Developing Storage Solutions with Amazon Simple Storage Service (S3)

(LAB-M06-01)

Version Control	
Document	Developing Storage Solutions with Amazon S3
Owner	Ahmad Majeed Zahoory
Version	3.1
Last Change	22nd May 2024
Description of Change	Task steps updated

Lab duration: **60 minutes**

Lab scenario

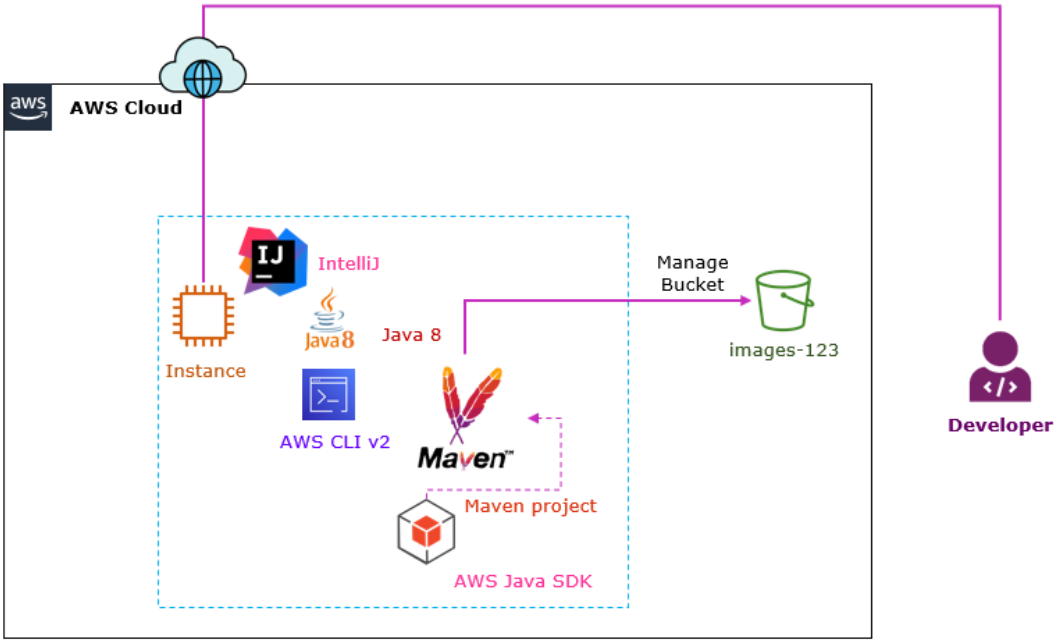
You're preparing to store binary data in AWS. As a development group, your team has decided to use Java to manage the data from AWS storage programmatically.

Objectives

After you complete this lab, you will be able to:

- Create a bucket.
- Delete a bucket.
- List the existing buckets.
- List the bucket contents.
- Upload the objects in bucket.
- Get the objects from bucket.
- Upload the objects with metadata in bucket.
- Update metadata for the existing objects.
- Create the Pre-signed URL.
- Get the objects using Pre-signed URL.

Ahmad Majeed Zahoory



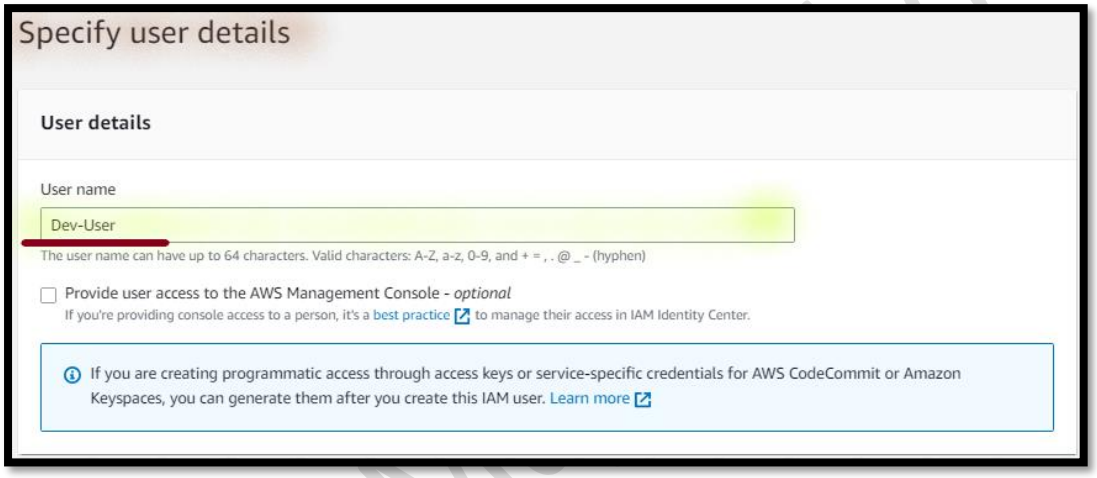
© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Task 1: Create IAM User

In this task, you will create AWS IAM User with AWS S3 permission.

Step 1: Create IAM User

1. In the **AWS Management Console**, on the **Services** menu, click **IAM**.
2. Select **Users**.
   a. Select **Add users**.
      i. In the **Specify user details** section:
         a) **Username**: Write **Dev-User-YOUR-ID**.

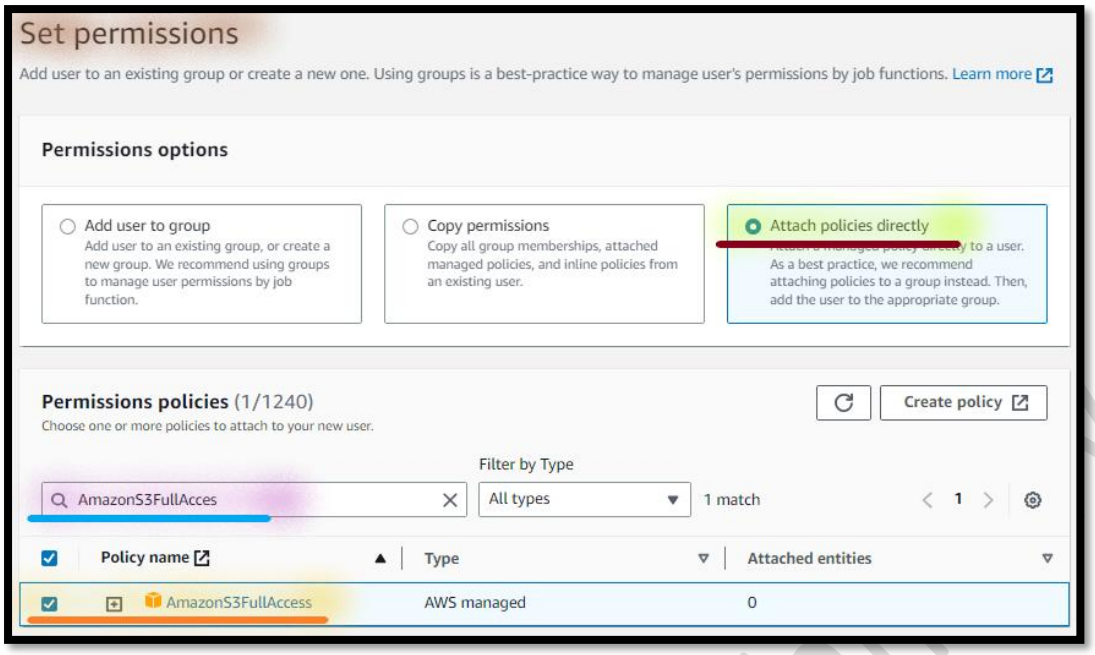


Note: Leave the other details as default.

         b) Select **Next**.

      ii. In the **Set permissions** section:
         a) Select **Attach policies directly**.
            1) In the **Search box**, write **AmazonS3FullAcces** and select **Enter Key**.
            2) Select **AmazonS3FullAcces**.

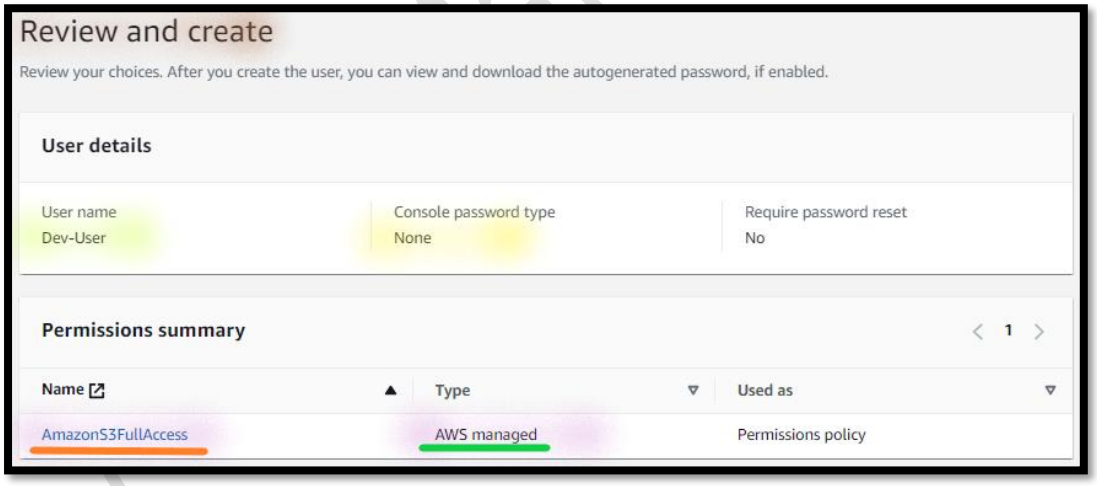
© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.



3) Select **Next**.

iii. In the **Review and create** section.

Note: In the **Permission summary**, you can see the **AmazonS3FullAccess** policy.



a) Select **Create user**.

Note: You can see the **message** "**User created succesfully**".

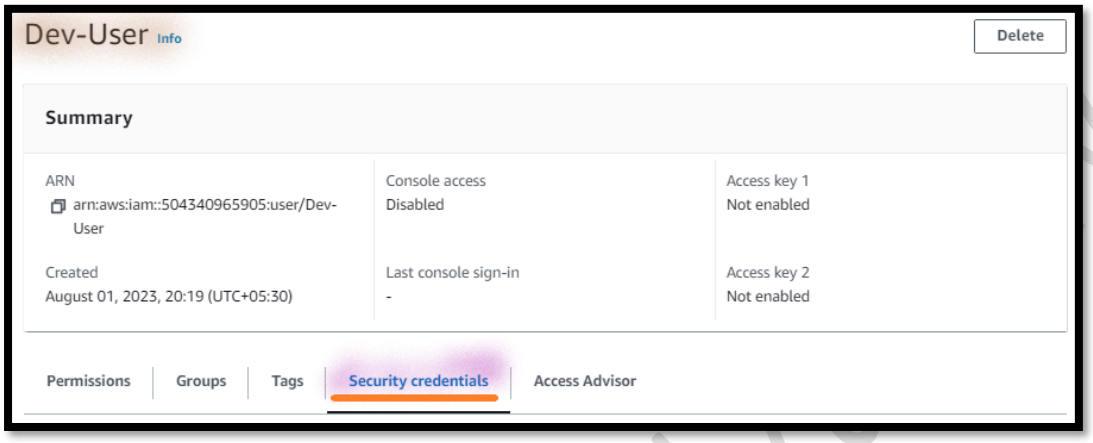
© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Step 2: Create IAM User Access Keys

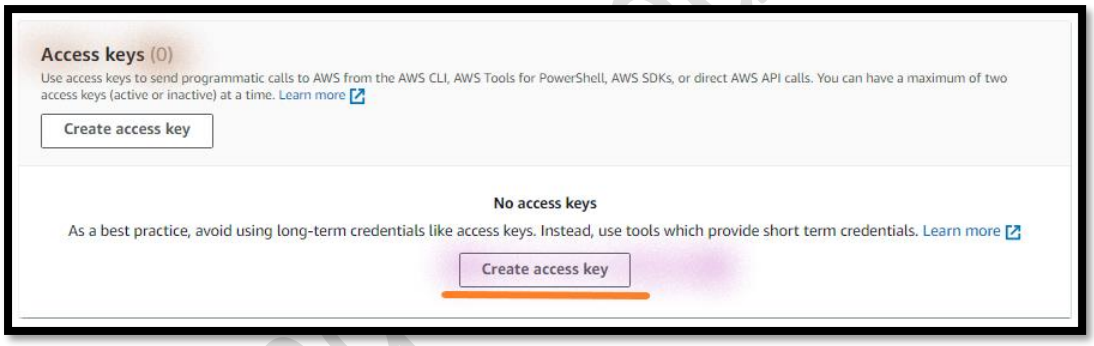
3. **From** the **IAM** console.

   a. Open the **Dev-User-YOUR-ID**.

      i. Select **Security credentials**.



a) **Access keys**: Select **Create access key**.

