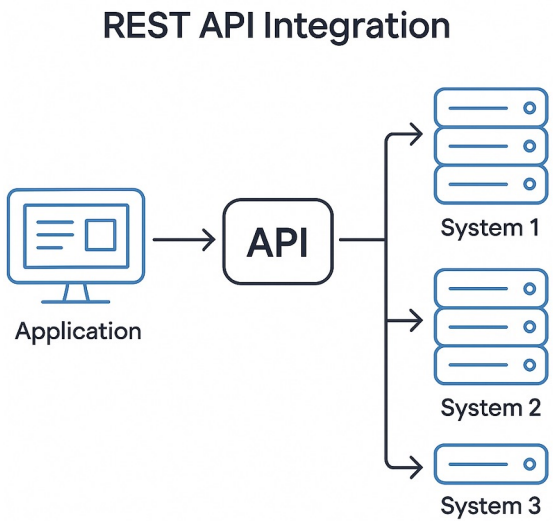
It relies on standard web methods (HTTP: GET, POST, PUT, DELETE) and usually exchanges data in **JSON format**.

🔹 Simple Explanation

Think of a **REST API** like a **waiter in a restaurant**:

- You (the client) tell the waiter what you want.
- The waiter (API) takes your request to the kitchen (the system).
- The kitchen prepares the food (data) and the waiter brings it back to you.

So instead of each system building a custom link to talk, they all just **use the API menu**.



✅ **Universal language** → any app that understands HTTP can use it.
✅ **Reusable** → once built, the same API can serve multiple apps.

✅ **Scalable** → works well for small and large systems.

✅ **Loose coupling** → systems don't need to know each other's internals.

Key Principles

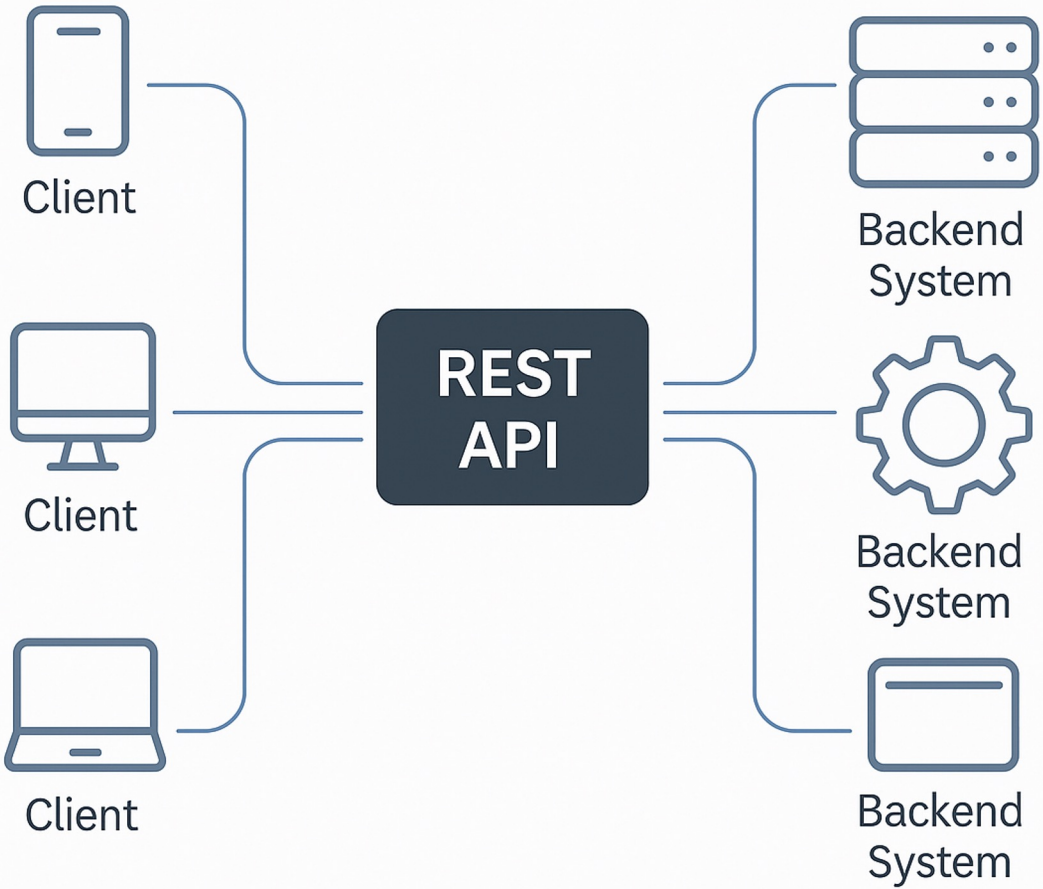
- **Stateless** – Every request from client to server must contain all the information needed (server doesn't remember previous requests).
- **Client-Server Separation** – The client (UI/mobile app) and the server (database/backend) are independent.
- **Uniform Interface** – Standard methods (GET, POST, PUT, DELETE) make APIs predictable and easy to use.
- **Resource-Based** – Everything is treated as a "resource" (user, product, order), identified by a unique URL.
  - Example: `/users/101` → represents User with ID 101.
- **Cacheable** – Responses can be cached to improve performance.

