```
import io.restassured.RestAssured;
import static io.restassured.RestAssured.*;

import io.restassured.path.json.JsonPath;

public class DiscoveryCredentials {

public static void main(String[] args) {

// given - All input details
// when - Submit the API Resource, Http Methods
// Then - validate the response

RestAssured.baseURI = "https://komalkiran.service-now.com/";

// Create a New Credentials in Discovery_Credentials Table (POST Operation)

String responsevalue = given().auth().basic("admin", "Admin1234").header("Content-Type",
"application/json")
.body("{\"name\":\"TestUser123\",\"type\":\"ssh\",\"user_name\":\"TestUser123\",\"password\":\"Tes
t@1234\"}")
.log().all().when().post("api/now/table/discovery_credentials").then().log().all()
.assertThat().statusCode(201).extract().response().asString();

JsonPath js = new JsonPath(responsevalue);
String SysID = js.getString("result.sys_id");
System.out.println("Record ID is " + SysID);

// Get the User information from Discovery_Credentials Table (GET Operation)
given().auth().basic("admin", "Admin1234").log().all().when()
.get("api/now/table/discovery_credentials/" + SysID).then().log().all().assertThat().statusCode(200)
.extract().response().asString();

String Username = js.getString("result.user_name");
System.out.println("Username is " + Username);

// Updating the Username in Discovery_Credentials Table (PUT Operation) (path parameter)
responsevalue = given().auth().basic("admin", "Admin1234").header("Content-Type", "application/json")
.body("{\"user_name\":\"UpdatedUser\"}").log().all().when()
.put("api/now/table/discovery_credentials/" + SysID).then().log().all().assertThat().statusCode(200)
.extract().response().asString();

JsonPath js1 = new JsonPath(responsevalue);
String UpdatedUsername = js1.getString("result.user_name");
System.out.println("Username is " + UpdatedUsername);


// Deleting the Credentials (DELETE Operation)
given().auth().basic("admin","Admin1234").log().all()
.when().delete("api/now/table/discovery_credentials/"+SysID).then().log().all().
assertThat().statusCode(204);

System.out.println("Username is " +UpdatedUsername+ " deleted successfully");

// Checking the Credentials Exist
responsevalue =given().auth().basic("admin", "Admin1234").log().all().when()
.get("api/now/table/discovery_credentials/" + SysID).then().log().all().assertThat()
.statusCode(404).extract().response().asString();

js1= new JsonPath(responsevalue);
String Message = js1.getString("error.message");
System.out.println("Message : " + Message);

 }
}
```

2nd Way

```
import io.restassured.RestAssured;
import io.restassured.path.json.JsonPath;

import static io.restassured.RestAssured.*;
public class TestCreds {

        public static void main(String[] args) {
        RestAssured.baseURI = "https://komalkiran.service-now.com";

        String response = given().auth().basic("admin", "Admin1234")
        .pathParam("tableName", "discovery_credentials")
        .header("Content-Type", "application/json")
        .body("{\"name\":\"TestUser123\",\"type\":\"ssh\",\"user_name\":\"TestUser123\","
                        + "\"password\":\"Test@1234\"}")
        .when().post("/api/now/table/{tableName}").then().assertThat()
        .statusCode(201)
        .extract().response().asString();

        JsonPath js = new JsonPath(response);
        String SysID = js.getString("result.sys_id");
        System.out.println("Record ID is " + SysID);


        // Get the User information from Discovery_Credentials Table (GET Operation)
        given().auth().basic("admin", "Admin1234").
        pathParam("tableName", "discovery_credentials").pathParam("sys_id", SysID)
        .when()
        .get("/api/now/table/{tableName}/{sys_id}").then().assertThat()
        .statusCode(200).extract().response().asString();

        String Username = js.getString("result.user_name");
        System.out.println("Username is " + Username);

        // Updating the Username in Discovery_Credentials Table (PUT Operation)
        String responsevalue = given().auth().basic("admin", "Admin1234")
        .pathParam("tableName", "discovery_credentials").pathParam("sys_id", SysID)
        .header("Content-Type", "application/json")
        .body("{\"user_name\":\"UpdatedUser\"}").when()
        .put("/api/now/table/{tableName}/{sys_id}").then().assertThat()
        .statusCode(200).extract().response().asString();

        JsonPath js1 = new JsonPath(responsevalue);
        String UpdatedUsername = js1.getString("result.user_name");
        System.out.println("Username is " + UpdatedUsername);

        // Deleting the Credentials (DELETE Operation)
        given().auth().basic("admin","Admin1234")
        .pathParam("tableName", "discovery_credentials").pathParam("sys_id", SysID)
        .when().delete("/api/now/table/{tableName}/{sys_id}").then()
        .assertThat().statusCode(204);
        System.out.println("Credentials Deleted Successfully");

        // Checking the Credentials Exist
        responsevalue =given().auth().basic("admin", "Admin1234")
        .pathParam("tableName", "discovery_credentials").pathParam("sys_id", SysID)
        .when().get("/api/now/table/{tableName}/{sys_id}").then().assertThat()
        .statusCode(404).extract().response().asString();

        js1= new JsonPath(responsevalue);
        String Message = js1.getString("error.message");
        System.out.println("Message : " + Message);
        }

}
```