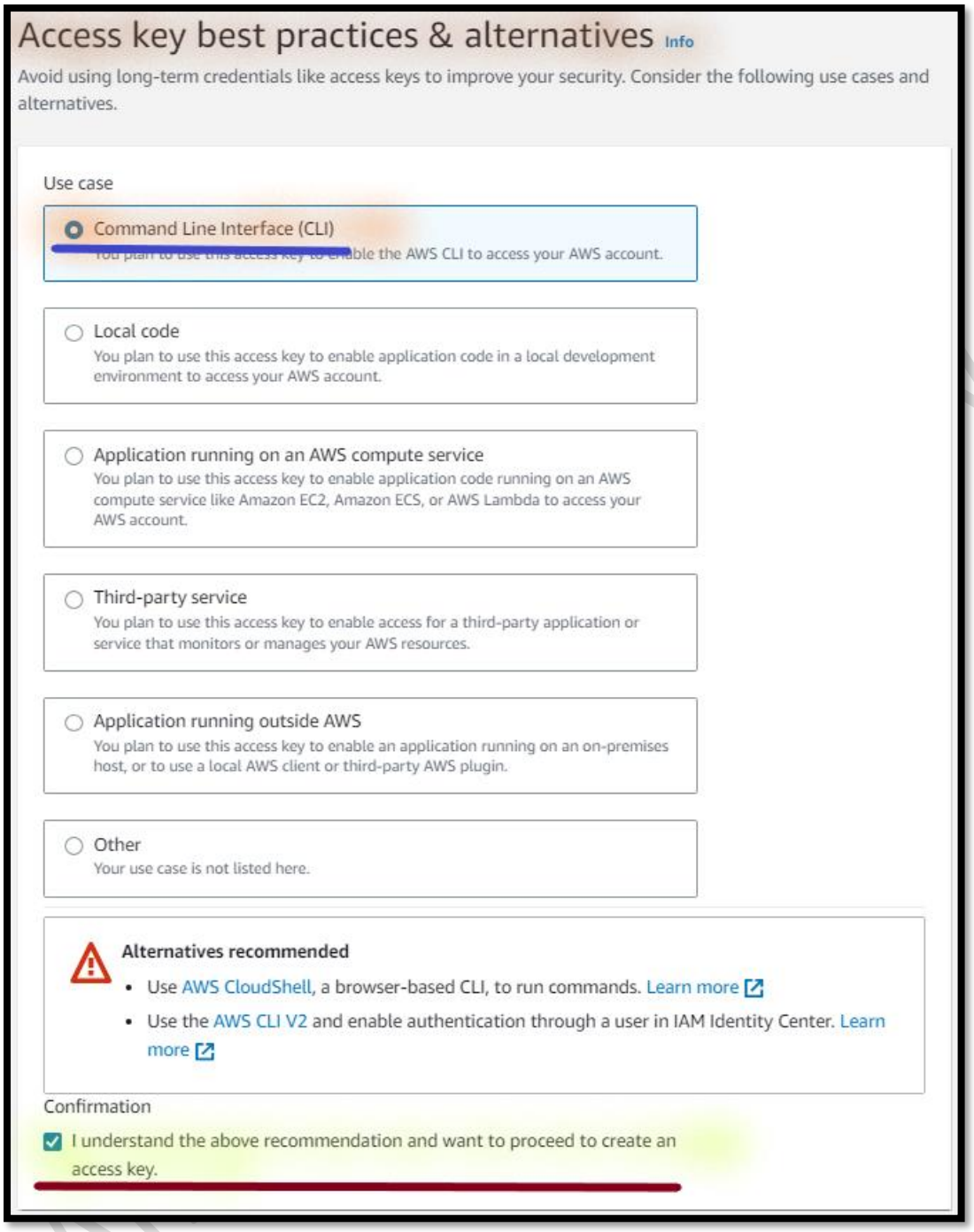


I. In the **Access key best practices & alternatives** section:

   A. **Use case**: Select **Command Line Interface (CLI)**.

   B. **I understand the above recommendation and want to proceed to create an access key**: **Enable** the **Checkmark**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

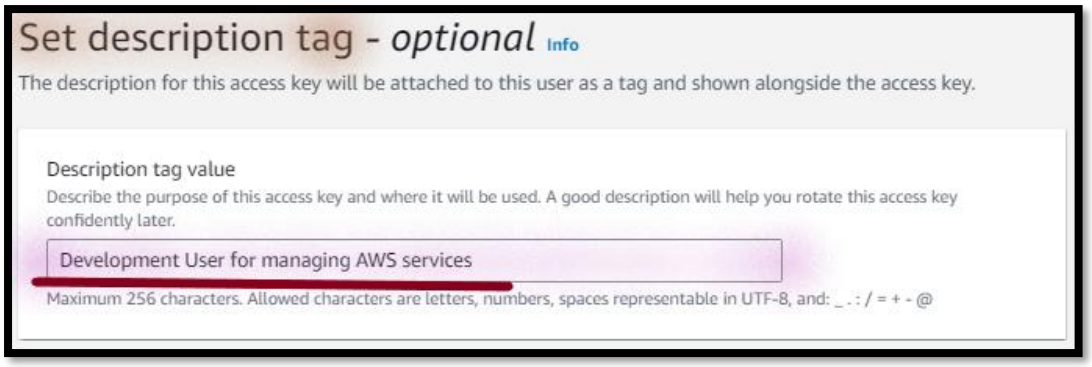


C. Select **Next**.

II. In the **Set description tag** section:

A. **Description tag value**: Write **Development User for managing AWS services**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

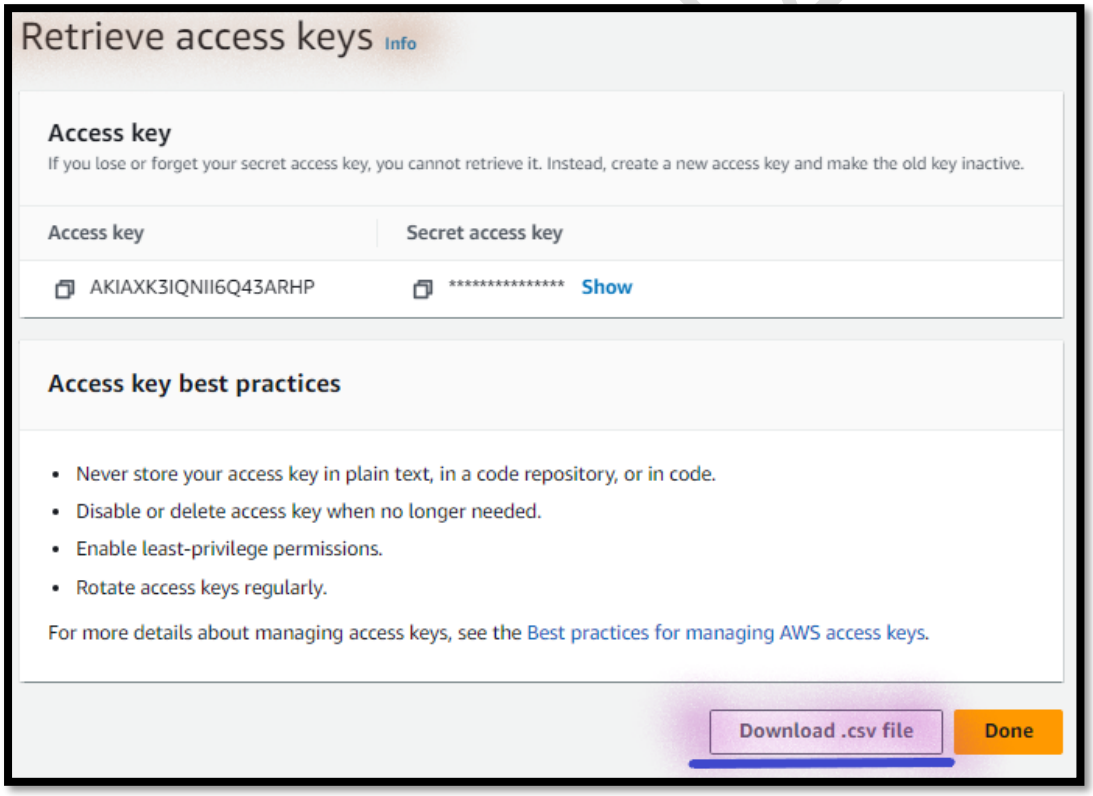


B. Select **Create access key**.

Note: You can see the **message** "**Access key created**".

III. In the **Retrieve access keys** section:

A. Select **Download .csv file**.



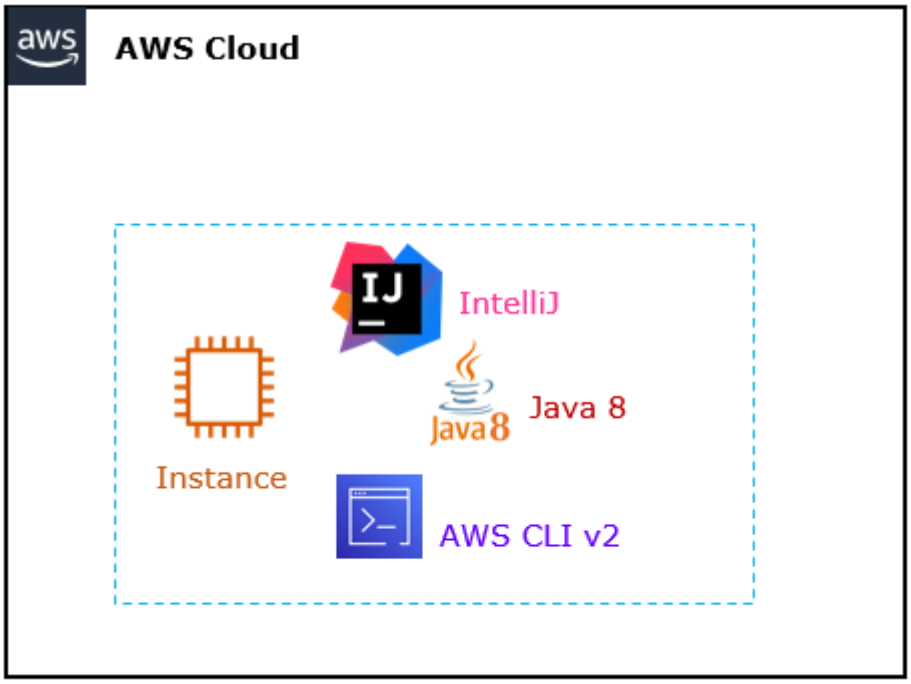
Note: **Access key ID** and **Secret access key** details of the **Dev-User-YOUR-ID** gets downloaded in your **local desktop/ laptop**.

B. Select **Done**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Task 2: Build Development Environment

In this task, you will build the AWS EC2 instance to build development environment and install Java, IntelliJ and AWS CLI using the cloud formation template.



Step 1: Create EC2 Instances

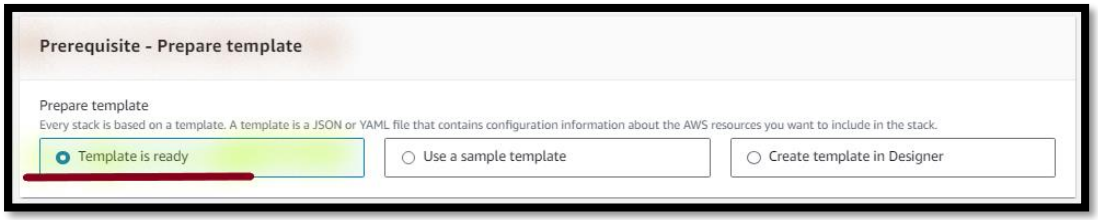
4. In the **AWS Management Console**, on the **Services** menu, search and select **CloudFormation**.

