Mumbai MuleSoft Meetup (In Collaboration with Guwahati Meetup Group)

MuleSoft Training for Salesforce Developers and Beginners - Module 2

Date: 2nd Nov 2025
Time: 11 AM to 1 PM

Safe Harbour Statement

- 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

MuleSoft
from Salesforce

Jitendra Bafna

Senior Solution Architect
EPAM Systems

Moderators/Speakers

Jitendra Bafna

Senior Solution Architect
EPAM Systems

Abhishek Bathwal
Technical Architect
NeuraFlash

What will we cover in Training?

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?

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

What have we learned on Day 1?

- What is Point-To-Point Integration?
- What is Integration?
- What is REST APIs?
- What is MuleSoft and Anypoint Platform?
- Walkthrough of Anypoint Platform.
- Understanding the API Lifecycle Management.
- Design the RAML to create and fetch Account and Contacts from Salesforce.
- Published API to Anypoint Exchange.

What will we learn on Day 2?

- What is RAML?
- Reusability of RAML using Traits, Library, Security Schemes.
- OAS (Open API Specification)
- API Governance
- Overview of Anypoint Studio
- Start with API Implementation.

RAML (RESTful API Modeling Language)

What is RAML?

RAML (RESTful API Modeling Language) is a **way to describe how an API works** — what endpoints it has, what data it accepts, and what it returns — all written in a **simple, human-readable format (YAML)**.
Think of it like a **blueprint or recipe for an API**.

It helps developers:

- **Plan** an API before building it
- **Document** it clearly for others to use
- **Reuse** parts easily (like data types or examples)

Key Points

🧠 **Full Form:** RESTful API Modeling Language

🏗️ **Main Use:** To design and model RESTful APIs before building them

🏢 **Created By:** MuleSoft

🗂️ **File Format:** Written in YAML (easy to read)

🎯 **Purpose:** Focuses on API design-first — plan your API before coding

🧩 **Reusability:** You can reuse common parts like data types, traits, and examples

📘 **Readable:** Simple, human-friendly syntax

🧱 **Structure:** Uses fragments to organize large APIs into smaller files

🧰 **Tool Support:** Works best with MuleSoft's Anypoint Studio and Design Center

📄 **Documentation:** Automatically generates API documentation from the design

🔄 **Versioning:** Common versions — RAML 0.8 and 1.0

🔌 **Integration:** Deeply integrated with MuleSoft tools

🔒 **Standardization:** Encourages consistent and reusable API structures

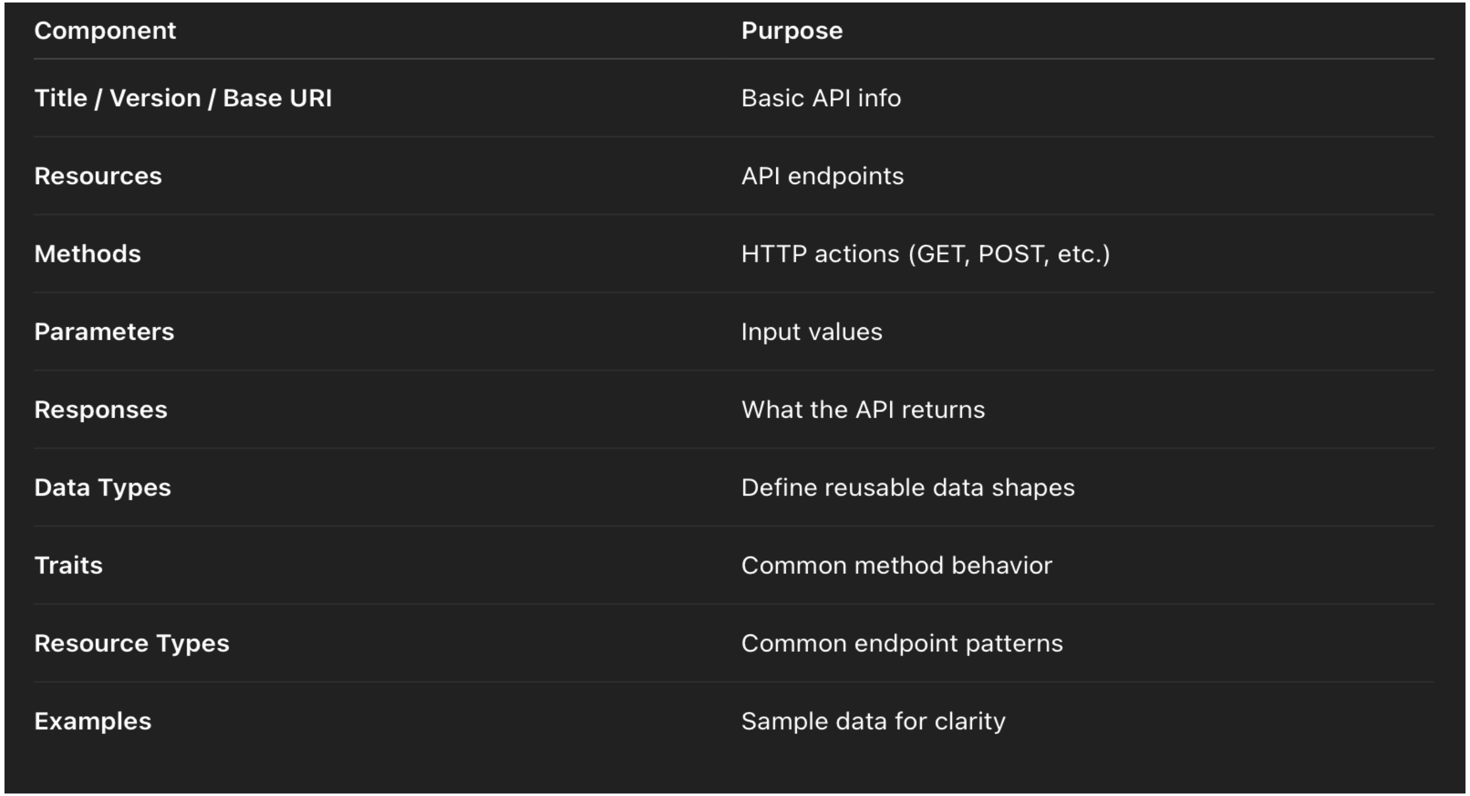
🌍 **Adoption:** Mostly used within the MuleSoft ecosystem