ServiceAccount JSON

```
{
  "name": "APIServiceAccount",
  "account_id": "432143214321",
  "discovery_credentials": "4956a440db96a01057a3f9afaa96194c",
  "datacenter_type": "cmdb_ci_aws_datacenter"
}
```

SSH Credentials & Service Account: Creation & Deletion of Both

```
import io.restassured.RestAssured;
import io.restassured.path.json.JsonPath;

import static io.restassured.RestAssured.*;

public class RestDiscoveryCreds {

        public static void main(String[] args) {
                // given - All the input details (logging can be added)
                // When- Submit Api repsone , http method (logging can be added)
                // Then --validate the response

                RestAssured.baseURI = "https://komalkiran.service-now.com/";

                // Create operation
                String response = given().auth().basic("admin", "Admin1234").pathParam("tableName",
"ssh_credentials")
                                .header("Content-Type", "application/json")

        .body("{\"name\":\"NewUser\",\"user_name\":\"NewUser\",\"type\":\"ssh\",\"password\":\"N
ewUser1234\"}")

        .when().post("/api/now/table/{tableName}").then().assertThat().statusCode(201).extract()
                                .asString();

                JsonPath js = new JsonPath(response);
                String Sys_id = js.getString("result.sys_id");
                System.out.println("SSH Credentials SYS_ID is " + Sys_id);

                // Get Operation
                String response1 = given().auth().basic("admin",
"Admin1234").pathParam("tableName", "ssh_credentials")
                                .pathParam("sys_id", Sys_id).header("Content-Type",
"application/json").when()

        .get("/api/now/table/{tableName}/{sys_id}").then().assertThat().statusCode(200).extract().asStri
ng();

                js = new JsonPath(response1);
                String name = js.getString("result.name");
                System.out.println("SSH Credentials Name is " + name);

                // Update Operation
                String response2 = given().auth().basic("admin",
"Admin1234").pathParam("tableName", "ssh_credentials")
```

```
                                        .pathParam("sys_id", Sys_id).header("Content-Type",
"application/json")

        .body("{\"user_name\":\"RestUser123\",\"name\":\"RestUser123\"}").when()

        .put("/api/now/table/{tableName}/{sys_id}").then().assertThat().statusCode(200).extract().asStr
ing();

                js = new JsonPath(response2);
                name = js.getString("result.name");
                System.out.println("Updated SSH Credentials Name is " + name);

                // Creation of Service Account
                String response3 = given().auth().basic("admin", "Admin1234")
                                        .pathParam("tableName",
"cmdb_ci_cloud_service_account").header("Content-Type", "application/json")

        .body("{\"name\":\"AWSServiceAccount\",\"account_id\":\"123412341234\",\"discovery_crede
ntials\":\""
                                                        + Sys_id +
"\",\"datacenter_type\":\"cmdb_ci_aws_datacenter\"}")

        .when().post("/api/now/table/{tableName}").then().assertThat().statusCode(201).extract().asStr
ing();

                js = new JsonPath(response3);
                name = js.getString("result.name");
                System.out.println("Service Account Name is " + name);
                String Sys_id1 = js.getString("result.sys_id");
                System.out.println("Service Account SYS_ID is " + Sys_id1);

        // Delete Service Account

        given().auth().basic("admin", "Admin1234")
        .pathParam("tableName", "cmdb_ci_cloud_service_account").pathParam("sys_id", Sys_id1)
        .when().delete("/api/now/table/{tableName}/{sys_id}")
        .then().assertThat().statusCode(204);

        System.out.println("Service Account deleted successfully");

        //Delete Credentials Operation
        given().auth().basic("admin", "Admin1234")
        .pathParam("tableName", "ssh_credentials").pathParam("sys_id", Sys_id)
        .when().delete("/api/now/table/{tableName}/{sys_id}")
        .then().assertThat().statusCode(204);

        System.out.println("Discovery Credential deleted successfully");
        }
    }
```

