# Testing different scenarios for the Serverless AWS EC2 Actions API

This document consolidates **all test case scenarios** for the **Serverless AWS EC2 Actions API**, ensuring that it validates across both user types.

* Internal Users - Authenticated via IAM-based authorization
* External Users - Authenticated via Lambda Authorizer

The testing is implemented across **two Python test files**, which covers different scenarios.

**Testcase scenarios for internal Users (IAM-based authentication):**

Functionality of the **test\_api\_iam\_based.py** file:

* This function is written in Python and using pytest module.
* It has 4 different parameterized functions which validates the create, start, stop and terminate action of the api functionality.
* The api endpoint is:

<https://nx0gn3q6e2.execute-api.us-east-1.amazonaws.com/test/actions>

and it is invoked using the query String Parameter.

* The api endpoint is accessible only if correct access key and secret key is given.
* To create and ec2 instance, we invoke the api endpoint with the below query String Parameter –

{“action”: “create”}

On successful execution of the API, an ec2 instance will get created and the api returns the details needed to connect the ec2 instance via SSH.

* For performing other operations – starting an ec2 instance, stopping an ec2 instance and terminating an ec2 instance, the below query String Parameter is used while invoking the api.

1. Start - {"action": "start", "instance\_id":"i-011e2bcbbb1130be1"}
2. Stop - {"action": "stop", "instance\_id":"i-011e2bcbbb1130cg5"}
3. Terminate - {"action": "terminate", "instance\_id":"i-011e25cd40778bbct"}

* To validate all these different actions, we have 4 functions in the testing code.

1. test\_api\_create\_action
2. test\_api\_start\_action
3. test\_api\_stop\_action
4. test\_api\_terminate\_action

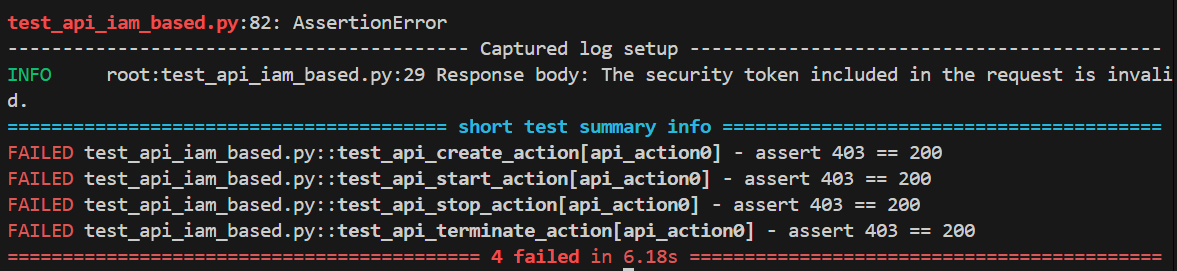
The below testcases are validated in the **test\_api\_iam\_based.py** file along with screenshots attached to it in the result.

NOTE – To test a specific api action, make sure to mention @pytest.mark.skip for other functions to skip the testcases.

Command used in terminal - pytest -s --log-cli-level=INFO test\_api\_iam\_based.py

**Scenario -1:** **Incorrect Access key and Secret Key Provided**

Result: This will give **403 error** and will raise **Assertion Error**.



**Scenario-2:**

**Valid Access Key and Secret Key provided and testing for create action**

Result:

This will successfully create an instance and returns the ec2 connection details.

A screen shot of a computer program

AI-generated content may be incorrect.

The below screenshot represents the ec2 instance created in aws environment.

A screenshot of a chat

AI-generated content may be incorrect.

**Scenario – 3: Valid Access Key and Secret Key provided and testing for start action with no instance id in parameter**

Result:

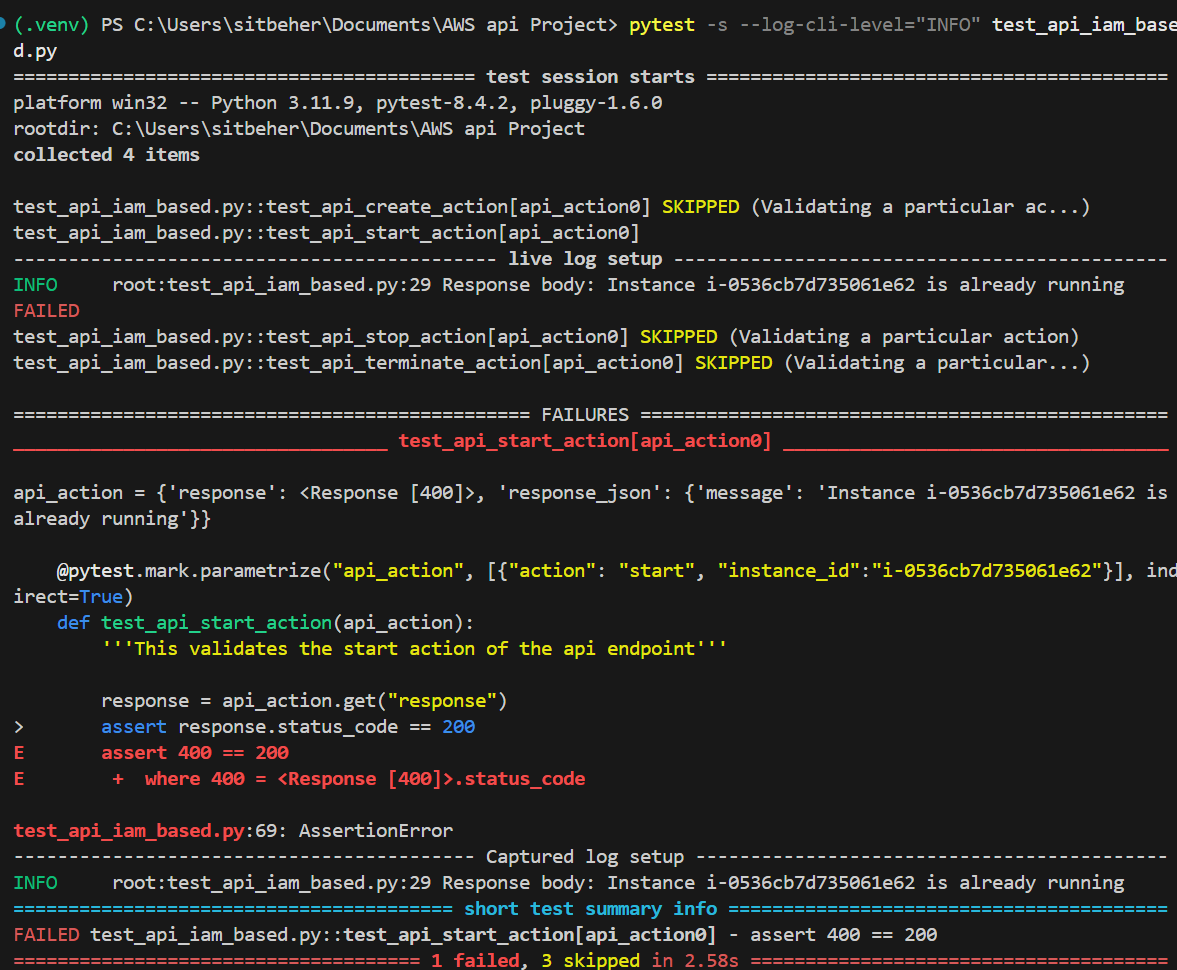
This will give Assertion Error with detailed message for the error.

A screenshot of a computer program

AI-generated content may be incorrect.

**Scenario – 4: Valid Access Key and Secret Key provided and testing for start action with correct parameters, instance is in “running” state**

Result: This will give Assertion Error with detailed message for the error.



**Scenario – 5: Valid Access Key and Secret Key provided and testing for start action with correct parameters, instance is in “stopped” state**

Result: This will successfully start the ec2 instance mentioned in the parameter.

A screen shot of a computer program

AI-generated content may be incorrect.

The below screenshot represents the mentioned ec2 instance in “running” state.

A screenshot of a chat

AI-generated content may be incorrect.

**Scenario – 5: Valid Access Key and Secret Key provided and testing for stop action with correct parameters**

Result:

This will successfully stop the ec2 instance.

**A screen shot of a computer program

AI-generated content may be incorrect.**

The below screenshot represents the mentioned ec2 instance in “stopped” state.

A screenshot of a computer

AI-generated content may be incorrect.

**Scenario – 6: Valid Access Key and Secret Key provided and testing for stop action with no instance id in parameter**

Result: This will Assertion Error with detailed message.

A computer screen shot of a program

AI-generated content may be incorrect.

**Scenario – 7:**

**Valid Access Key and Secret Key provided and testing for terminate action with no instance id in parameter**

Result:

This will raise Assertion Error with “missing instance id” message.

A screen shot of a computer program

AI-generated content may be incorrect.

**Scenario – 8:**

**Valid Access Key and Secret Key provided and testing for terminate action with correct parameters**

Result:

This will successfully terminate the ec2 instance.

