

## **Boto3 guide and Application**

What is Boto3?

Boto3 is the Amazon Web Services (AWS) SDK for Python. It allows developers to write software that makes use of Amazon services like S3 (Simple Storage Service), EC2 (Elastic Compute Cloud), DynamoDB and Lambda.

What boto3 does?

It provides Pythonic interfaces to AWS services so you can:

- Create, configure, and manage AWS services
- Automate infrastructure tasks using code (IaC principles)

Boto3 call types:

Boto3 offers two main types of calls to interact with AWS:

### 1. Client

- Low-level service access.
- Maps 1:1 with AWS API operations.
- Returns raw responses (dictionaries).
- Suitable when you want full control and direct access to the AWS API.

Example for that:

```
import boto3

s3_client = boto3.client('s3')

response = s3_client.list_buckets()

print(response['Buckets'])
```

## 2. Resource

- High-level object-oriented interface.
- Automatically handles pagination and object relationships.
- More Pythonic and convenient for many use cases.

Example for that :

```
import boto3

s3_resource = boto3.resource('s3')

for bucket in s3_resource.buckets.all():

    print(bucket.name)
```

### what is a Reservation?

- A **reservation** is just a **wrapper or group** that contains one or more EC2 instances.
- It's how AWS organizes the data internally before giving it back to you.
- Even if you **only have one instance**, AWS will **still wrap it in a reservation**.

#### #Let's explain "Reservations" in the simplest possible way







Let's say you ask AWS:

"Hey, give me a list of all my EC2 instances!"

Now, AWS says:

"Sure! But I'm going to return them grouped in **containers called reservations**."

Imagine AWS is giving you oranges (your EC2 instances), but they **always pack them in boxes** (reservations). So:

- You have:
  -  Box 1 →  EC2 #1,  EC2 #2
  -  Box 2 →  EC2 #3
  -  Box 3 → (empty)

If you want to reach **all oranges**, you have to:

1. Open each box (loop through reservations)
2. Take out the oranges (loop through instances inside each reservation)

That is it 😊

## Task & Results

ForgTech company wanna test your ability to Deliver their requirements utilizing Python, This will help you to build a good reputation.

The FrogTech Cloud Team requests you implement a local Python script with AWS Boto3 SDK that fetch the details below:

1. All VPC details in us-east-1 region.
2. All EC2 details in us-east-1 region.
3. Save all details using JSON Or Text format.

What I have done:

- installed boto3 and json library through this command via cmd
  - pip install boto3
  - pip install json
- configure AWS Credentials through "aws configure"
- create a python file then write this code:

```
# importing libraries
```

```
import boto3
```

```
import json
```

```
#Initialize the session call
```

```
session = boto3.Session(region_name="us-east-1")
```

```
# Initialize the client call
```

```
ec2_client = session.client("ec2")
```

```
# Fetch VPC details
```

```
#describe_vpc() it list all available VPCs in the region specified
```

```
vpc_response = ec2_client.describe_vpcs()
```

```
#it get all VPCs available through .get
```

```
vpcs = vpc_response.get("Vpcs", [])
```

```
# Fetch EC2 details
```

```
ec2_response = ec2_client.describe_instances()
```

```
instances = [ ]
```

```
for reservation in ec2_response["Reservations"]:
    for instance in reservation["Instances"]:
        instances.append(instance)
```

```
# Create a dictionary to store results
```

- #organize your data into a dictionary with two keys: VPCs and EC2\_Instances.

```
data = {
    "VPCs": vpcs,
    "EC2_Instances": instances
}
```

```
# Save data as JSON
```

```
#Opens a file for writing.
```

```
with open("aws_details.json", "w") as json_file:
```

```
#Writes the Python dictionary into the file in pretty-printed JSON format.
```

```
# It uses 4 spaces to indent nested structures.
```

```
    json.dump(data, json_file, indent=4)
```

```
print("AWS details have been saved to aws_details.json")
```