PayLoad

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;

public class Payload {

        public static String CredsDetails()
        {
                return "{\"name\":\"TestCreds12\",\"type\":\"ssh\",\"user_name\":\"TestUser13\","
                                + "\"password\":\"Test@1234\"}";
        }

        public static String CreateCredsDetails(String name,String typeName, String username, String
password)
        {
                String response = "{\"name\":\""+name+"\",\"type\":\""+typeName+"\","
                                + "\"user_name\":\""+username+"\",\"password\":\""+password+"\"}";
                return response;
        }

        public static String UpdateCredsDetails(String name,String username)
        {
                String json = "{\"name\":\""+name+"\",\"user_name\":\""+username+"\"}";
                return json;
        }

        public static String generateStringFromJson(String string) throws IOException
        {
                return new String(Files.readAllBytes(Paths.get(string)));
        }

}
```

JSON File

```
{
 "name": "NewUser1234",
 "user_name": "NewUser1234",
 "type": "ssh",
 "password": "Password@1234"
}
```

Using JSON File

```
import static io.restassured.RestAssured.given;

import java.io.IOException;
import org.testng.annotations.Test;

import files.Payload;
import io.restassured.RestAssured;
import io.restassured.path.json.JsonPath;

public class CreateCredUsingJSON {

 @Test
 public void testData() throws IOException {
        RestAssured.baseURI = "https://komalkiran.service-now.com/";
        String path = "C:\\Users\\Araj2\\eclipse-workspace\\DiscoveryProject\\jsonfiles\\creds.json";

        // Create a New Credentials in Discovery_Credentials Table (POST Operation)
        // But with body having payload json

        String responsevalue = given().auth().basic("admin", "Admin1234")
                        .header("Content-Type", "application/json")
        .body(Payload.GenerateStringFromJson(path))
        .when().post("api/now/table/discovery_credentials")
        .then().assertThat().statusCode(201).extract().response().asString();

        JsonPath js = new JsonPath(responsevalue);
        String SysID = js.getString("result.sys_id");
        System.out.println("Record ID is " + SysID);
  }

}
```

Using Request and Response Spec Builder

```
import io.restassured.authentication.PreemptiveBasicAuthScheme;
import io.restassured.builder.RequestSpecBuilder;
import io.restassured.builder.ResponseSpecBuilder;
import io.restassured.http.ContentType;
import io.restassured.path.json.JsonPath;
import io.restassured.response.Response;
import io.restassured.specification.RequestSpecification;
import io.restassured.specification.ResponseSpecification;

import static io.restassured.RestAssured.given;

public class ReqRespCreds {
        public static void main(String[] args) {
        String url = "https://komalkiran.service-now.com/";

        PreemptiveBasicAuthScheme authScheme = new PreemptiveBasicAuthScheme();
        authScheme.setUserName("admin");
        authScheme.setPassword("Admin1234");

        RequestSpecification req = new RequestSpecBuilder().setAuth(authScheme).setBaseUri(url)
        .setContentType(ContentType.JSON).build();

        ResponseSpecification resp = new ResponseSpecBuilder().expectStatusCode(201)
                    .expectContentType(ContentType.JSON).build();

        Response response = given().spec(req).pathParam("tableName", "discovery_credentials")
        .body(Payload.CreateCredsDetails("TestUser12345", "ssh", "TestUser12345",
"Password@12345"))
        .when().post("/api/now/table/{tableName}")
        .then().spec(resp).extract().response();

        String responsevalue=response.asString();

        JsonPath js = new JsonPath(responsevalue);
        String SysID = js.getString("result.sys_id");
        System.out.println("Record ID is " + SysID);

        String Username = js.getString("result.user_name");
        System.out.println("Username is " + Username);