🔧 **Conversion:** Can be converted to OpenAPI (OAS) format

🧠 **Learning Curve:** Easy to learn, especially for MuleSoft developers

📈 **Main Benefit:** Makes API design clear, consistent, and faster

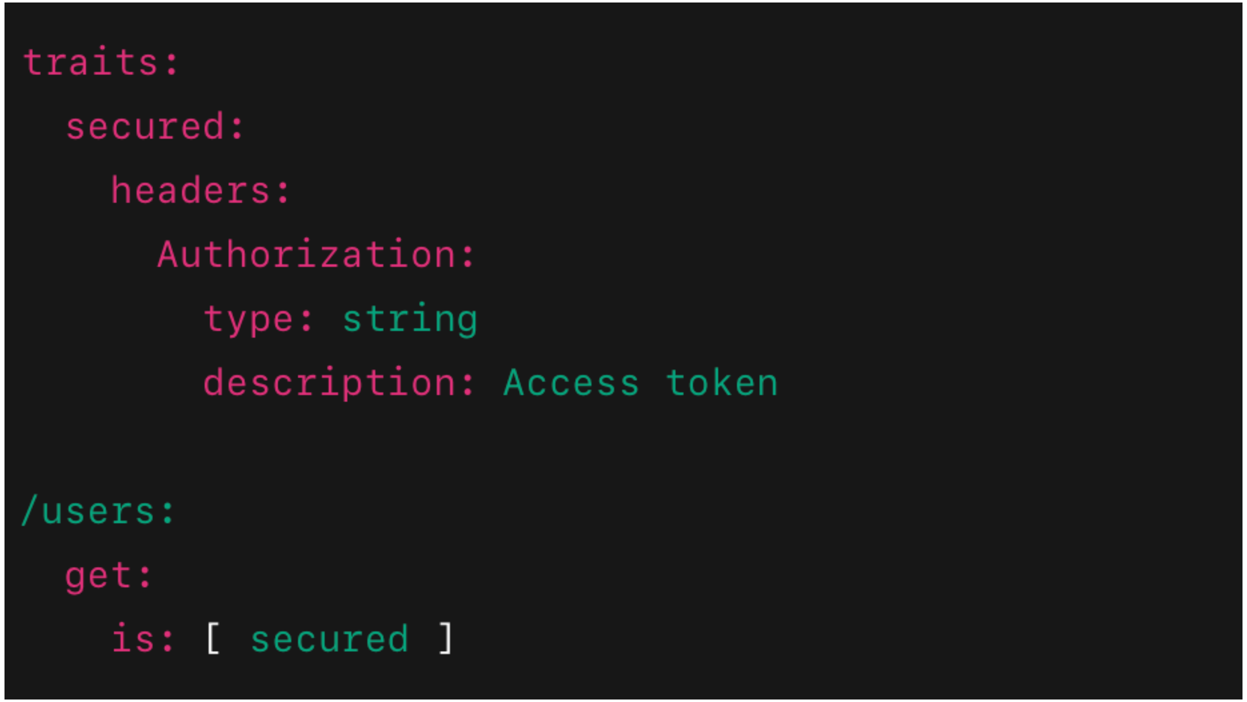
Components of RAML



Component	Purpose
Title / Version / Base URI	Basic API info
Resources	API endpoints
Methods	HTTP actions (GET, POST, etc.)
Parameters	Input values
Responses	What the API returns
Data Types	Define reusable data shapes
Traits	Common method behavior
Resource Types	Common endpoint patterns
Examples	Sample data for clarity

