1. What is Integration?

Ans:

In Simple Words, Integration is exchanging data b/w two cross platforms.

2. What is API (Application Program Interface)?

Ans:

An API (Application Program Interface) is a way for you to get information needed in a consistent format.

For e.g., some of applications allows you to login through gmail accounts, right. Because those applications basically using google api for the same.

Another example is weather applications, those applications are also using some api to get information from the satellites. So, here api is actually a middle man between the two applications.

3. What is a “ServiceNow® integration”?

Ans:

A ServiceNow integration is an information exchange between the Now Platform® and another system or source. Integrations are essential to implementing digital workflows and creating seamless user experiences.

ServiceNow integrations support three use cases:

• Process integrations move data across systems to facilitate a process, like sending ServiceNow records to Jira to support a development process.

• UI integrations present public web information within the Now Platform, such as displaying weather forecasts

. • Data integrations enable data sharing, such as viewing demographic HR data, like titles or managers, within the platform.

4. What is Rest API and its Methods?

Ans:

REST API integration refers to connecting with third-party applications using HTTPS requests to access and transmit data.

REST (Representational State Transfer) is a simple stateless architecture that provides standards between computer systems on the web, making it easier for them to communicate with each other.

The Now Platform provides various REST APIs, which are active by default. These APIs provide the ability to interact with various ServiceNow functionality within your application. Such functionality includes the ability to perform create, read, update, and delete (CRUD) operations on existing tables (Table API), insert data into, retrieve information from,

That data can be used to GET, PUT, POST and DELETE data types, which refers to the reading, updating, creating and deleting of operations concerning resources.

- Post (used to create information)
- Get (used to read information or resource)
- Put (used to update the information or resource)
- Delete (used to delete the information or resource)
- Patch (used to update an existing entity with new information. You can't patch an entity that doesn't exist.)

5. What is REST API Explorer?

Ans:

The REST API Explorer is a ServiceNow tool that uses information from your instance to provide a list of endpoints, methods, and variables that you can use to build and send REST requests.

- **sysparm_query** - An encoded query string used to filter the results.
- **sysparm_display_value** - Return field display values (true), actual values (false), or both (all) (default: false)
- **sysparm_exclude_reference_link** - True to exclude Table API links for reference fields (default: false)
- **sysparm_suppress_pagination_header** - True to supress pagination header (default: false)
- **sysparm_fields** - A comma-separated list of fields to return in the response
- **sysparm_limit** - 1 (Limited to 1 result for testing) The maximum number of results returned per page (default: 10,000)

- **sysparm_view Render** - the response according to the specified UI view (overridden by sysparm_fields)
- **sysparm_query_category** - Name of the query category (read replica category) to use for queries
- **sysparm_query_no_domain** - True to access data across domains if authorized (default: false)
- **sysparm_no_count** - Do not execute a select count (*) on table (default: false)

6. What is SOAP API?

Ans:

Simple Object Access Protocol (SOAP) is an XML-based protocol for accessing web services over HTTP.

Difference b/w REST and SOAP API?

- Representational state transfer (REST) is a set of architectural principles.
- Simple object access protocol (SOAP) is an official protocol maintained by the World Wide Web Consortium (W3C).
- **SOAP uses services interfaces to expose the business logic.**
- **REST uses URI to expose business logic**.
- SOAP defines standards to be strictly followed.
- REST does not define too much standards like SOAP.

7. What is Uni-Directional Integration?

Ans:

Uni-Direction Integration means only one app can able Performed Operations on another App; it's a One-way communication.

8. What is Bi-Directional Integration?

Ans:

Bi-Direction Integration means both the App can able Performed Operations on each other; basically, we can say it's a two-way communication.

9. What is LDAP and why it is used?

Ans:

Lightweight directory access protocol (LDAP) is a protocol that helps users find data about organizations, persons, and more.

Or

LDAP is the Lightweight Directory Access Protocol. It is used for user data population and User authentication. Servicenow integrates with LDAP directory to streamline the user login process and to automate the creation of user and assigning them roles

LDAP has two main goals:

- to store data in the LDAP directory and
- authenticate users to access the directory.

Steps to establish LDAP Integration:

Determine the LDAP Communication Channel. ...

Upload the X.509 Certificate. ...

Define the LDAP Server. ...

Provide LDAP server login credentials. ...

Test the Connection. ...

Define OUs within the Server. ...

Create a Data Source. ...

Select/Create a Transform Map for LDAP Data.

10. What is Import Set API and its Methods?

Ans:

The Import Set API allows you to interact with import set tables.

- **Import Set - GET /now/import/{stagingTableName}/{sys_id}**

  Retrieves the specified import staging record and resulting transformation result.

- **Import Set - POST /now/import/{stagingTableName}**

  Inserts incoming data into a specified staging table and triggers transformation based on predefined transform maps in the import set table.

11. What is Scripted REST API?

Ans:

The scripted REST API feature allows application developers to build custom web service APIs.