  }
}
```

Create a UTILS folder and Add Request and Reponse Spec Builder

```
package utils;

import io.restassured.authentication.PreemptiveBasicAuthScheme;
import io.restassured.builder.RequestSpecBuilder;
import io.restassured.builder.ResponseSpecBuilder;
import io.restassured.http.ContentType;
import io.restassured.specification.RequestSpecification;
import io.restassured.specification.ResponseSpecification;

public class ReqRespBuild {

        static String url = "https://komalkiran.service-now.com/";

        public static RequestSpecification reqBuilder(){
                PreemptiveBasicAuthScheme authScheme = new PreemptiveBasicAuthScheme();
                authScheme.setUserName("admin");
                authScheme.setPassword("Admin1234");
                RequestSpecification req = new RequestSpecBuilder().setAuth(authScheme)
                .setBaseUri(url).setContentType(ContentType.JSON).build();
                return req;
        }

        public static ResponseSpecification respBuilder(int statusCode) {
                ResponseSpecification resp = new
ResponseSpecBuilder().expectStatusCode(statusCode)
                                        .expectContentType(ContentType.JSON).build();
                return resp;
        }

}
```

Create a New Class and Call the above method

```
import static io.restassured.RestAssured.given;

import org.testng.annotations.Test;

import io.restassured.path.json.JsonPath;
import io.restassured.response.Response;
import utils.ReqRespBuild;

public class NewCredsTest {

Response response;
static String SysID;
JsonPath js;
String responsevalue;
String Username;

 @Test(priority=1,description="Creation of a New Credentials")
 public void createCreds(){
 response = given().spec(ReqRespBuild.reqBuilder()).pathParam("tableName", "discovery_credentials")
        .body(Payload.CreateCredsDetails("TestUser12345", "ssh", "TestUser12345",
"Password@12345"))
        .when().post("/api/now/table/{tableName}")
        .then().spec(ReqRespBuild.respBuilder(201)).extract().response();

        responsevalue=response.asString();

        js = new JsonPath(responsevalue);
        SysID = js.getString("result.sys_id");
        System.out.println("Record ID is " + SysID);
```

```
            Username = js.getString("result.user_name");
            System.out.println("Username is " + Username);
    }

    @Test(priority=2,description="Updation of a New Credentials")
    public void updateCreds(){
    response = given().spec(ReqRespBuild.reqBuilder()).pathParam("tableName", "discovery_credentials")
                    .pathParam("sys_id", SysID)
                    .body(Payload.UpdateCredsDetails("APIUser12345", "APIUser12345"))
                    .when().put("/api/now/table/{tableName}/{sys_id}")
                    .then().spec(ReqRespBuild.respBuilder(200)).extract().response();

            responsevalue=response.asString();

            js = new JsonPath(responsevalue);
            Username = js.getString("result.user_name");
            System.out.println("Updated Username is " + Username);

    }
}
```

Output:

```
[RemoteTestNG] detected TestNG version 6.14.2
[TestNGContentHandler] [WARN] It is strongly recommended to add "<!DOCTYPE suite SYSTEM
"http://testng.org/testng-1.0.dtd" >" at the top of your file, otherwise TestNG may fail or not work as
expected.


Record ID is 4956a440db96a01057a3f9afaa96194c
Username is TestUser12345
Updated Username is APIUser12345
PASSED: createCreds
        Creation of a New Credentials
PASSED: updateCreds
        Updation of a New Credentials

===============================================
    Default test
    Tests run: 2, Failures: 0, Skips: 0
===============================================
```

Logging Options

```
PrintStream log= new PrintStream(new FileOutputStream("loggint.txt"));

RequestSpecification req = new RequestSpecBuilder().setAuth(authScheme)
.setBaseUri(url).addFilter(RequestLoggingFilter.logRequestTo(log))
.addFilter(ResponseLoggingFilter.logResponseTo(log))
.setContentType(ContentType.JSON).build();
```

Run the test. Refresh the project to see the logging.txt file. Inside the file we will be seeing the logs of both request and response.

Reusable Method with Logging Info