Traits

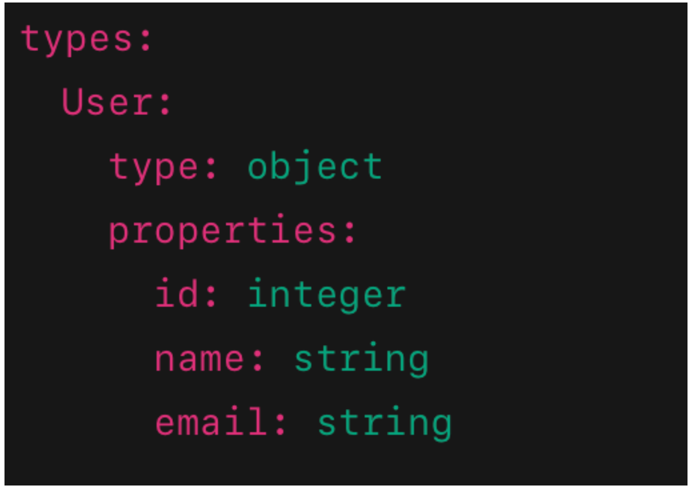
A trait in RAML is a reusable rule or feature that you can add to many API methods to avoid repeating the same details.



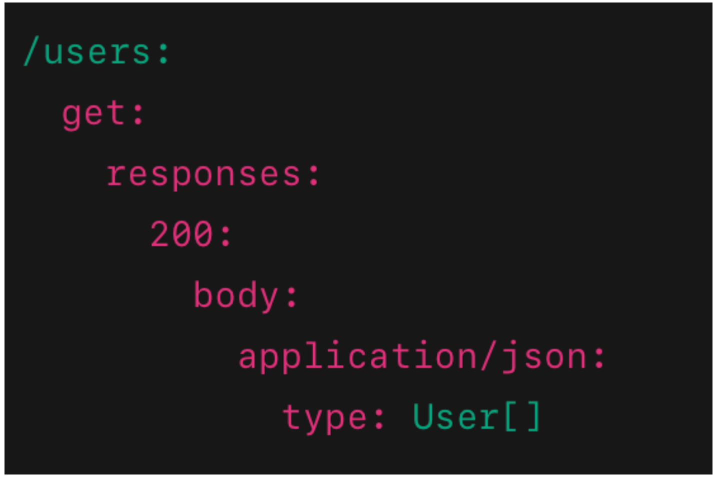
```
traits:
  secured:
    headers:
      Authorization:
        type: string
        description: Access token

/users:
  get:
    is: [ secured ]
```

Data Types

A data type in RAML is a template for data — it defines what kind of information an object has (like fields, their types, and rules).



```
types:
  User:
    type: object
    properties:
      id: integer
      name: string
      email: string
```



```
/users:
  get:
    responses:
      200:
        body:
          application/json:
            type: User[]
```

Library

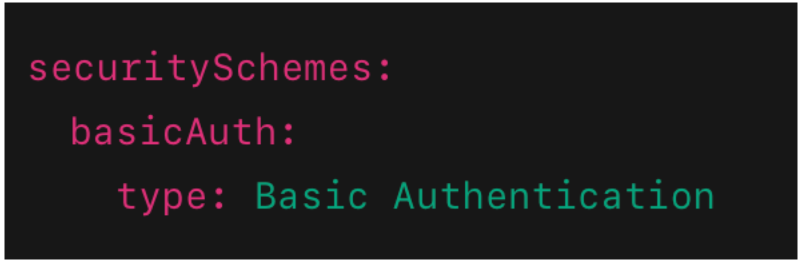
A library in RAML is a separate file where you can store reusable parts of your API — such as data types, traits, resource types, or security schemes — so that you can reuse them in different APIs without rewriting everything.

```
#%RAML 1.0 Library

types:
  User:
    type: object
    properties:
      id: integer
      name: string

traits:
  secured:
    headers:
      Authorization:
        type: string
        description: Access token
```