5. Choose the **YOUR ALLOCATED REGION**, region list to the right of your account information on the navigation bar.

6. Select **Create stack** and configure:

   a. In the **Create stack** page:

      i. **Prepare template**: Select **Template is ready**.



© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

ii. **Template source**: Select **Upload a template file**.

iii. **Choose file**: Click on **Choose file**.

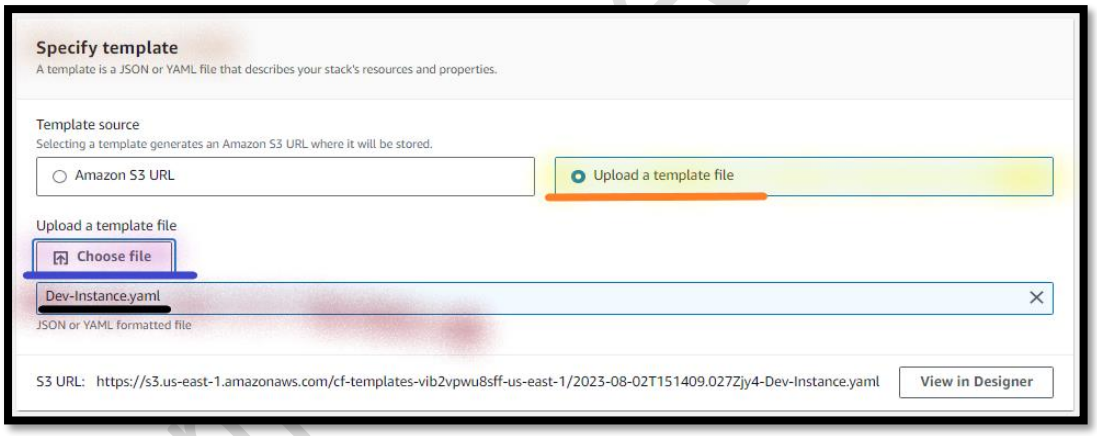
a) **Navigate** and **select** the **Dev-Instance**.yaml file.

Note: **Dev-Instance**.yaml template is provided with the Lab manual.

Note: AWS template **performing** the **following** tasks:

1. **Creating Windows instances**.
2. **Creating t2.medium** instance (**2 vCPU** and **4 GB memory**) [*This instance type attract charges*].
3. **Set** the **Administrator's Password**.

Note: You can also use **t2.micro**, but the **performance will be low** to build development environment.



iv. Select **Next**.

b. In the **Specify stack details** page:

i. **Stack name**: Write **Dev-Instance-JV**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Note: Leave other details as default.

ii. Select **Next**.

c. In the **Configure stack options** page:

Note: Leave all the details as default.

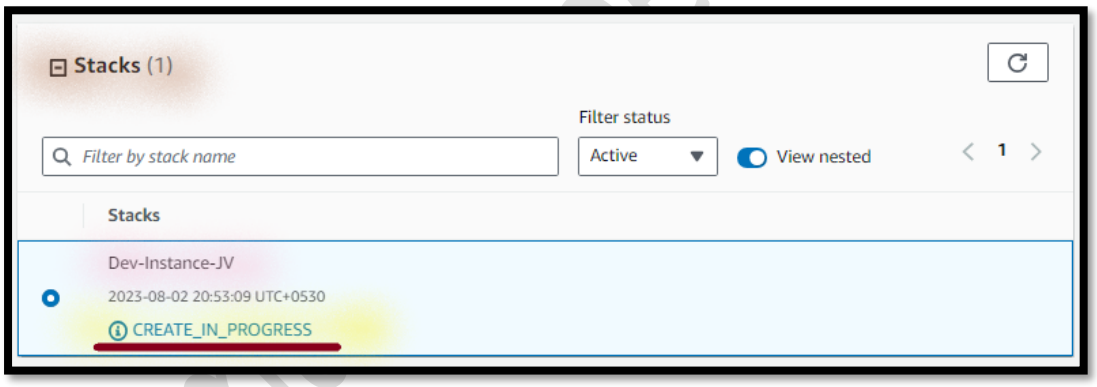
i. Select **Next**.

d. In the **Review Dev-Instance-JV** page:

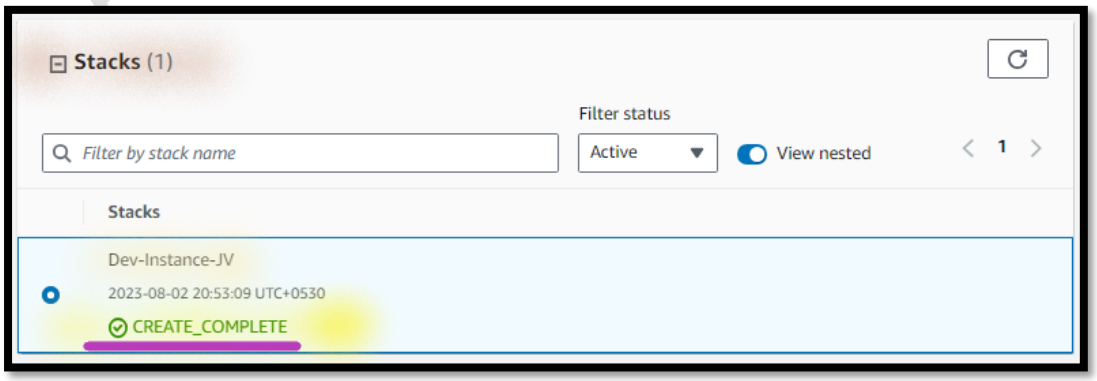
Note: Review all the details.

i. Select **Submit**.

Note: You can see the **Stack** status as **CREATE_IN_PROGRESS**.



Note: **Wait**, till you can see the **Stack** status as **CREATE_COMPLETE**. You can **Refresh** your screen



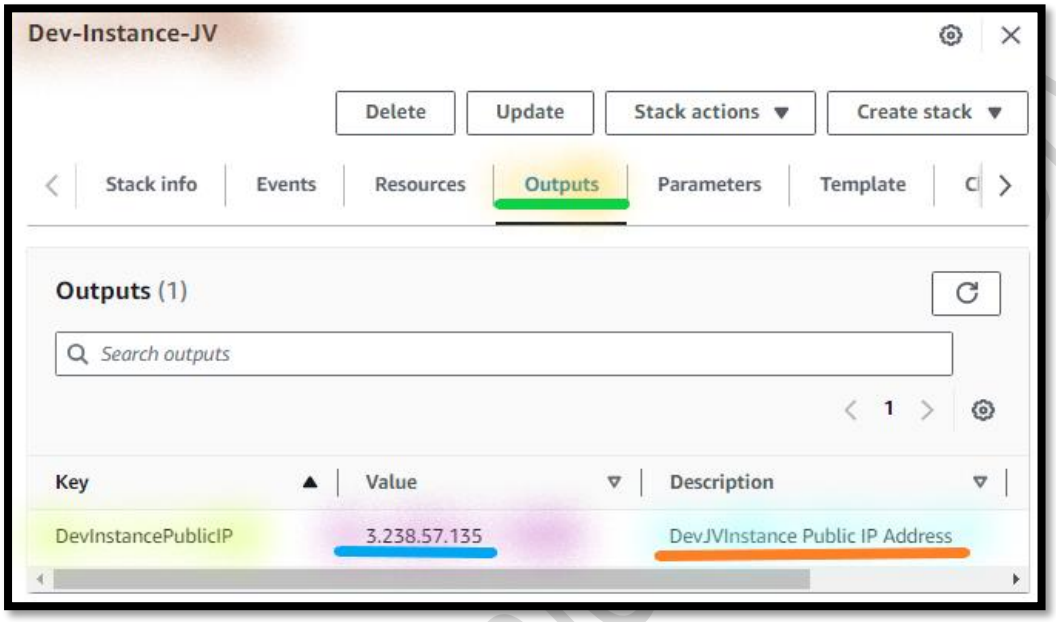
© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Step 2: View the Output

7. **From** the **Dev-Instance-JV** **CloudFormation** console:

    a. Select **Outputs**.

Note: **Copy** the **DevJVInstance Public IP** address in the **Notepad**.



Step 3: Connect to Instance

8. From the **Local Desktop/ Laptop** (*Windows desktop*), right click on **Start** & **Run**.

    a. In the **Open**, write **mstsc**.

    b. Select **Ok**.

        i. **From** the **Remote Desktop Connection**:

            a) **Computer**: Write the **Public IP Address** of the **DevJVInstance**.

            b) Select **Connect**.

Note: You can **get the prompt** to enter the **Username** and **Password**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

1) **Username:** Write **Administrator**.

2) **Password**: Write **lab-password@123**.

3) Select **Ok**.

Step 4: Install the Java Development Kit

9. From the **DevJVInstance** (*Windows Server 2022*).

   a. **Download** and **install** the ***Java SE Development Kit 8*** for **Windows x64**.

Note: Use the below **URL** to download the **Java SE Development Kit 8**.

https://bitbucket.org/ahmadzahoory/dev/downloads/jdk-8u351-windows-x64.exe

Info: You can also download the Java SE Development Kit 8 from the Oracle site.
https://www.oracle.com/java/technologies/javase/javase-jdk8-downloads.html

Note: **Wait**, till **Java SE Development Kit 8** install **succesfully**.

Step 5: Check the JDK Version

10. From the **DevJVInstance**, right click on **Start** & **Run**.

    a. In the **Open**, write **cmd**.

    b. Select **Ok**.

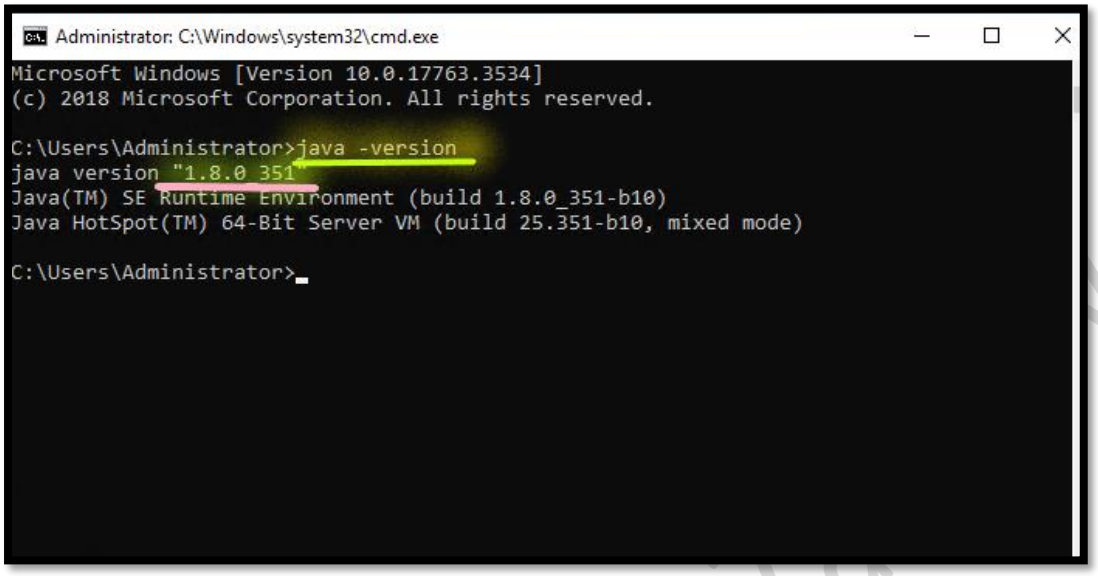
       i. From the **Command line interpreter**:

          a) **Execute** the ***below command*** to **verify** the **Java version**:

```
Java -version
```

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Note: You can see the **Java SDK** installed **version**.



Step 6: Install the IntelliJ IDE

11. From the **DevJVInstance**.

    a. **Download** and **install** the ***IntelliJ IDE 2024*** for **Windows x64**.

Note: Use the below **URL** to download the **IntelliJ IDE 2024 Community edition**.

https://bitbucket.org/ahmadzahoory/devenv/downloads/ideaIC-2024.1.1.exe

Info: You can also download the IntelliJ from the intellij site.
https://www.jetbrains.com/idea/download/?section=windows

Note: **Wait** till **IntelliJ IDE** install **succesfully**.