```
package utils;

import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintStream;

import io.restassured.authentication.PreemptiveBasicAuthScheme;
import io.restassured.builder.RequestSpecBuilder;
import io.restassured.builder.ResponseSpecBuilder;
import io.restassured.filter.log.RequestLoggingFilter;
import io.restassured.filter.log.ResponseLoggingFilter;
import io.restassured.http.ContentType;
import io.restassured.specification.RequestSpecification;
import io.restassured.specification.ResponseSpecification;

public class ReqRespBuild {

        static String url = "https://komalkiran.service-now.com/";

        public static RequestSpecification reqBuilder() throws FileNotFoundException{
                PreemptiveBasicAuthScheme authScheme = new PreemptiveBasicAuthScheme();
                authScheme.setUserName("admin");
                authScheme.setPassword("Admin1234");

                PrintStream log = new PrintStream(new FileOutputStream("logging.txt"));
                RequestSpecification req = new RequestSpecBuilder().setAuth(authScheme)
                .setBaseUri(url).addFilter(RequestLoggingFilter.logRequestTo(log))
                .addFilter(ResponseLoggingFilter.logResponseTo(log))
                .setContentType(ContentType.JSON).build();
                return req;
        }
public static ResponseSpecification respBuilder(int statusCode) {
                    ResponseSpecification resp = new
ResponseSpecBuilder().expectStatusCode(statusCode)
                                                  .expectContentType(ContentType.JSON).build();
                    return resp;
          }

}
```

Cucumber Framework

Create a MAVEN Framework. Add Dependencies details to pom.xml file

```
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.11</version>
  <scope>test</scope>
</dependency>
<dependency>
        <groupId>io.cucumber</groupId>
        <artifactId>cucumber-java</artifactId>
        <version>6.7.0</version>
</dependency>
<dependency>
        <groupId>io.cucumber</groupId>
        <artifactId>cucumber-junit</artifactId>
        <version>6.7.0</version>
        <scope>test</scope>
</dependency>
<dependency>
        <groupId>io.rest-assured</groupId>
        <artifactId>rest-assured</artifactId>
        <version>4.3.3</version>
        <scope>test</scope>
</dependency>
<dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-databind</artifactId>
        <version>2.10.2</version>
</dependency>
```

Install natural plugins from Eclipse Help→Eclipse MarketPlace→Search→Natural 0.9 Plugin→Install

Create 4 Folder under the java project

1. discovery.Options (should contains only TestRunner file)
2. features (should contains only feature file)
3. stepDefinations (should contains only stepDefination file)
4. utils (should contain only reusable file)

Navigate to feature package→Right click on the package→New→file→discovery.feature (add the below content)