A screen shot of a computer program

AI-generated content may be incorrect.

This screenshot represents the mentioned ec2 instance is in “terminated” state.

A screenshot of a computer

AI-generated content may be incorrect.

**Scenario – 9:**

**Valid Access Key and Secret Key provided and testing for terminate action with correct parameters and instance is already terminated state**

Result:

This will raise Assertion Error as the instance is already terminated.

A computer screen with text and numbers

AI-generated content may be incorrect.

**Testcase scenarios for external Users (Lambda Authorizer based authentication):**

Functionality of the **test\_api\_authorizer\_based.py** file:

* This function is written in Python and using pytest module.
* The functionality is similar to the previous testing file, but here the authentication is done using Lambda Authorizer, which validates custom tokens or credentials before granting access.
* The api endpoint is:

<https://nx0gn3q6e2.execute-api.us-east-1.amazonaws.com/test/events>

The api endpoint is accessible only if correct token is given.

**Scenario -1:**

**No token Provided to call the api**

Result:

This will raise Assertion error with message as “unauthorized”.

A computer screen shot of a program

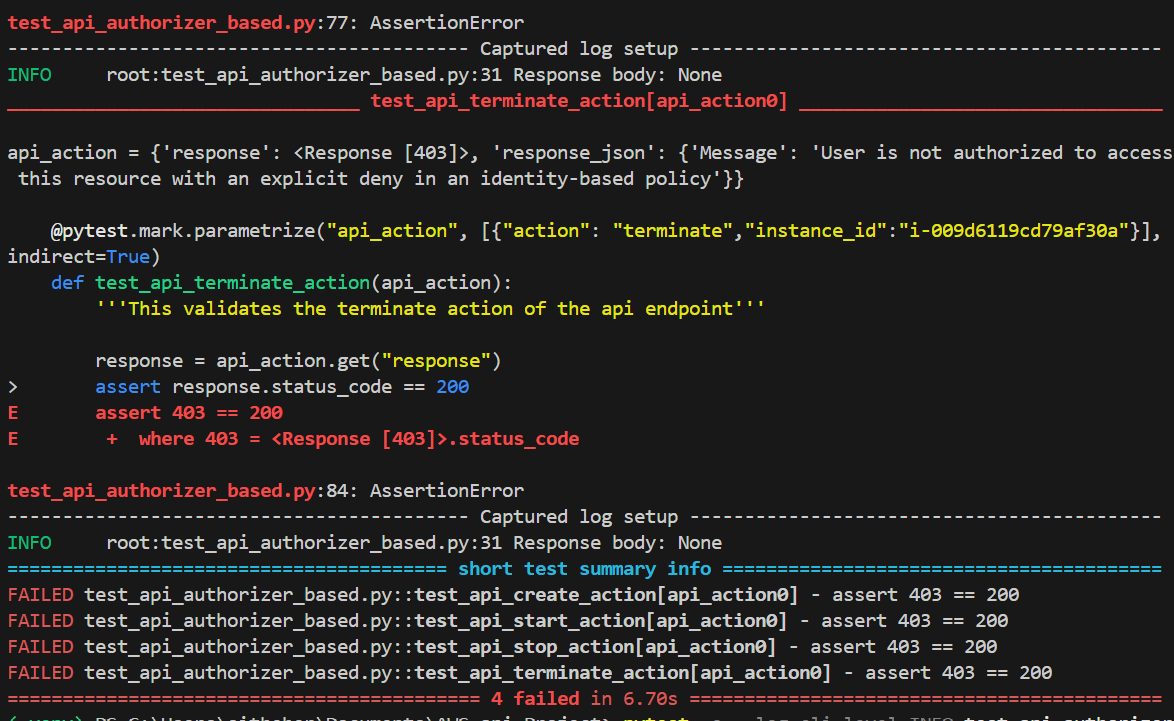
AI-generated content may be incorrect.

**Scenario -2:**

**Invalid token Provided to call the api**

Result:

This will raise Assertion error with “User is not authorized to access the resource” as the Lambda logic will return an explicit deny accessing the api endpoint.



**Scenario -3:**

**Valid token Provided to call the api**

Result:

* This will successfully run all the testcases and we can see all the testcases are successful.
* While testing this scenario, no testcases are skipped to check a proper flow of all the actions that is covered using the api.
* The scenarios for the particular action items are similar to the previous testing done using the IAM-based authentication.
* Here we should make sure that the valid token is used to call the api for other actions to successfully test all the details.

Here is the screenshot of all the actions performed by the api for all the testcases.