Note: **Don't launch** the **IntelliJ IDE**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

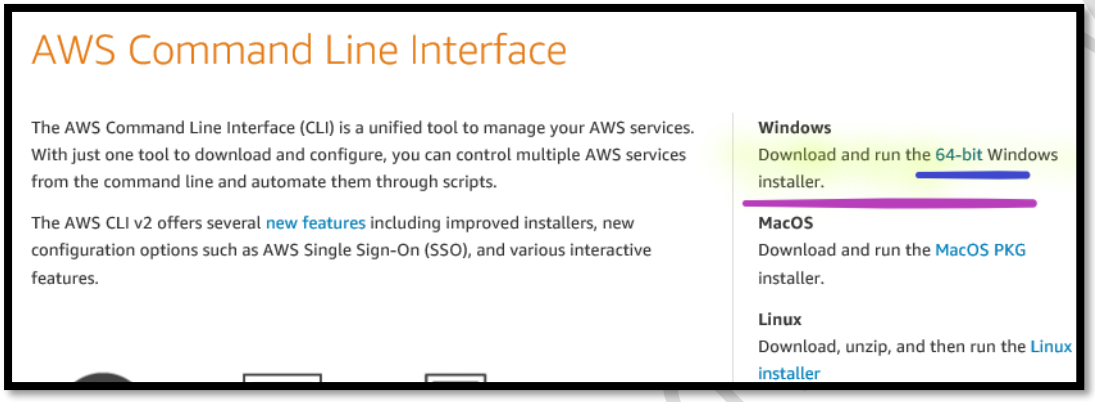
Step 7: Install the AWS CLI V2

12. From the **DevJVInstance**.

    a. **Download** and **install** the **AWS CLI v2**.

Note: Use the below **URL** to download the **AW CLI v2**.

https://aws.amazon.com/cli/



Note: **Wait**, till **AWS CLI v2** install **succesfully**.

Step 8: Check the AWS CLI Version

13. From the **DevJVInstance**, right click on **Start** & **Run**.

    a. In the **Open**, write **cmd**.

    b. Select **Ok**.

        i. From the **Command line interpreter**:

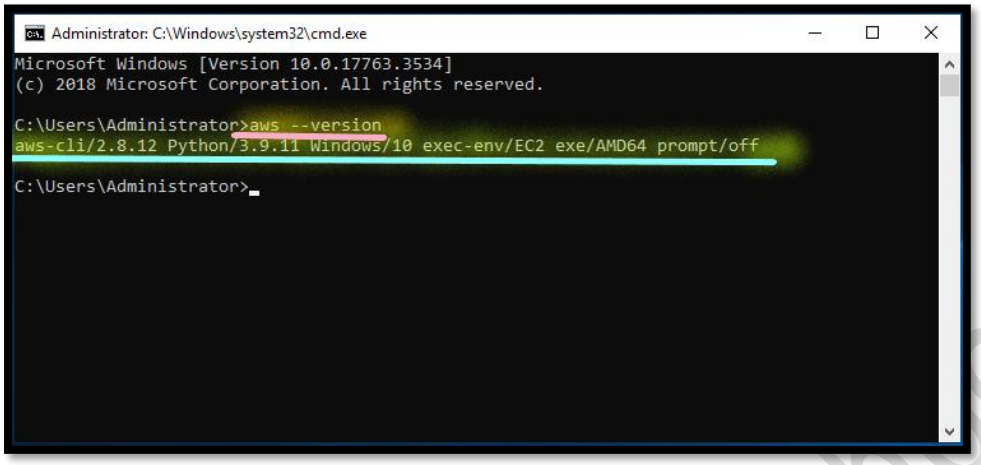
            a) **Execute** the ***below command*** to **verify** the **AWS version**.

```
aws --version
```

Note: You can see the **AWS CLI** installed **version**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Note: If you can see the "**'aws' is not recognized as an internal or external command**" message, **Restart** the **DevJVInstance**.



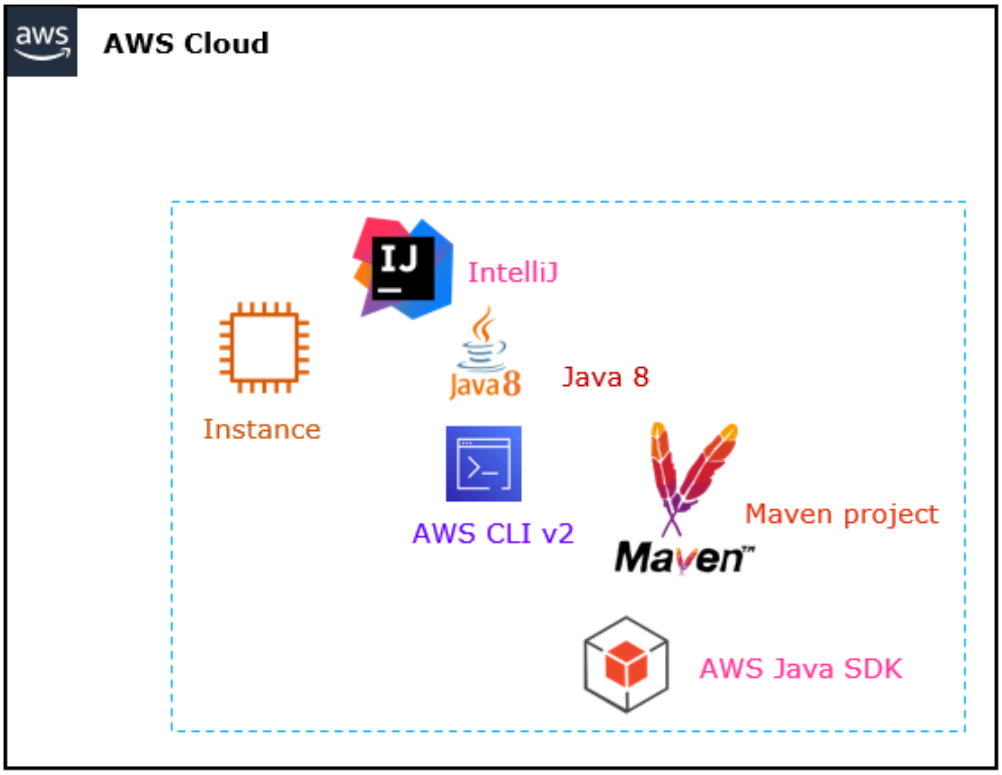
Note: Go to the next task. But **Don't close** the **DevJVInstance**.

Ahmad Majeed Zahoory

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Task 3: Create Maven Project

In this task, you will create Maven project to manage AWS S3 programmatically.



Step 1: Develop the Java Code

14. Unzip the **LAB-m06-01-Code**.zip (Java code).

Note: **lab-m06-01-code**.zip code file is available with the **Lab manual**.

Note: **Review** the **Code**, by **opening** in the **Notepad**.

Step 2: Configure the Credentials and Configuration

15. From the **DevJVInstance**, right click on **Start** & **Run**.

    a. In the **Open**, write **cmd**.

    b. Select **Ok**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

i. From the **Command line interpreter**:

a) **Execute** the ***below command*** to **configure** the **AWS credentials**.

```
aws configure
```

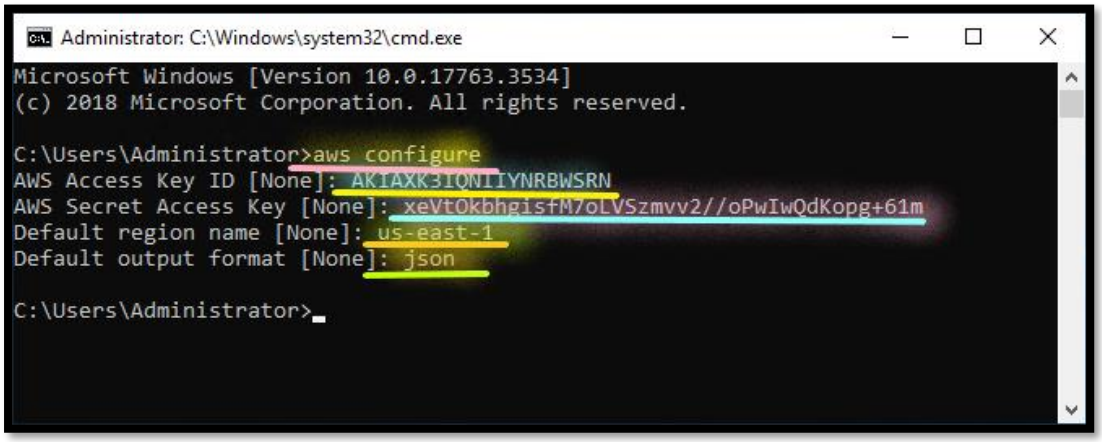
a) **AWS Access Key ID**: Type **Dev-User-YOUR-ID's**, **access key** (*copy from the .csv file*), press **Enter** key to continue.

b) **AWS Secret Access Key**: Type **Dev-User-YOUR-ID's**, **secret access key** (*copy from the .csv file*), press **Enter** key to continue.

Note: Copy the **access key** and **secret access key** of the IAM user **Dev-User** from **.csv file** which you have downloaded in the previous step.

c) **Default region name**: Type **us-east-1**, press **Enter** key to continue.

d) **Default output format**: Type **json**, press **Enter** key to continue.



b) **Execute** the ***below command*** to **exit**.

```
exit
```

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Step 3: Verify the Configuration

16. From the **DevJVInstance**, right click on **Start** & **Run**.

    a. In the **Open**, write **C:\Users\Administrator**.

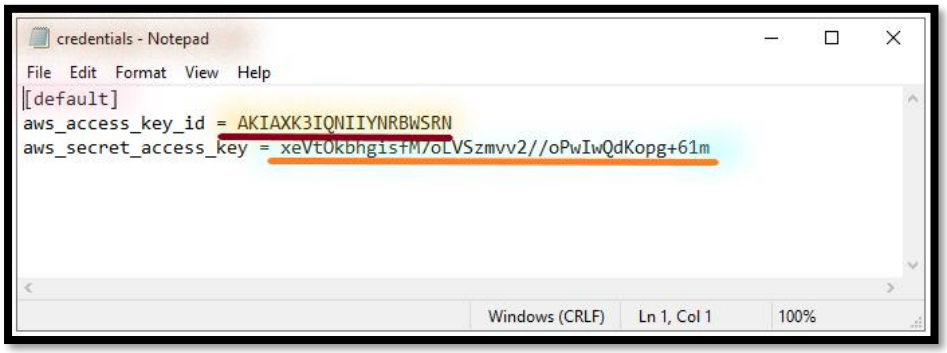
    b. Select **Ok**.

        i. From the **File explorer**:

            a) **Open** the **.aws** folder.

                1) **Open** the **Credentials** file in **Notepad**.

Note: You can see the **access key** and **secret access key** details.

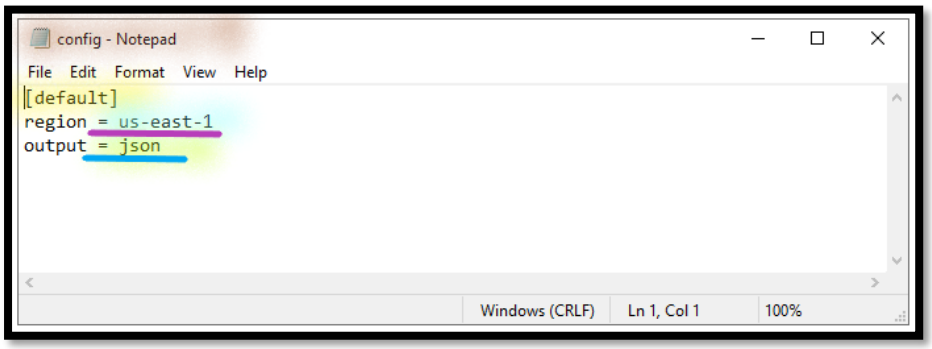


                I. Select **File**.

                II. Select **Exit**.

                2) **Open** the **Config** file in **Notepad**.

Note: You can see the **region** and **output** format details.



© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

1) Select **File**.

2) Select **Exit**.

b) **Close** the **File explorer**.

Step 4: Launch the IntelliJ IDE

17. From the **DevJVInstance**.

    a. Open the **IntelliJ IDE**.