```
Feature: Create Credentials

Scenario: Create Discovery Creds
```

Given Create Discovery payload
When user calls CreateAPI with Post http request
Then the API call got success with status code 201
And "user_name" in value reponse body is "TestUser12345"

Navigate to discovery.options package

Create a TestRunner.java file and add the below contents

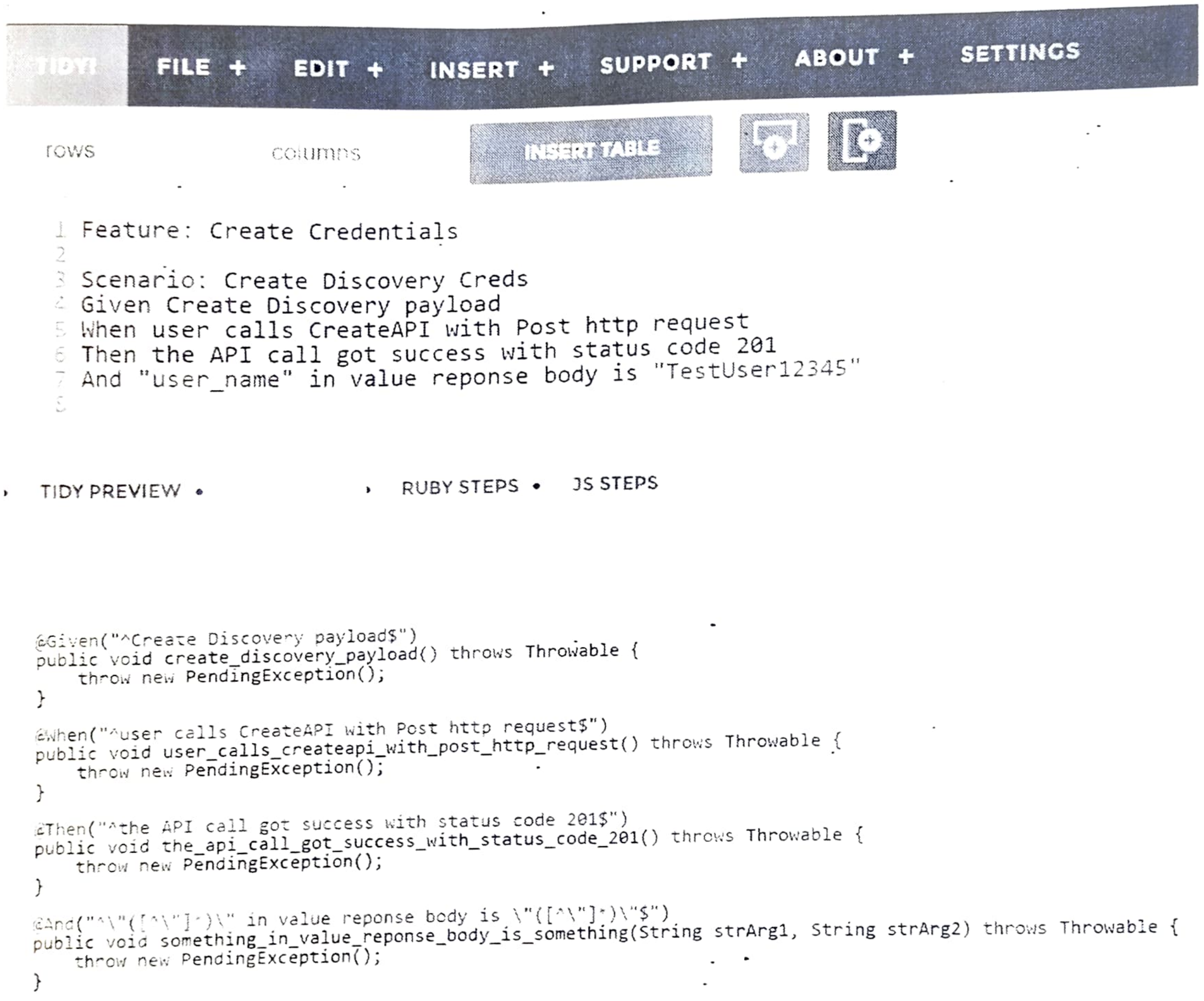
```
package discovery.Options;

import org.junit.runner.RunWith;

import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;

@RunWith(Cucumber.class)
@CucumberOptions(features="src/test/java/features",glue= {"stepDefinations"},tags=("@Create"))
public class TestRunner {

}
```

Go to utils package: Create payload with below content

```
package utils;

import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;

public class Payload {

	public static String CredsDetails()
	{
		return "{\"name\":\"TestCreds12\",\"type\":\"ssh\",\"user_name\":\"TestUser13\","
				+ "\"password\":\"Test@1234\"}";
	}

	public static String CreateCredsDetails(String name,String typeName, String username, String
password)
	{
		String response = "{\"name\":\""+name+"\",\"type\":\""+typeName+"\","
				+ "\"user_name\":\""+username+"\",\"password\":\""+password+"\"}";
		return response;
	}
}
```

Open the Chrome Browser→Search for tidy gherkin plugin→Install the plugin→click on Launch app

Copy paste the Feature, Scenario and details in the tidy gherkins, as below



```
Feature: Create Credentials

Scenario: Create Discovery Creds
Given Create Discovery payload
When user calls CreateAPI with Post http request
Then the API call got success with status code 201
And "user_name" in value reponse body is "TestUser12345"
```

```
@Given("^Create Discovery payload$")
public void create_discovery_payload() throws Throwable {
    throw new PendingException();
}

@When("^user calls CreateAPI with Post http request$")
public void user_calls_createapi_with_post_http_request() throws Throwable {
    throw new PendingException();
}

@Then("^the API call got success with status code 201$")
public void the_api_call_got_success_with_status_code_201() throws Throwable {
    throw new PendingException();
}

@And("^\"([^\"]*)\" in value reponse body is \"([^\"]*)\"$")
public void something_in_value_reponse_body_is_something(String strArg1, String strArg2) throws Throwable {
    throw new PendingException();
}
```

Then Click on Java Steps→ Copy the method content as below and paste in the stepDefinations file.