You can define service endpoints, query parameters, and headers for a scripted REST API, as well as scripts to manage the request and response.

Scripted REST APIs generally follow the REST architecture, but you can customize them to use different conventions. You define scripted REST APIs using the Scripted REST Service form found under **Scripted Web Services** → **Scripted REST APIs**.

Scripted REST API URIs has the following format:

https://<instance.service-now.com>/api/<name_space>/<version>/<api_id>/<relative_path>
In this URI:

- **<instance.service-now.com>:** Path to the ServiceNow instance where users access the scripted REST API.
- **<name_space>:** For web services in the global scope, the name space is the value of the property glide.appcreator.company.code. For web services in a scoped application, the name space is the scope name, such as x_company_appname. For additional information on name spaces, see Application scope.
- **<version>:** Optional. Version of the endpoint to access if the API uses versioning, such as **v1**. You can access the default version of a versioned API by specifying the URI without a version number.
- **<api_id>:** Value of the **API ID** field on the Scripted REST Service form. By default this value is based on the service name.
- **<relative_path>:** Relative path defined for the resource in the Scripted REST Service form. Specifying a relative resource path allows you to have multiple resources using the same HTTP method, such as GET, in one web service. For example, a resource may specify the path /{id} when the web service has only one GET resource, or /user/{id} and /message/{id} when the web service has different resources for requesting user and message records.

12. What is Rest Message?

Ans:

ServiceNow outbound REST functionality allows you to retrieve, create, update, or delete data on a web services server that supports the REST architecture. A REST message can be sent by a REST workflow activity or by using the RESTMessageV2 script API

13. Types of Rest Message Authentications?

Ans:

Different web service providers may require a specific type of authentication. Outbound REST supports the following authentication formats.

- Basic authentication using a username and password
- OAuth 2.0 using an OAuth provider and profile
- Mutual authentication using protocol profiles.
- No authentication.

14. What is Integration Hub and How to activate Plugin?

Ans:

Plugin – ServiceNow Integration-Hub Installer

IntegrationHub is a framework used to interact with third-party platforms in ServiceNow. Developers can use IntegrationHub to build integration actions to execute commands against external platforms.

IntegrationHub includes steps to call REST APIs, run PowerShell commands, and write scripts to interact with other APIs. Steps are the building blocks of actions used to integrate Flow Designer with external platforms. Actions are reusable building blocks to use in Flow Designer.

<u>IntegrationHub provides developers with these benefits:</u>

- Automates application logic to interact with external platforms.
- Enables SMEs to develop and share integration actions developers can leverage in custom applications.
- Provides natural-language descriptions of integration logic in an application to help non-technical users author flows and understand what flows do.

15. What is Integration Hub Spoke?

Ans:

- A spoke is a scoped application that includes Flow Designer or IntegrationHub actions or subflows.
- A spoke is a logical grouping of related actions, subflows, and supporting application files.
- Developers and Process Designers can reuse spoke logic in their own applications.

- IntegrationHub includes baseline spokes.
- Developers can obtain additional spokes from the ServiceNow Store, Share, or create their own.

16. What is Action in flow designer?

Ans:

An action is a reusable operation that enables process analysts to automate Now Platform features without having to write code.

17. What is Status code?

Ans:

200 OK	Indicates that the request has succeeded.
201 Created	Indicates that the request has succeeded and a new resource has been created as a result.
202 Accepted	Indicates that the request has been received but not completed yet. It is typically used in log running requests and batch processing.
400 Bad Request	The request could not be understood by the server due to incorrect syntax. The client SHOULD NOT repeat the request without modifications.
401 Unauthorized	Indicates that the request requires user authentication information. The client MAY repeat the request with a suitable Authorization header field
404 Not Found	The server cannot find the requested resource.
405 Method Not Allowed	The request HTTP method is known by the server but has been disabled and cannot be used for that resource.
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request.
502 Bad Gateway	The server got an invalid response while working as a gateway to get the response needed to handle the request.

504 Gateway Timeout	The server is acting as a gateway and cannot get a response in time for a request.

18. Find out the difference between Table API and Import set API, when to use what.

- The Table API allows you to perform create, read, update, and delete (CRUD) operations on existing tables.
- The API transforms incoming data based on associated transform maps. The import set API supports synchronous transforms. The Import Set API mirrors the existing SOAP interface.
- In table API we give table name and in import set API we give the staging table name.
- In table API we have **get, put, patch, post & delete** methods, In import set API we have only **get & post** method.