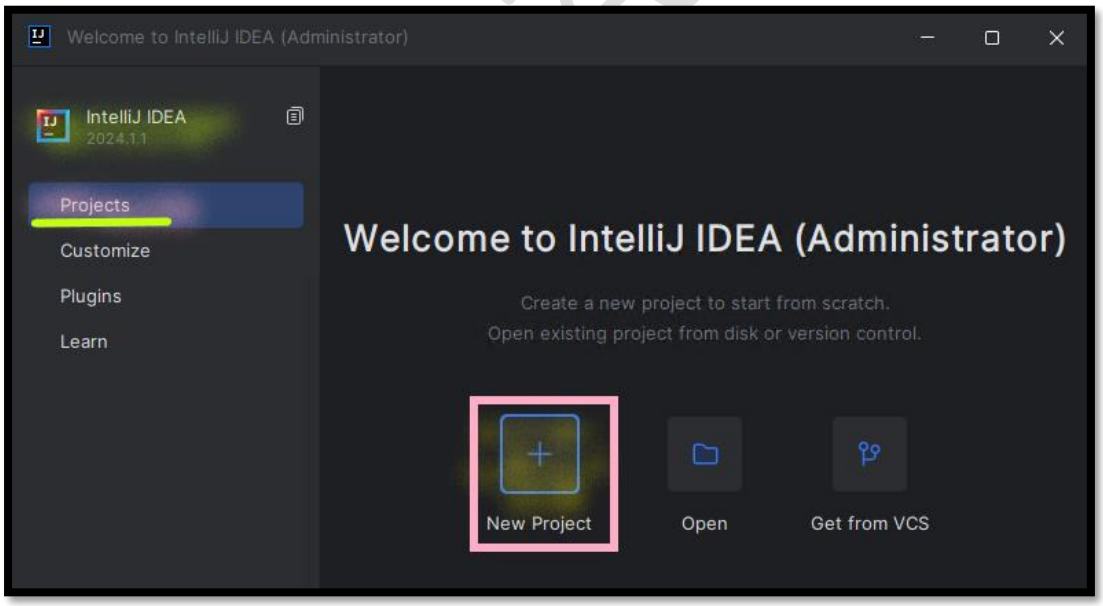
> **Note**: **Wait**, till you can see the **IntelliJ IDE Workspace**.

Step 5: Create Maven Project

18. From the **IntelliJ IDE**.

    a. Select **Project**.

        i. Select **New Project**.



        ii. In the **New project** page:

            a) Select **Java**.

                1) **Name**: Write **AWS-S3**.

                2) **Build system**: Select **Maven**.

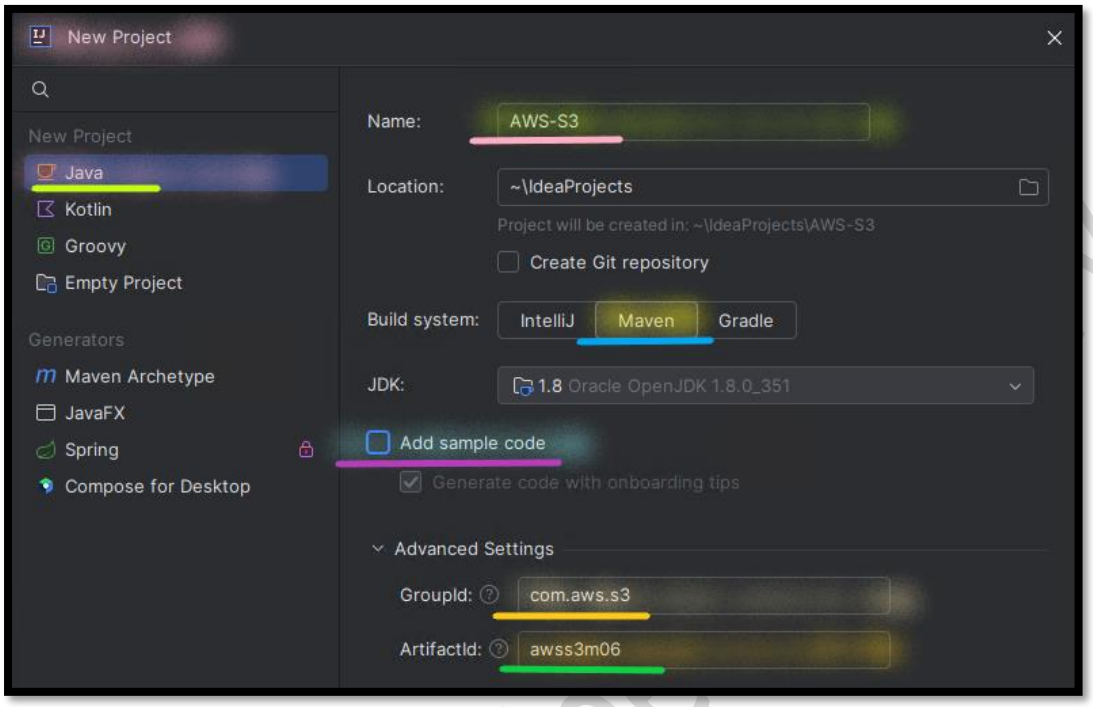
                3) **Add sample code**: **Uncheck** the **Checkmark**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

4) **Expand** the **Advanced Settings**:

I. **Group Id**: Write **com.aws.s3**.

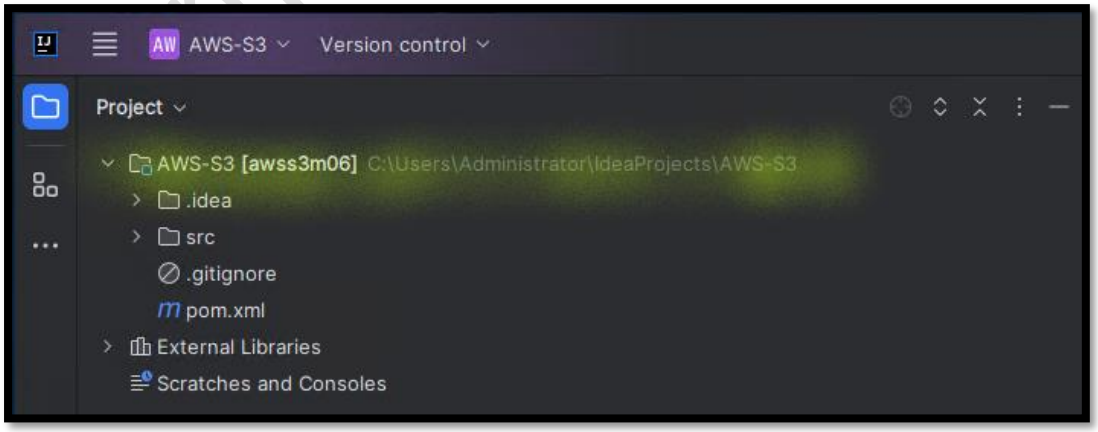
II. **Artifact Id**: Write **awss3m06**.



Note: Leave the other details as default.

5) Select **Create**.

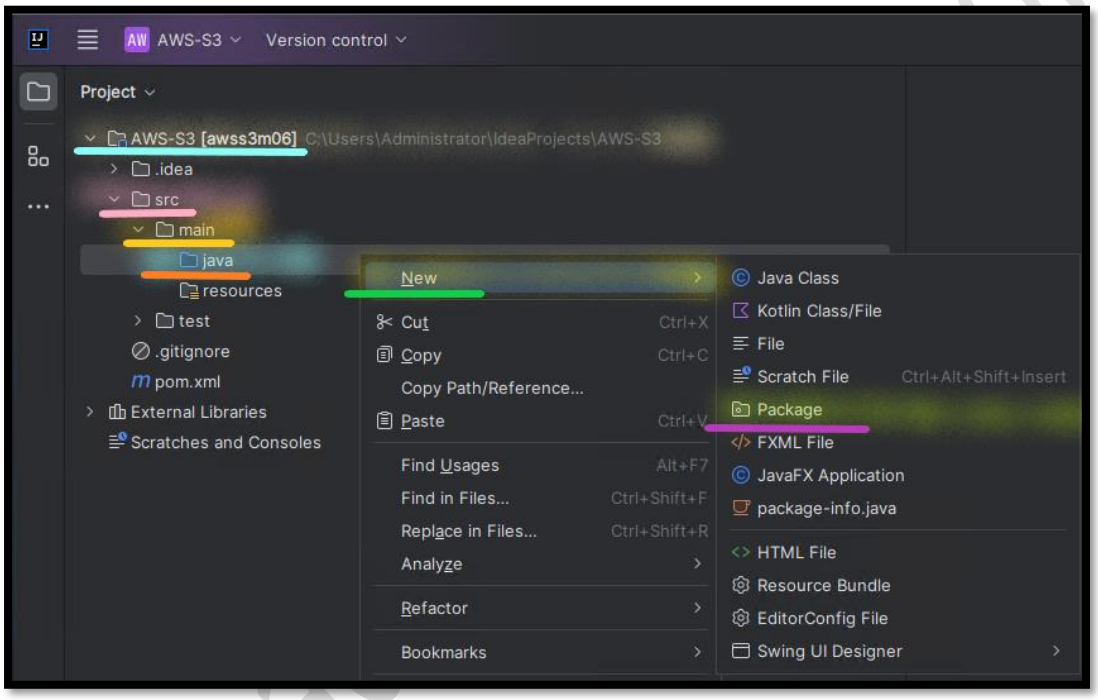
Note: You can see the **AWS-S3** project.



© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Step 6: Create the Package

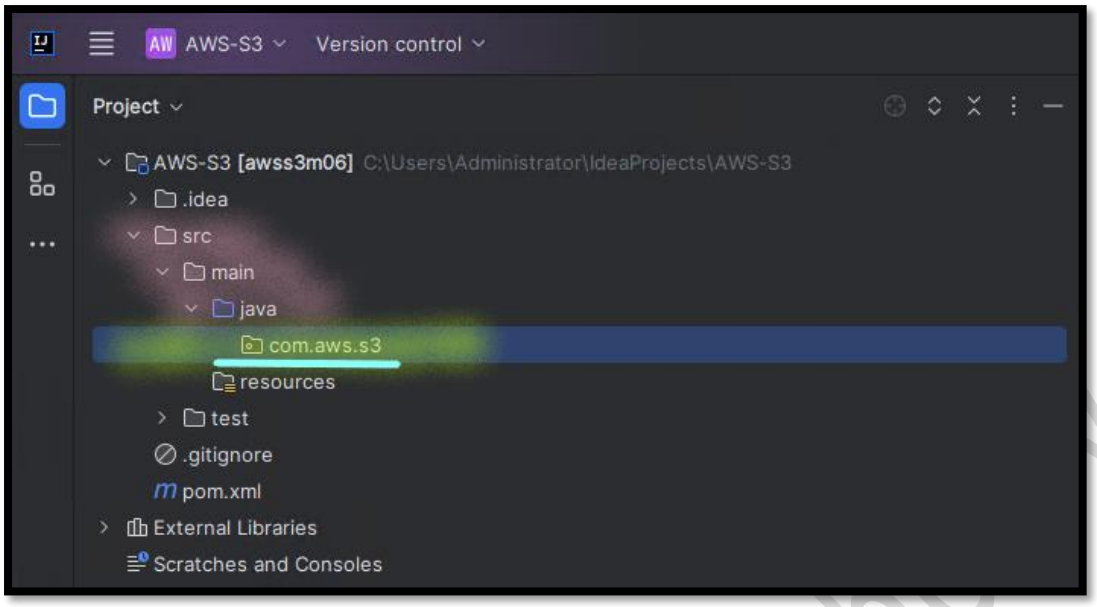
19. From the **IntelliJ IDE**.
    a. **Expand** the **awss3m06** project.
        i. **Expand** the **src**.
            a) **Expand** the **main**.
                1) Select the **Java**.
    b. **Right-click** on the **Java** project.
        i. Select **New**.
            a) Select **Package**.



b) In the **New Package** section:
    1) **Name**: Write **com.aws.s3**.
        I. Press **Enter** key.