HTTP Methods

Method	Meaning	Example URL	What Happens?
GET	Read data	`GET /users`	Returns a list of all users
POST	Create data	`POST /users`	Creates a new user in the system
PUT	Replace data	`PUT /users/101`	Updates all details of user 101
PATCH	Update part	`PATCH /users/101`	Updates only specific fields (e.g., email)
DELETE	Remove data	`DELETE /users/101`	Deletes user 101 permanently
HEAD	Get headers	`HEAD /users`	Checks if resource exists (without data)
OPTIONS	Allowed methods	`OPTIONS /users`	Shows which HTTP methods are supported

REST APIs Benefits

🌍 **Universal** → Any system (Java, Python, .NET, Salesforce, etc.) can use it.
⚡ **Fast & Lightweight** → Works over HTTP, no extra overhead.
🔌 **Easy Integration** → Connects apps, databases, and services smoothly.
🔄 **Scalable** → Can handle millions of requests easily because it's stateless.
🛠️ **Maintainable** → Client and server can be updated separately.
🔁 **Reusable** → Same API can serve **web, mobile, and third-party apps**.
📊 **Widespread adoption** → REST is the most common API style in the industry.


Client
Client
Client
REST API
Backend System
Backend System
Backend System

Role of REST APIs in Digital Transformation

REST APIs act as the **bridge** between legacy systems, new applications, and digital platforms.

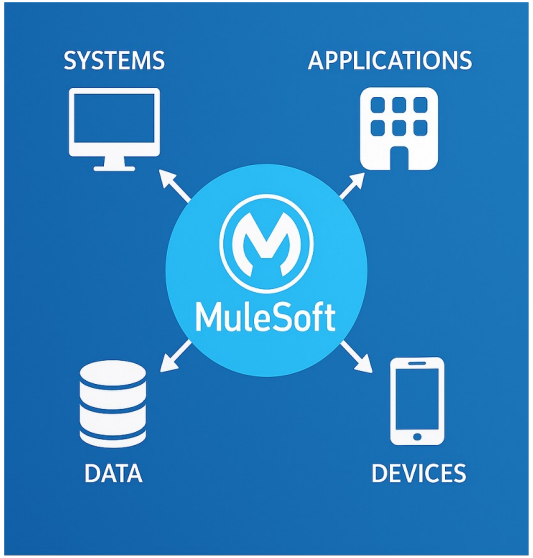
- **Connectivity** → Connects mobile apps, web apps, IoT, and cloud services.
- **Agility** → Enables businesses to quickly launch new apps/features by reusing existing APIs.
- **Scalability** → Handles growing user demands with stateless architecture.
- **Integration** → Easily integrates older systems (ERP, CRM) with modern cloud solutions.
- **Ecosystem Expansion** → Opens opportunities for third-party integrations (e.g., partners, marketplaces).

MuleSoft
from Salesforce

What is MuleSoft?

What is MuleSoft?

MuleSoft is an **Integration Platform as a Service (iPaaS)** that helps organizations connect **data, applications, devices, and systems**—both on-premises and in the cloud. It acts as a **central hub** that ensures different technologies "talk" to each other seamlessly, enabling automation, efficiency, and innovation across the business.