```
@Given("^Create Discovery payload$")
public void create_discovery_payload() throws Throwable {
   throw new PendingException();
}

@When("^user calls \"([^\"]*)\" with Post http request$")
public void user_calls_something_with_post_http_request(String strArg1) throws Throwable {
   throw new PendingException();
}

@Then("^the API call got success with status code 201$")
public void the_api_call_got_success_with_status_code_201() throws Throwable {
   throw new PendingException();
}
```

```
    @And("^\"([^\"]*)\" in value reponse body is \"([^\"]*)\"$")
    public void something_in_value_reponse_body_is_something(String strArg1, String strArg2) throws
Throwable {
        throw new PendingException();
    }
```

Remove the **throw new** PendingException() inside the method.

Now Your java file looks as below.

```
package stepDefinations;

import io.cucumber.java.en.And;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;

public class MyStepDefinitions {

    @Given("^Create Discovery payload$")
    public void create_discovery_payload() throws Throwable {

    }

    @When("^user calls \"([^\"]*)\" with Post http request$")
    public void user_calls_something_with_post_http_request(String strArg1) throws Throwable {

    }

    @Then("^the API call got success with status code 201$")
    public void the_api_call_got_success_with_status_code_201() throws Throwable {

    }

    @And("^\"([^\"]*)\" in value reponse body is \"([^\"]*)\"$")
    public void something_in_value_reponse_body_is_something(String strArg1, String strArg2) throws
Throwable {

    }
}
```

Now Start Adding Your implementation inside the @Given, @When, @Then and @And Annotations

Finally, Your stepDefinations Look as below.

```
package stepDefinations;

import static io.restassured.RestAssured.given;

import org.junit.Assert;

import demo.Payload;
import io.cucumber.java.en.And;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;
import io.restassured.authentication.PreemptiveBasicAuthScheme;
import io.restassured.builder.RequestSpecBuilder;
import io.restassured.builder.ResponseSpecBuilder;
import io.restassured.http.ContentType;
import io.restassured.path.json.JsonPath;
import io.restassured.response.Response;
import io.restassured.specification.RequestSpecification;
import io.restassured.specification.ResponseSpecification;

public class MyStepDefinitions {

        String url = "https://komalkiran.service-now.com/";
        static RequestSpecification req;
        static ResponseSpecification resspec;
        static String SysID;
        static Response response;

  @Given("^Create Discovery payload$")
  public void create_discovery_payload() throws Throwable {
          PreemptiveBasicAuthScheme authScheme = new PreemptiveBasicAuthScheme();
          authScheme.setUserName("admin");
          authScheme.setPassword("Admin1234");

                  req = new RequestSpecBuilder().setAuth(authScheme)
                                  .setBaseUri(url).setContentType(ContentType.JSON).build();
  }

  @When("^user calls \"([^\"]*)\" with Post http request$")
  public void user_calls_something_with_post_http_request(String strArg1) throws Throwable {
                resspec = new ResponseSpecBuilder().expectStatusCode(201)
                                .expectContentType(ContentType.JSON).build();

        response = given().spec(req)
                        .body(Payload.CreateCredsDetails("TestUser12345", "ssh", "TestUser12345",
"Password@12345"))
```

```
                        .when().post("api/now/table/discovery_credentials").then().spec(resspec)
                        .extract().response();

    }

    @Then("^the API call got success with status code 201$")
    public void the_api_call_got_success_with_status_code (Integer int1) throws Throwable {
        Assert.assertEquals(response.getStatusCode(), int1.intValue());
    }

    @And("^\"([^\"]*)\" in value reponse body is \"([^\"]*)\"$")
    public void something_in_value_reponse_body_is_something(String string, String string2) throws
Throwable {
                String responsevalue=response.asString();

                JsonPath js = new JsonPath(responsevalue);
                SysID = js.getString("result.sys_id");
                System.out.println("Record ID is " + SysID);
                String actualValue = js.getString("result."+string+"");
                System.out.println("User name is " + actualValue);
                Assert.assertEquals(actualValue, string2);

    }
}
```

Now go to discovery.options→TestRunner.java→right click→run as→junit

Output:

```
Record ID is 011ccf10db5ae01057a3f9afaa9619c5
User name is TestUser12345
```