**Basic Interview Question:**

**What are the interfaces developed by you? Explain the flow?**

A. Explain the interfaces (how mule is involved in the integration & transformation) and APIs flow of your project. Also, explain the various connectors you have used for the implementation.

**Which version of Mule you have used?**

A. Based on your experience you can tell either Mule 3.9 or Mule 4.

**What are the frontend & backend applications?**

A. Explain the frontend application which sends the payload to Mule and the backend application to which the payload is sent.

**What HTTP methods you have used**

A. GET/POST calls (mention based on your project experience)

**Difference between REST & SOAP services**

A. REST API has no official standard at all because it is an architectural style.

SOAP API, on the other hand, has an official standard because it is a protocol.

REST APIs use multiple standards like HTTP, JSON, URL, and XML while SOAP APIs are largely based on HTTP and XML.

SOAP API defines too many standards, and its implementer implements the things in a standard way only. In the case of miscommunication from service, the result will be the error. REST API, on the other hand, doesn’t make emphasize on too many standards and results in corrupt API in the end.

REST API uses Web Application Description Language, and SOAP API used Web Services Description language for describing the functionalities being offered by web services.  
REST APIs are more convenient with JavaScript and can be implemented easily as well. SOAP APIs are also convenient with JavaScript but don’t support for greater implementation.

**What is Mule ESB?**  
Mule ESB is a Java-based enterprise service bus (ESB) and integration platform, a developer can connect their application with ESB. Mule use service-oriented architecture. Apart from the different technologies the applications use, including JMS, Web Services, SMTP, HTTP. The advantage of ESB, it’s allow communicate different application. Messages can be any format SOAP to JSON. Mule ESB Development provides a messaging framework that enables the exchange of data among application.

**What is the definition of Web Services?**  
Web service is a function or program in any language that can be accessed over HTTP. Message format can be XML or JSON or any other program as long as the other programs can understand and communicate. Any web service has a server-client relationship. Web services can be synchronous or asynchronous. Any web service can have multiple clients.

**Difference between URI & query parameters**

A. URI parameter  is basically used to identify a specific resource or resources whereas a query parameter is used to sort/filter those resources

**What are the core principles of ESB Integration?**

**Transportation —**Transport protocol negotiation between different formats including JMS, JDBC, HTTP, etc.

**Transformation —** Transformation of data between certain data formats as needed by every ESB connector individually

**Non-functional consistency** — The way monitoring and security policies are applied as well as implemented should be consistent

**Mediation** — This involves offering many different interfaces to:

* Enable different channels to the same unlying component implementation
* Support different service versions for backwards compatibility

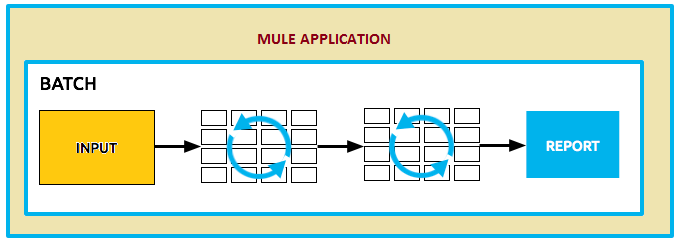
**Explain Flow in Mule ESB.**

A flow is the construct within which you link together several individual elements to handle the receipt, processing, and eventual routing of a message.

A Flow is a sequence of message-processing events. A message that enters a flow may pass through a wide variety of processors. In the example diagram below, Mule receives the message through a request-response inbound endpoint, transforms the content into a new format, and processes the business logic in a component before returning a response via the message source.

**Explain Batch Jobs in Mule ESB.**

A batch job is a top-level element in Mule ESB which exists outside all Mule flows. Batch jobs split large messages into records which Mule processes asynchronously; just as flows process messages, batch jobs process records.



A batch job contains one or more batch steps which, in turn, contain any number of message processors that act upon records.

A batch job executes when triggered by either a batch executor in a Mule flow or a message source in a batch-accepting input. when triggered, Mule creates a new batch job instance. When all records have passed through all batch steps, the batch job instance ends and the batch job result can be summarized in a report to indicate which records succeeded and which failed during processing.