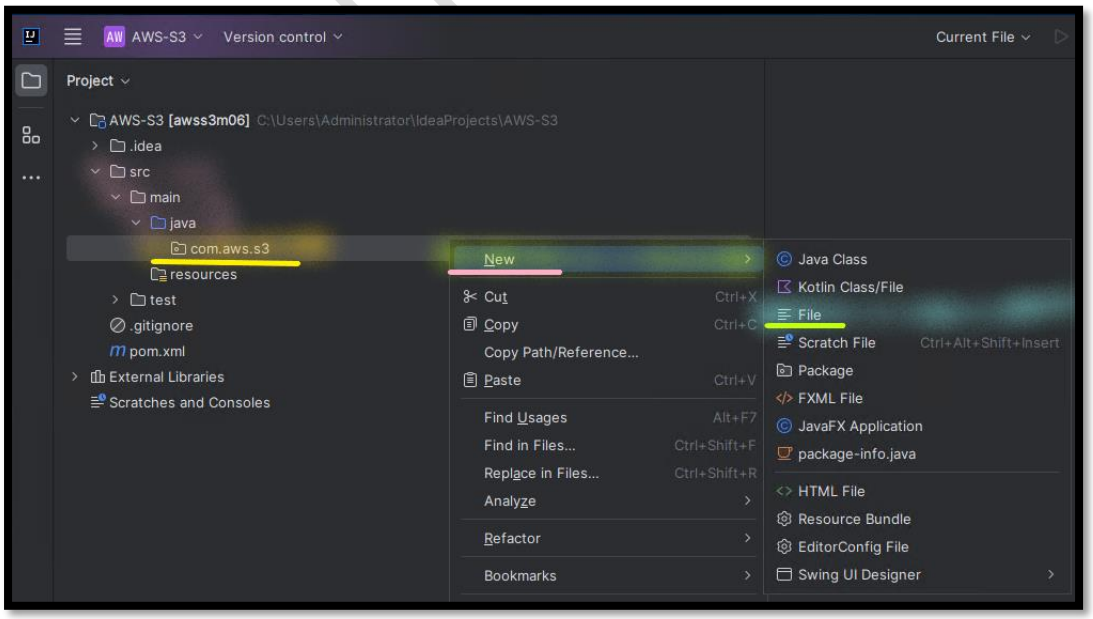
Note: You can see the **com.aws.s3** package under **src/main/java**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.



Step 7: Create the Executor.java File in the Package

20. From the **IntelliJ IDE**.
    a. **Expand** the **awss3m06** java project.
        i. **Expand** the **src/main/java** resource path.
            a) **Select** the **com.aws.s3** java package.
            b) **Right-click** on the **com.aws.s3** java package.
                1) Select **New**.
                2) Select **File**.

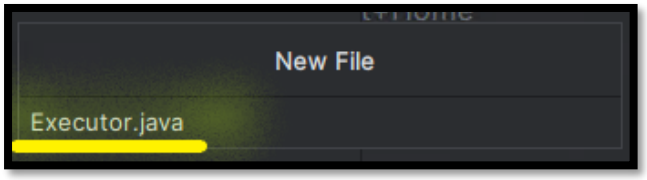


© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

c) In the **New File** section:

1) **Name**: Write **Executor.java**.

Note: Ensure that in the executor, **E** should be **Capital**.



I. Press **Enter** key.

Step 8: Create the Utilities.java File in the Package

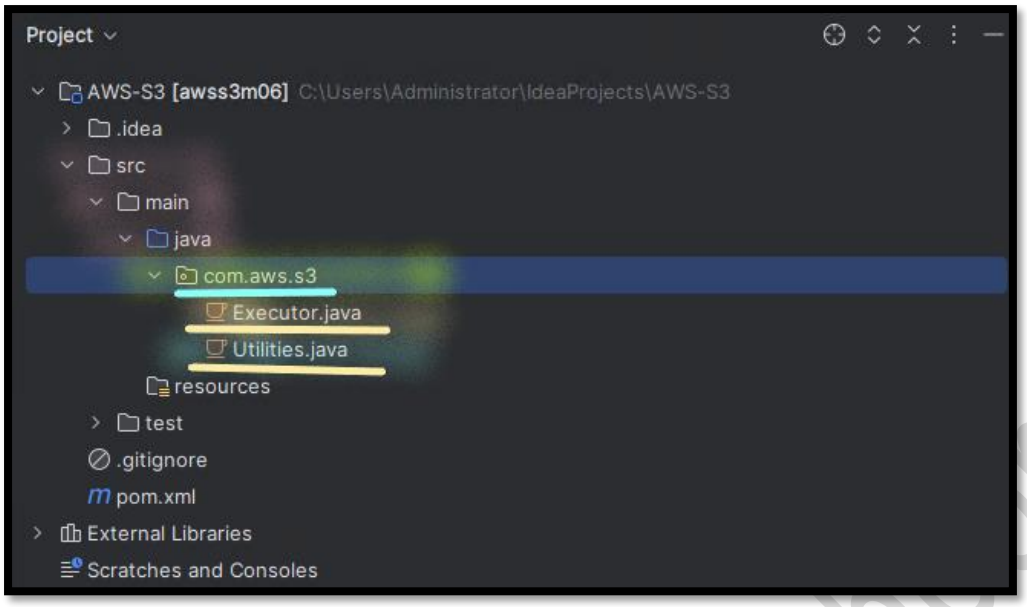
21. From the **IntelliJ IDE**.
    a. **Expand** the **awss3m06** java project.
        i. **Expand** the **src/main/java** resource path.
            a) **Select** the **com.aws.s3** java project.
            b) **Right-click** on the **com.aws.s3** java package.
                1) Select **New**.
                2) Select **File**.
            c) In the **New File** section:
                1) **File name**: Write **Utilities.java**.

Note: Ensure that in the utilities, **U** should be **Capital**.

I. Press **Enter** key.

Note: You can see the **Executor.java** and **Utilities.java** under Java package.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.



Step 9: Update the Java Code

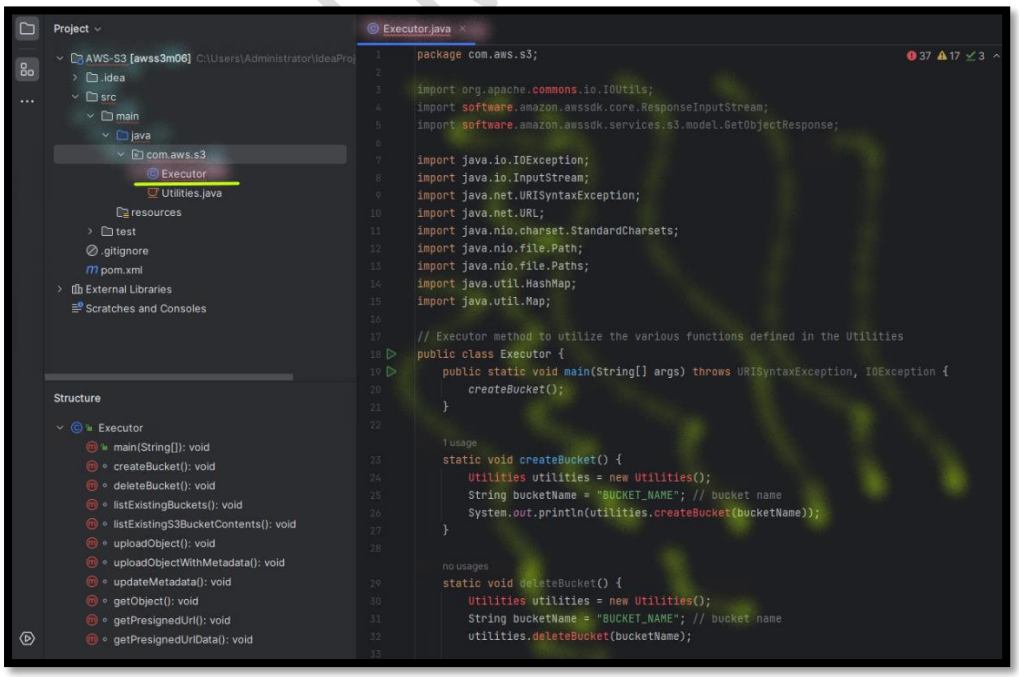
Update the Executor.java

22. **Double-click** on the **Executor.java** file.

    a. **Paste** the **Code** from **Executor.txt** file.

Note: Code is available with the **Lab manual**.

i. Select **CTRL + S** *(to save)*.



© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

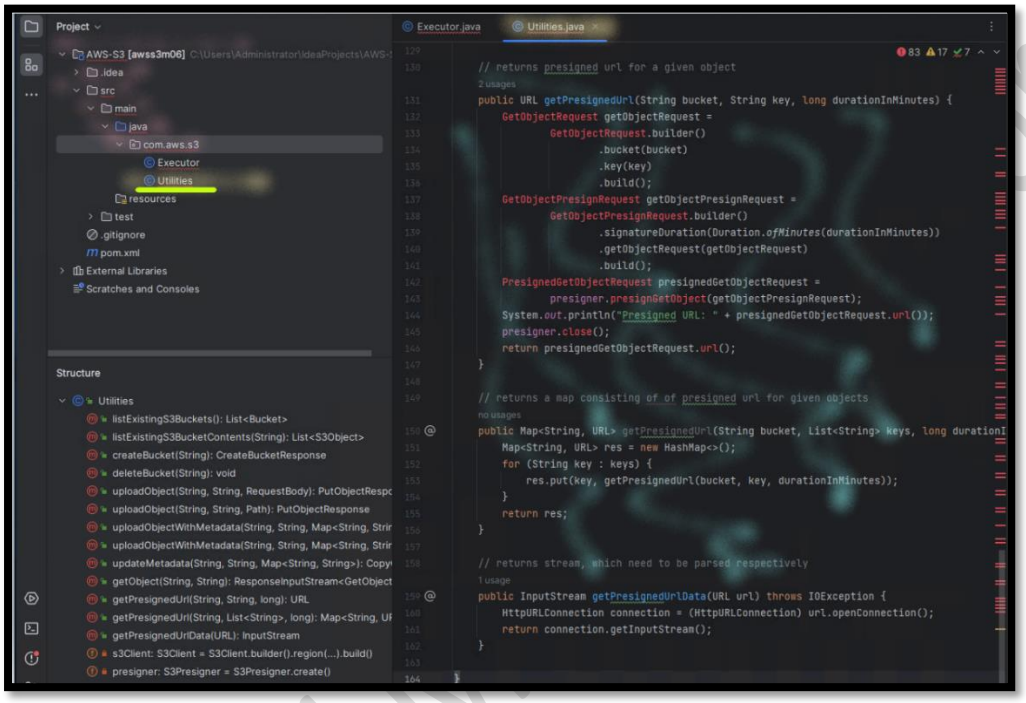
Update the Utilities.java

23. **Double-click** on the **Utilities.java** java file.

    a. **Paste** the **Code** from **Utilities**.txt file.

Note: Code is available with the **Lab manual**.

    i. Select **CTRL + S** *(to save)*.



Update the Pom.xml

24. **Double-click** on the **pom.xml**.

    a. **Remove** the **existing code**.

    b. **Copy** the **Code** from ***pom.xml*** file.