19. Codes for

GET one Incident:

```
(function process( /*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {

  var queryParams = request.pathParams;

  var sys_id = queryParams.sys_id;

  var gr = new GlideRecord('incident');

  gr.addQuery('sys_id', sys_id);

  gr.query();

  var body = {};


  if (gr.next()) {

    body.number = gr.number;

    body.short_description = gr.short_description;

    body.category = gr.category;

    body.description = gr.description;

  }
```

```
   response.setBody(body);
})(request, response);
```

GET all INCIDENT:

```
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
  var gr = new GlideRecord('incident');
         gr.addActiveQuery();
         gr.query();
         var bodylist = [];
         while(gr.next()){
                  var body = {};
                  body.number = gr.number.toString();
                  body.short_description = gr.short_description.toString();
                  bodylist.push(body);
         }
         response.setBody(bodylist);
})(request, response);
```

POST CODE:

```
(function process( /*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
   var requestBody = request.body; // can give (request.body.data) also here itself
   var requestData = requestBody.data;
   var inc = [];
   var gr = new GlideRecord('incident');
   gr.initialize();
   gr.short_description = requestData.short_description;
```

```
    gr.caller_id.setDisplayValue(requestData.caller_id);
    gr.category = requestData.category;
    gr.description = requestData.description;
    gr.insert();
    gr.addQuery('number', gr.number);
    gr.query();
    if (gr.next()) {
        inc.push({
            'Number': gr.number,
            'State': gr.state.getDisplayValue(),
            'Description': gr.description,
            'Short Description': gr.short_description,
            'Category': gr.category.getDisplayValue()
        });
    }
    response.setBody(inc);
})(request, response);
```

PUT CODE:

Relative path: /{sys_id}

```
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
  var sys_id = request.pathParams.sys_id;
  var res = request.body.data;
            var gr = new GlideRecord('incident');
            gr.addQuery('sys_id',sys_id);
            gr.query();
            var body = {};
```

```
        if(gr.next()){
                gr.description = res.description;
                gr.update();
                body.des = gr.description;
        }
        response.setBody(body);
})(request, response);
```

DELETE CODE:

```
var sys_id = request.pathParams.sys_id;
 var res = request.body.data;
var gr = new GlideRecord('incident');
gr.addQuery('sys_id',sys_id);
 gr.query();
var body = {};
if(gr.next()){
body.number = gr.number.toString();
body.short_description = gr.short_description.toString();
 gr.delete();
 }
 response.setBody(body);
```

20.What are Path Params?

Ans:

Path params are used to identify a specific resource within an API. They are typically used when retrieving a single resource, such as a user profile or a specific product. Path params are usually included in the URL, and they are usually the first part of the URL after the domain name.

21. What are Query Params?
Ans:
Query params are used to filter and sort the data that is returned from an API. They are typically used when retrieving multiple resources, such as a list of products or a list of users. Query params are usually included in the URL, and they are usually the last part of the URL.

22. Difference b/w Path params and Query params?

Ans:

Path params are used to identify a specific resource, while query params are used to filter and sort the data. Path params are typically used when retrieving a single resource, while query params are used when retrieving multiple resources.

23. What is Relative path and Resource path?

Ans:

- **Relative path**: Specify the unique part of the URI. The Relative path is appended to the Base API path defined in the Scripted REST API. Enclose path parameters in { }. Users pass values for path parameters in the service's URL.

- The **Resource path** field displays after the new record is saved for the first time. The Resource path is the Base API path linked with the Relative path.

24. Which port needs to open while doing LDAP integration?

Ans:

An SSL-encrypted LDAP integration (LDAPS) communicates over TCP on port 636 by default. This communication channel requires a certificate.

25.What is the purpose of using SAML 2.0 (SSO)?
Ans:
SAML 2.0 (Security Assertion Markup Language) is an open standard created to provide cross-domain single sign-on (SSO). In other words, it allows a user to authenticate in a system and gain access to another system by providing proof of their authentication

26. What is metadata in SAML 2.0?

Ans:

SAML metadata is an XML document which contains information necessary for interaction with SAML-enabled identity or service providers.

27.What is keystore in SAML 2.0?

Ans:

These are used for Security implications for signing SAML assertions, SAML protocol request and response. Certificates in SAML SSO will be used to digitally sign the SAML assertion/request/response and KeyStore is the persistent storage to store the keys/certificates.

28.What is 509 x certificate in SAML 2.0?

Ans:

The X. 509 certificates are the IdP certificates that a SAML configuration uses. After you install a certificate, you can add as many certificates as necessary. When there are multiple certificates, the system uses the first active certificate that is found

29.What is E- Bonding?

Ans:

eBonding (or e-Bonding) is a B2B software integration approach that enables the automatic exchange of data between two business applications.

30. What is email inbound processing?

Ans:

Image result for Have you worked on Email Inbound Integrations Inbound email processing allows users to interact with your application via email. Once you've set up inbound processing, the Transactional API handles receiving, processing, and parsing inbound email, then sends the parsed results to your application via a webhook.

31.How to check error handling while doing any integrations?

Ans:

The very first step in any data integration error handling is to figure out what your data integration errors are due to. The most typical errors either come from poor data connections. They could also come from bad or missing data in the source system causing validation errors in the target system