Key Highlights

- **Universal Connectivity** → Connects any system, database, application, or device, whether cloud-based or on-prem.
- **API-Led Approach** → Uses APIs (Application Programming Interfaces) to make integrations reusable, secure, and scalable.
- **Accelerates Digital Transformation** → Breaks down silos and allows organizations to respond faster to market needs.
- **Automation & RPA** → Automates repetitive business processes and workflows across applications.
- **Real-Time Data Sharing** → Ensures that data flows smoothly, so customers, employees, and partners always have up-to-date information.

Example in Action

- A **retail company** connects its **e-commerce platform**, **inventory system**, **payment gateway**, and **Salesforce CRM** with MuleSoft.
- When a customer places an order, MuleSoft automatically updates stock, processes payment, and updates customer details in Salesforce—all in real-time.

Why MuleSoft?

Business Problem

Companies today use **hundreds of applications** — CRMs (Salesforce), ERPs (SAP), payment gateways, HR systems, databases, mobile apps, IoT devices, etc. These systems are often **isolated (silos)** and don't easily share data.

Without MuleSoft:

- Teams build **point-to-point integrations** (custom code between apps).
- This becomes **complex, expensive, slow, and hard to maintain**.

The MuleSoft Solution

MuleSoft solves this by:

- Acting as a **central integration platform**.
- Connecting apps, systems, and data through **APIs**.
- Promoting **reusable building blocks** instead of re-inventing the wheel.
- Supporting both **cloud and on-premise** systems.

Key Reason to Choose MuleSoft

✅ Faster Time-to-Market

- Build integrations quickly with pre-built connectors and templates.
- Launch new apps and digital services faster.

✅ Reusable APIs

- Create APIs once and reuse them across multiple projects.
- Cuts down development time and costs.

✅ Scalability

- Grows with your business.
- Works for startups, enterprises, and global organizations.

✅ Hybrid & Multi-Cloud Support

- Works across **cloud, on-premise, and hybrid** environments.

Key Reason to Choose MuleSoft

✅ Security & Governance

- Built-in tools to secure APIs and control access.
- Ensures compliance and safety of sensitive data.

✅ Part of Salesforce Ecosystem

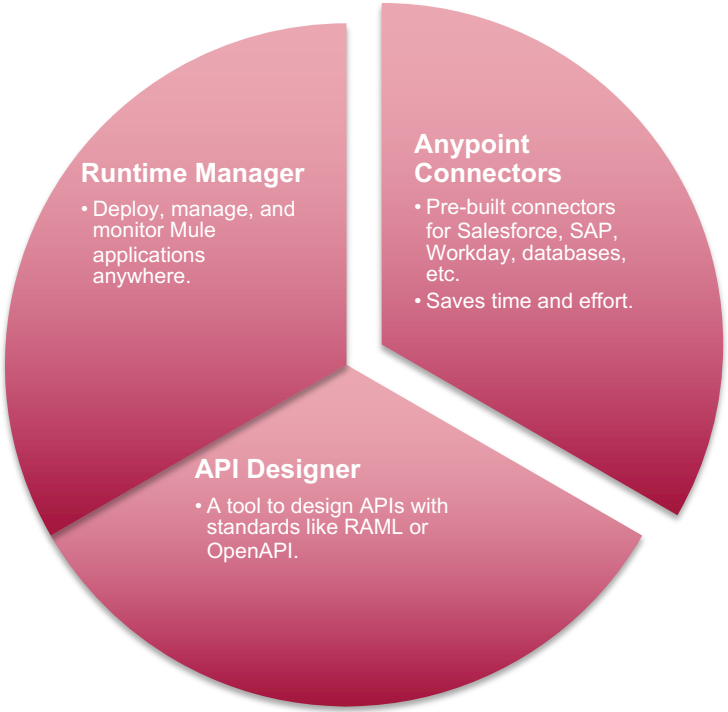
- Seamlessly connects customer data into Salesforce.
- Enables **360-degree customer view**.

Key Reason to Choose MuleSoft

MuleSoft from Salesforce

Anypoint Studio	A graphical IDE to design and test integrations (drag-and-drop). Developers can create flows to connect systems.
Anypoint Exchange	A marketplace for APIs, templates, and connectors. Encourages **reuse instead of rebuilding**.
Anypoint Management Center	A centralized dashboard for managing, monitoring, and securing APIs and apps.
Mule Runtime Engine	The “engine” that executes integration flows and APIs. Runs on-premise, in the cloud, or hybrid.