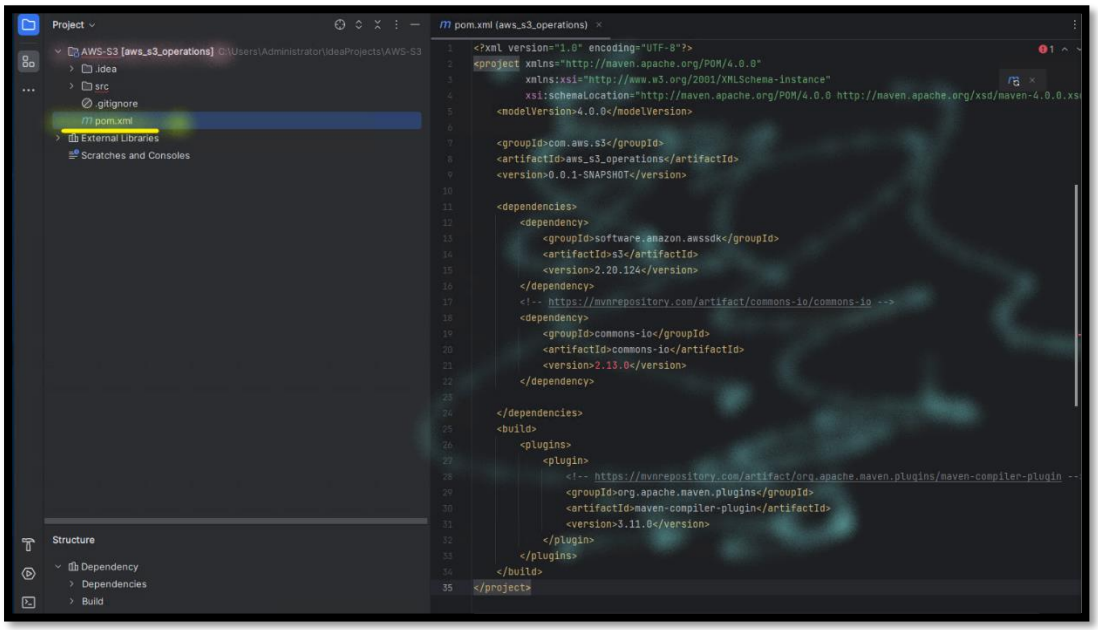
Note: Code is available with the **Lab manual**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Note: In the **POM** you can see the:

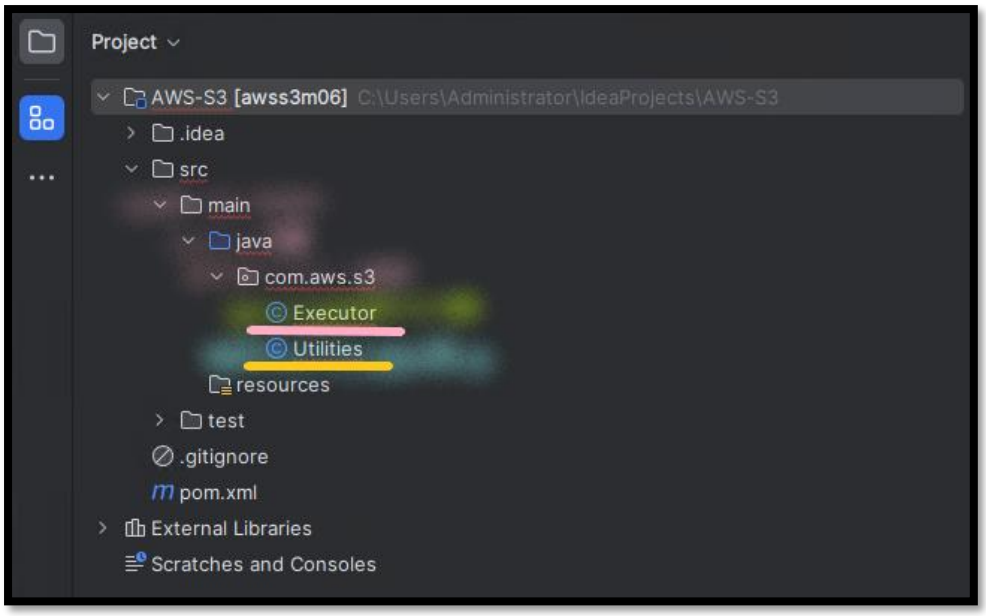
1. **AWS Java SDK** [*software.amazon.awssdk*] - The AWS Java SDK for Amazon S3 module holds the client classes that are used for communicating with Amazon Simple Storage Service.
2. **Apache Common IO** [*common-io*] - The Apache Commons IO library contains utility classes, stream implementations, file filters, file comparators, endian transformation classes, and much more.
3. **Apache Maven Compiler Plugin** [*org.apache.maven.plugins*] - The Compiler Plugin is used to compile the sources of your project.

i. Select **CTRL + S** *(to save)*.



Note: You can see the **Error** against **Executor.java** and **Utilities.java** under Java package.

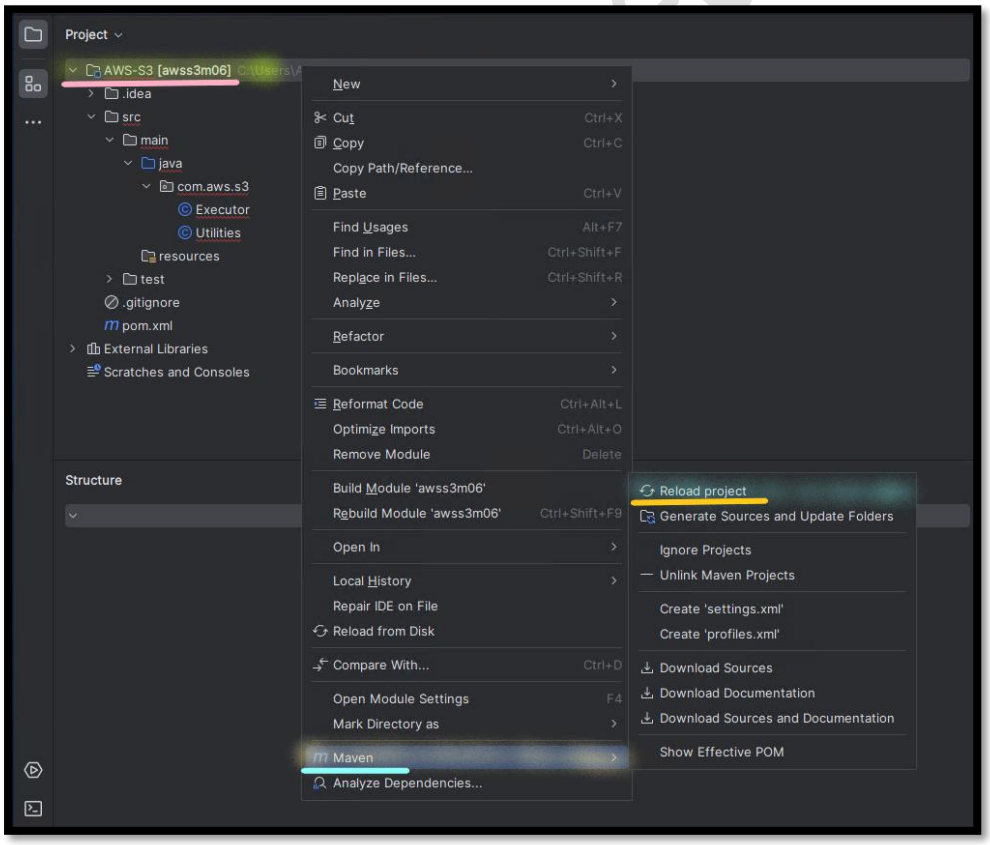
© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.



25. **Right-click** on the **awss3m06** java project.

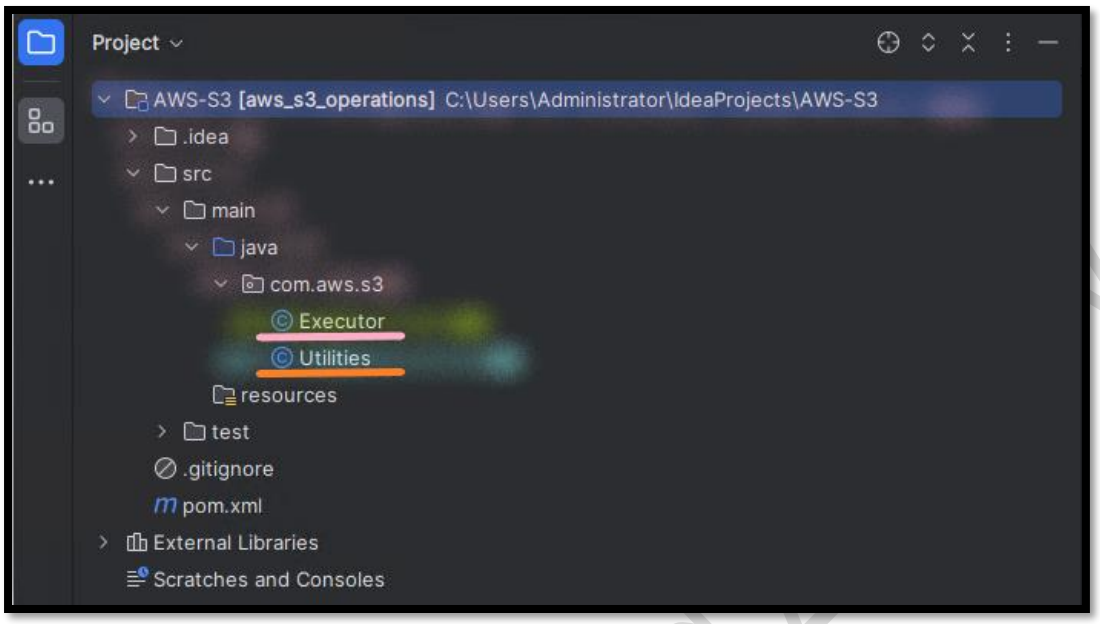
   a. Select **Maven**.

      i. Select **Reload project**.



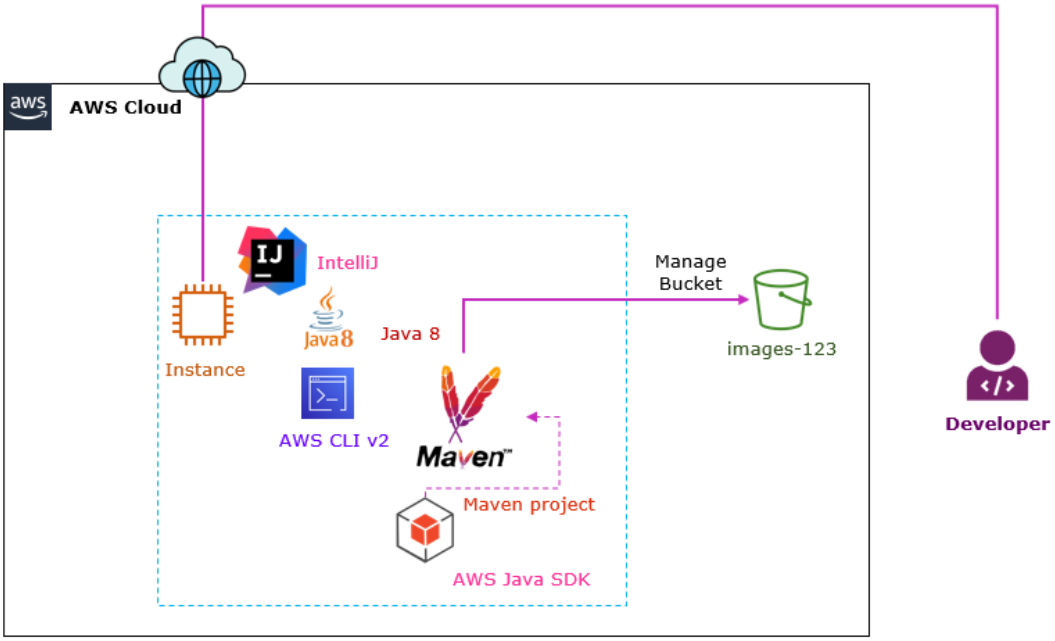
© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Note: **Wait** for **indexing** and **ensure** that you **Don't see** any **Errors** in the Java project.



Task 4: Manage AWS S3 from IntelliJ

In this task, you will manage AWS S3 from IntelliJ using Java.



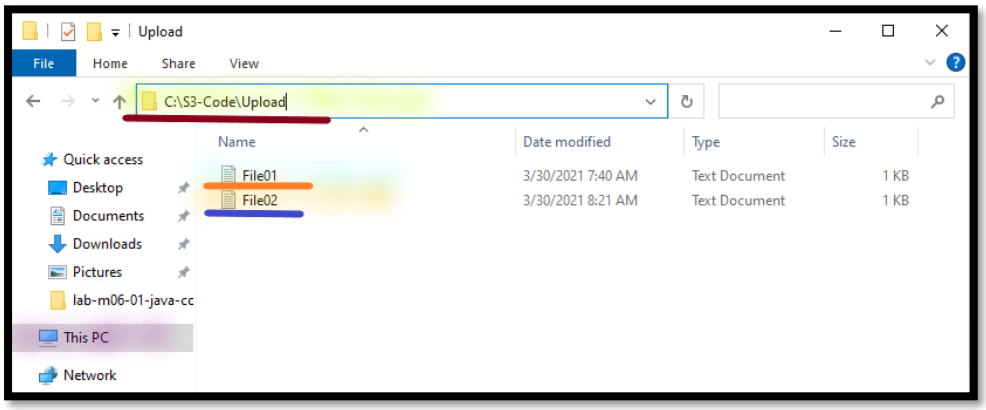
© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Step 1: Copy the Objects

26. From the **DevJVInstance** (*Windows Server 2022*), right click on **Start** & **Run**.
    a. In the **Open**, write **C:**.
    b. Select **Ok**.
        i. From the **File explorer**:
            a) **Create** **S3-Code** folder in **C drive**.
            b) **Copy** the **Upload** **folder** in the **S3-Code** folder.