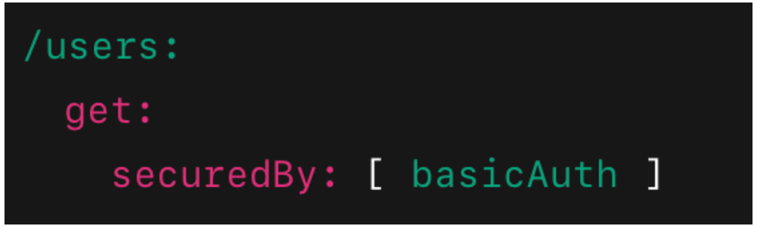
```
#%RAML 1.0
title: My API
uses:
  common: common-library.raml    # Import the library

/users:
  get:
    is: [ common.secured ]      # Using the trait from library
    responses:
      200:
        body:
          application/json:
            type: common.User[] # Using the type from library
```

Security Schemes

A security scheme in RAML is a way to describe how your API is protected — for example, using an API key, OAuth 2.0, or Basic Authentication.



```yaml
securitySchemes:
  basicAuth:
    type: Basic Authentication
```



```yaml
/users:
  get:
    securedBy: [ basicAuth ]
```

Anypoint Studio

Anypoint Studio is a software tool made by MuleSoft.

It helps developers design, build, test, and run integrations between different systems — like databases, APIs, cloud apps, and more. Think of it like an IDE (Integrated Development Environment) — similar to Eclipse or IntelliJ — but specifically for creating and managing Mule applications that connect different systems together.

Anypoint Studio Capabilities

- 🧩 **API and integration development** – Build Mule applications to connect apps, data, and services easily.
- 🎨 **Graphical design environment** – Drag-and-drop interface for creating data flows and transformations without heavy coding.
- 🔌 **Connector support** – Comes with built-in connectors for Salesforce, SAP, HTTP, Databases, FTP, and many others.
- 🔁 **Data transformation (DataWeave)** – Allows converting data between formats like JSON, XML, CSV, etc.
- 🧪 **Testing and debugging tools** – You can run, debug, and test Mule applications directly within the Studio.
- 🚀 **Deployment options** – Deploy apps to CloudHub, Anypoint Runtime Manager, or on-premise Mule runtimes.
- 📄 **API design and documentation** – Integrates with Anypoint Platform to design and document REST or SOAP APIs.

Anypoint Studio Capabilities

- 🔒 **Error handling and security** – Built-in components for managing exceptions and applying security policies.
- 📈 **Performance monitoring** – Works with Anypoint Monitoring to track app performance and health.
- 🧠 **Version control support** – Compatible with Git for managing and sharing code across teams.
- ☁️ **Integration with Anypoint Exchange** – Reuse existing connectors, templates, and examples from MuleSoft's repository.

What is OAS?

OAS stands for **OpenAPI Specification**. It is a **standard way to describe APIs** — like a **blueprint** that explains how an API works.
It tells people:

- what the API does,
- what information it needs, and
- what it gives back.

Key Points

- 📘 **Describes REST APIs** — what endpoints exist, what parameters they need, and what responses they return.
- 🗂️ **Written in YAML or JSON** — easy for both people and programs to read.
- 🧰 Used by many tools like SwaggerHub, Postman, and Stoplight.
- 🌍 **Industry Standard** — used worldwide by API developers.
- 🧱 **Helps create API docs automatically** — tools can turn OAS files into nice documentation or even test APIs automatically.

RAML V/S OAS

Feature	RAML	OAS (OpenAPI Specification)
Full Form	RESTful API Modeling Language	OpenAPI Specification
Also Known As	—	Swagger
Purpose	Used to design and model APIs	Used to describe and document APIs
Created By	MuleSoft	OpenAPI Initiative (Linux Foundation)
File Format	YAML	YAML or JSON
Focus Area	Design-first approach	Documentation-first and industry-standard
Readability	Easy and clean syntax	Easy but more verbose
Supported Tools	Anypoint Studio, Design Center	SwaggerHub, Postman, Stoplight, etc.
Use in MuleSoft	Fully supported and preferred	Supported but requires import/conversion
Industry Adoption	Mostly within MuleSoft ecosystem	Global standard used widely
Versioning	0.8, 1.0	2.0, 3.0, 3.1
Reusability	Strong support for reusing fragments and data types	Has components for reuse but less modular than RAML
Documentation Quality	Structured and readable	Widely recognized and tool-supported
Extensibility	Allows reusable templates and traits	Supports extensions via vendor tags (x-...)
Learning Curve	Easier for MuleSoft users	Easier for general API developers
Tooling Ecosystem	Tight integration with MuleSoft products	Large open ecosystem
Conversion	Can be converted to OAS	Can be converted from RAML
Common Use Case	Designing APIs before implementation	Sharing APIs with clients or public developers
Example Platform Use	Anypoint Design Center	SwaggerHub, Postman, Redoc
Community Support	Smaller, MuleSoft-focused	Very large, open community
Specification Governance	Managed by MuleSoft	Managed by OpenAPI Initiative
Example Syntax Simplicity	Cleaner, simpler for modeling	Richer for documentation and examples

Thank You