Key Reason to Choose MuleSoft

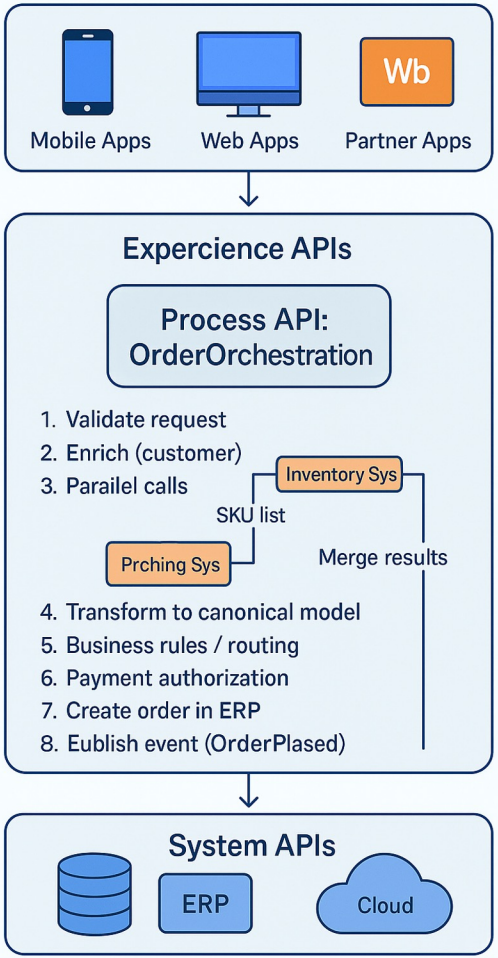


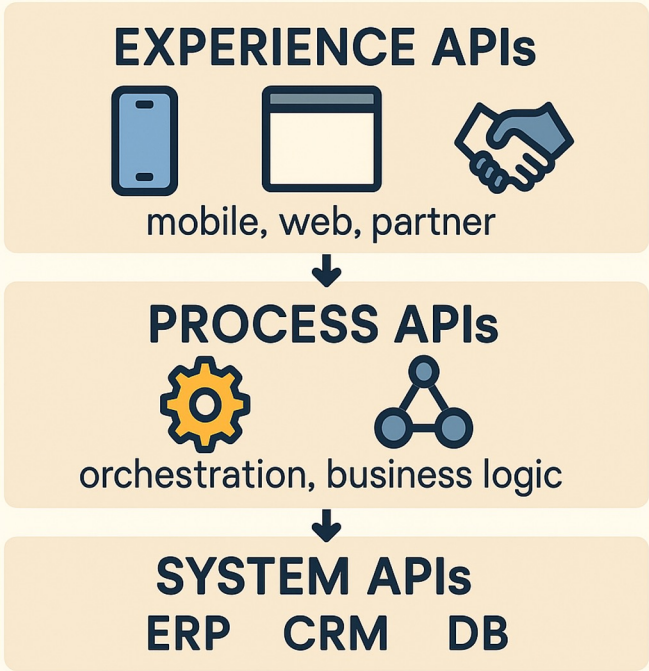
API Led Connectivity

MuleSoft from Salesforce

How MuleSoft Works (API-led Connectivity)

- MuleSoft promotes a **layered approach** with APIs:
  - **System APIs** – Connect directly to core systems (databases, ERP, CRM).
  - **Process APIs** – Combine, filter, and process data from multiple systems.
  - **Experience APIs** – Deliver customized data to applications (web, mobile, partners).
- This **separates concerns**, makes APIs reusable, and reduces duplication.


Mobile Apps
Web Apps
Wb
Partner Apps
Expercience APIs
Process API:
OrderOrchestration
1. Validate request
2. Enrich (customer)
3. Parailel calls
Inventory Sys
SKU list
Prching Sys
Merge results
4. Transform to canonical model
5. Business rules / routing
6. Payment authorization
7. Create order in ERP
8. Eublish event (OrderPlased)
System APIs
ERP
Cloud


EXPERIENCE APIs
mobile, web, partner
PROCESS APIs
orchestration, business logic
SYSTEM APIs
ERP CRM DB

Benefits of API Led Connectivity

MuleSoft from Salesforce

Reusability	Build APIs once → reuse across multiple projects.
Faster Delivery	Quickly create new apps and services by using existing APIs.
Flexibility	Easy to plug in new systems or replace old ones without breaking everything.
Better Collaboration	Business and IT can work together → APIs are clear building blocks.