**What is the status code for successful POST call**

A. 201

**What is Mule Cache Scope and what are its storage types?**

Caching in Mule ESB can be done by Mule Cache Scope. Mule Cache Scope has 3 storage types –

**In-memory:** This store the data inside system memory. The data stored with In-memory is non-persistent which means in case of API restart or crash, the data been cached will be lost.

Configuration Properties:

* Store Name
* Maximum number of entries
* TTL (Time to live)
* Expiration Interval

**Managed-store:** This stores the data in a place defined by ListableObjectStore. The data stored with Managed-store is persistent which means in case of API restart or crash, the data been cached will no be lost.

**HTTP Status codes**

1xx Informational.  
2xx Success scenarios  
3xx Redirection Errors  
4xx Client Errors  
5xx Server Errors

**RAML Interview Questions in Mulesoft interviews**

**What is RAML?**

A. RESTful API Modeling Language (also known as RAML) is a YAML-based modeling language to define RESTful APIs. It provides a structured and rich format to define the API. According to RAML’s website, it “makes it easy to manage the whole API lifecycle.”

RAML can be used in a multitude of ways: to implement interactive API consoles, create documentation, describing an API you are planning to build, and more. Regardless of the name, RAML can describe APIs that do not follow all of the REST rules

**What are the traits and resource type?**

A. Traits are like function and are used to define common attributes for HTTP method (GET, PUT, POST, PATCH, DELETE, etc) such as whether or not they are filterable, searchable, or pageable

ResourceType is like a template that is used to specify the descriptions, methods, and parameters that can be used by multiple other resources without the need to rewrite the duplicate code or repeating code.

**Can we use two or more methods for a single resource?**

1. No, we can use only one HTTP method once for single resource in RAML. Example: If a resource is “/employees/employee” then we can define GET, PUT POST, PATCH or DELETE HTTP methods only once.

**What connectors have you used?**

A. Salesforce, Database, JMS, HTTP, File etc

**Difference between VM queue & JMS**

A. VM Transport is for intra JVM communication between Mule flows. We can use VM transport to communicate with different flows of the same application whereas JMS is used for communicating with the external applications also.

**How to filter duplicate records?**

A. Using Idempotent filter to discard duplicate records

**What are the core components of Mule ESB?**

A. Scheduler, For Each, Logger, Batch (Batch Aggregator, Batch Job, Batch Step), Flow reference, Transform message, Error handling Routers, Scopes and Transformers. Refer [Core components of Mule](https://docs.mulesoft.com/mule-runtime/4.2/about-components" \t "_blank)

How to read a file as input other than FTP connector

**Scenario:** How to fetch and process 5M records(few duplicates) into DB

**Different types of flows, flow processing strategies**

A. Flow processing strategies in Mulesoft are:

* Synchronous Flow Processing Strategy
* Queued Flow Processing Strategy
* Asynchronous Flow Processing Strategy
* Thread Per Processing Strategy
* Queued Asynchronous Flow Processing Strategy
* Non-blocking Flow Processing Strategy
* Custom Processing Strategy

**Difference between sub-flow, private flow**

A. **Subflow** processes messages synchronously but inherits processing strategy and exception handling strategy from the calling flow (parent flow). It can be used to split common logic and be reused by other flows.

**Private flow** does not use a source. It can be synchronous or asynchronous based on the processing strategy selected. Also, has its own exception handling strategy. Allows you to define different threading profile.

**How can we create a custom policy in Anypoint Studio?**

A. Navigate to File > New > API custom policy Project(Beta) in Anypoint Studio

**Difference between async flow & async scope**

*Async Flow:*Async Flow is a transactional flow in which the requestor does not wait for the response from the target service and processes the flow

*Async scope:* The Async scope is a branch processing block that executes simultaneously with the main flow. The main flow continues to execute while it initiates and processes the Async scope

**How do you consume a SOAP-based web service in Studio**

A. Using the Web Service Consumer component. Configure the connector with WSDL or SOAP web service configuration details like Service name, URL, Binding port, etc.

Steps involved in MUnit creation

**How do you set environment-specific values in Mule application?**

At first, Create a properties file for each development environment in your application. Configure a properties placeholder in your app to look for the environment upon launch. Set an environment variable to point to a specific development environment during application deployment.

Explain Collection aggregator, splitter, object store, cache

**What are the variables available in Mule 3?**

A. Flow variables, Record variables, Session variables