Benefits of API Led Connectivity

MuleSoft from Salesforce

Scalability

- Supports small projects to large enterprise ecosystems.

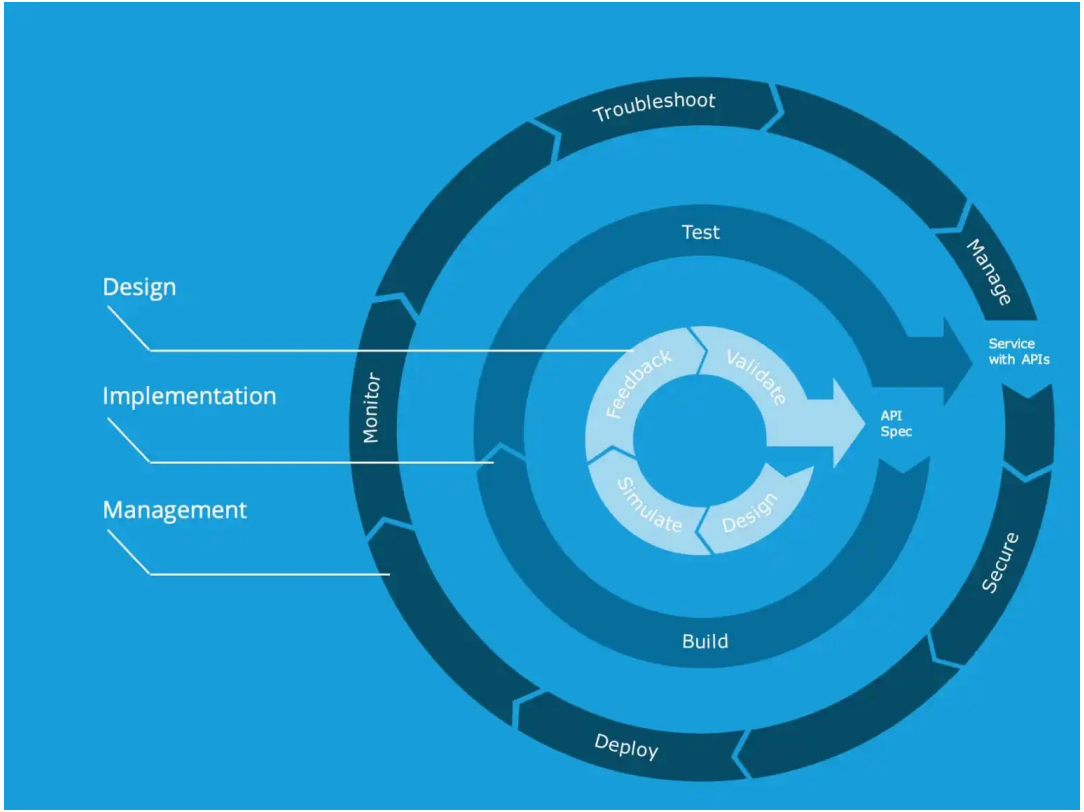
Security & Control

- Each API layer can be secured and managed separately.

Reduced Costs

- Less custom coding, fewer errors, easier maintenance.

MuleSoft
from Salesforce



API Lifecycle Phase 1: Design (Designing API with RAML)

MuleSoft from Salesforce

Use Case

MuleSoft from Salesforce

We need to develop an API that integrates with Salesforce to manage Account and Contact information. The API should support the following functionalities:

- **Create Accounts** – Add new Account records in Salesforce.
- **Create Contacts for Accounts** – Add new Contact records associated with specific Accounts.
- **Retrieve Accounts** – Fetch existing Account details from Salesforce.
- **Retrieve Contacts for Accounts** – Fetch Contact details linked to specific Accounts.

API Lifecycle Phase 2,3,4: Simulate, Feedback and Validate API (Publishing API to Anypoint Exchange)

MuleSoft
from Salesforce

Thank You