Note: **Upload** folder is available with the **Lab manual**.

Note: You can view the **File01.txt** and **File02.txt** file **content**.



Step 2: Create New Bucket

27. **From** the **IntelliJ IDE**.
    a. **Expand** the **awss3m06** java project.
        i. **Expand** the **src/main/java** resource path.
            a) **Select** the **com.aws.s3** java project.
                1) **Double-click** on the **Executor.java** java file.

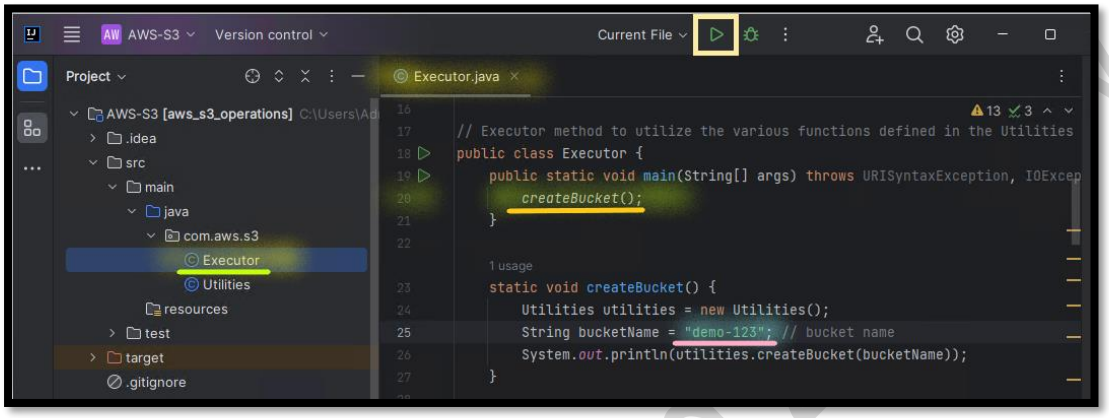
© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

28. You can see the ***createBucket*** method in the ***line no. 20***.

a. **Replace** the **BUCKET_NAME** with the **demo-123** bucket name you want to create in the ***line no. 25***.

Note: **Don't Replace** the **start** and **end** quote (**" "**).

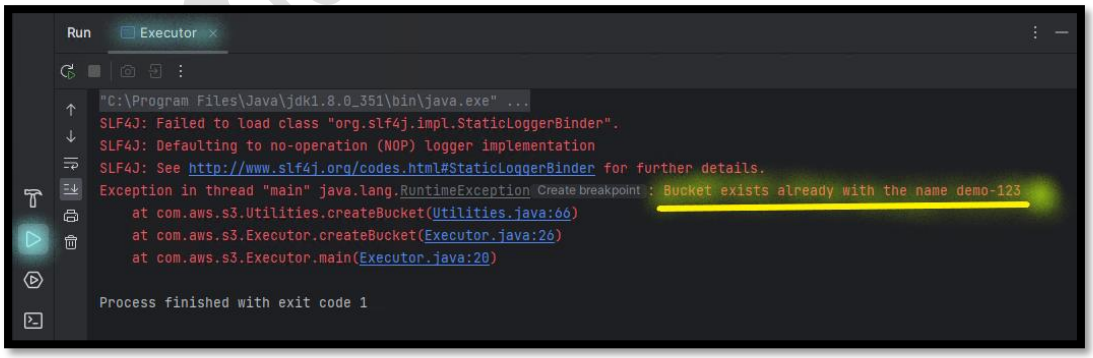
i. **Execute** the **Run Executor**.



Note: If bucket **Created succesfully**, in the **Console**, you will see the "**CreateBucketResponse**" message.

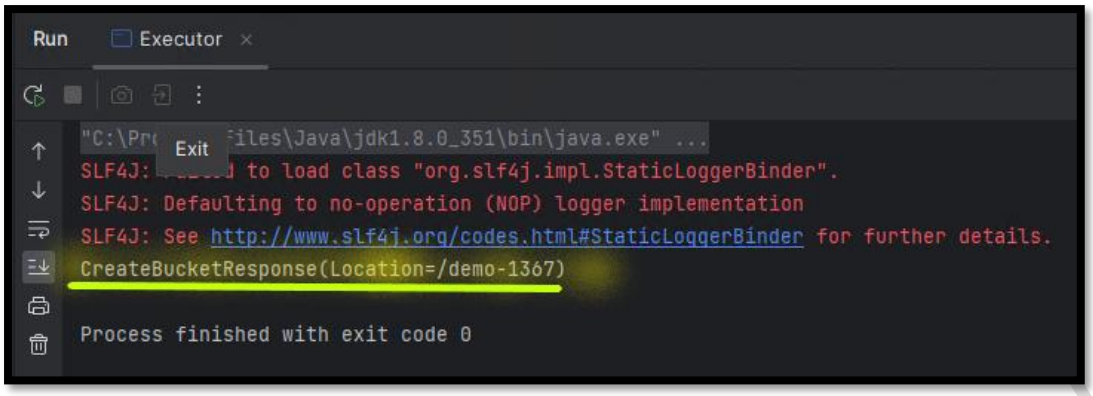
Note: If bucket **Already exist**, in the **Console**, you will see the "**Bucket exist already with the name demo-123**" message.

Note: If you are getting error **Bucket exist already exist** in console, **Replace 123** to make the bucket name unique.



Note: **Replace 123** to make the **bucket name unique** and create the Bucket again.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.



Step 3: Create New [Second] Bucket

29. You can see the ***createBucket*** method in the ***line no. 20***.

    a. **Replace** the **demo-123** with the **images-123**, bucket name you want to create in the ***line no. 25***.

        i. **Execute** the **Run Executor**.

Note: If bucket **Created succesfully**, in the **Console**, you will see the "**CreateBucketResponse**" message.

Note: If bucket **Already exist**, in the **Console**, you will see the "**Bucket exist already with the name images-123**" message.

Note: If you are getting error **Bucket exist already exist** in console, **Replace 123** to make the bucket name unique.

Step 4: List the Existing Buckets

30. **From** the ***Executor.java***.

    a. **Replace** the ***createBucket*** method with the **listExistingBuckets** method in the ***line no. 20***.

Note: Ensure there should be no space between **listExistingBuckets** and **start** & **end** bracket **()**. It should be like **listExistingBuckets()**.

        i. **Execute** the **Run Executor**.

Note: In the **Console**, you will see the **demo-123** and **images-123** bucket details.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Note: You may also see **other buckets** created by CloudFormation.

```
Run    Executor
"C:\Program Files\Java\jdk1.8.0_351\bin\java.exe" ...
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
[Bucket(Name=cf-templates-vib2vpwu8sff-us-east-1, CreationDate=2023-08-01T05:45:15Z), Bucket
 (Name=demo-1367, CreationDate=2023-08-15T10:27:02Z), Bucket(Name=images-173,
 CreationDate=2023-08-15T10:15:49Z)]

Process finished with exit code 0
```

Step 5: Delete the Bucket

31. **From** the ***Executor.java***.

    a. **Replace** the **listExistingBuckets** method with the ***deleteBucket*** method in the ***line no. 20***.

    b. **Replace** the **BUCKET_NAME** with the **demo-123**, bucket name (***which you have created in the previous step***) in the ***line no. 31***.

        i. **Execute** the **Run Executor**.

Step 6: List the Existing Buckets

32. **From** the ***Executor.java***.

    a. **Replace** the ***deleteBucket*** method with the **listExistingBuckets** method in the ***line no. 20***.

        i. **Execute** the **Run Executor**.

Note: In the **Console**, you will see only the **images-123** bucket details.

Step 7: Upload the New Object (File01.txt)

33. **From** the ***Executor.java***.

    a. **Replace** the ***listExistingBuckets*** method with the **uploadObject** method in the ***line no. 20***.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

b. **Replace** the **BUCKET_NAME** with the **images-123**, bucket name (*which you have created in the previous step*) in the ***line no. 49***.

Note: In the code, we have already defined the **upload file path**, (**File01.txt**) which you have copied in the S3-Code folder in C drive.

i. **Execute** the **Run Executor**.

Note: If Object uploaded succesfully, in the **Console**, you will see the **PutObjectResponse**.

Step 8: List Bucket Objects

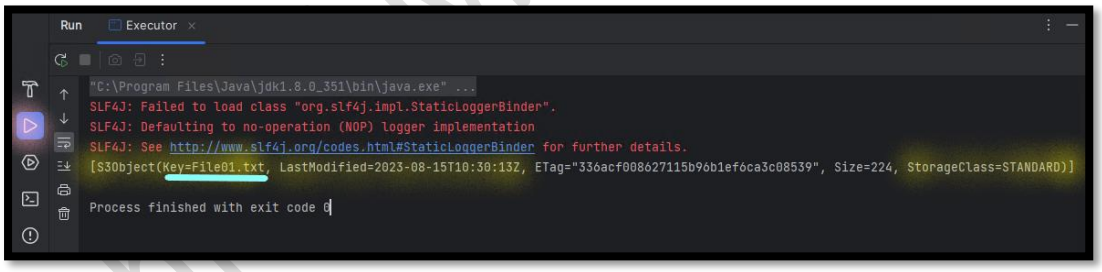
34. **From** the ***Executor.java***.

a. **Replace** the ***uploadObject*** method with the ***listExistingS3BucketContents*** method in the ***line no. 20***.

b. **Replace** the **BUCKET_NAME** with the **images-123**, bucket name (which you have created in the previous step) in the ***line no. 43***.

i. **Execute** the **Run Executor**.

Note: In the **Console**, you will see the **Object details**.



Step 9: Get (Read) Existing Object (File01.txt)

35. **From** the ***Executor.java***.

a. **Replace** the ***listExistingS3BucketContents*** method with the **getObject** method in the ***line no. 20***.

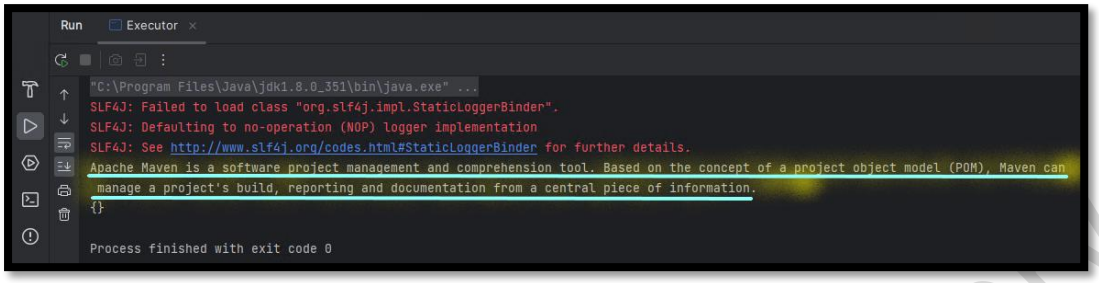
b. **Replace** the **BUCKET_NAME** with the **images-123**, bucket name (which you have created in the previous step) in the ***line no. 77***.

Note: In the code, we have already defined the file name (**File01.txt**).

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

i. **Execute** the **Run Executor**.

Note: In the **Console**, you will see the **Object content**.



Step 10: Upload New Object with Metadata (File02.txt)

36. **From** the ***Executor.java***.

a. **Replace** the ***getObject*** method with the ***uploadObjectWithMetadata*** method in the ***line no. 20***.

b. **Replace** the **BUCKET_NAME** with the **images-123**, bucket name (which you have created in the previous step) in the ***line no. 57***.

Note: In the **code**, we have already defined upload **file path** and file name (**File02.txt**) which you have copied in the S3-Code folder in C drive and **Project** and **Owner metadata**.

i. **Execute** the **Run Executor**.

Note: If Object uploaded succesfully, in the **Console**, you will see the **PutObjectResponse.**

Step 11: Get (Read) Existing Object (File02.txt)

37. **From** the ***Executor.java***.

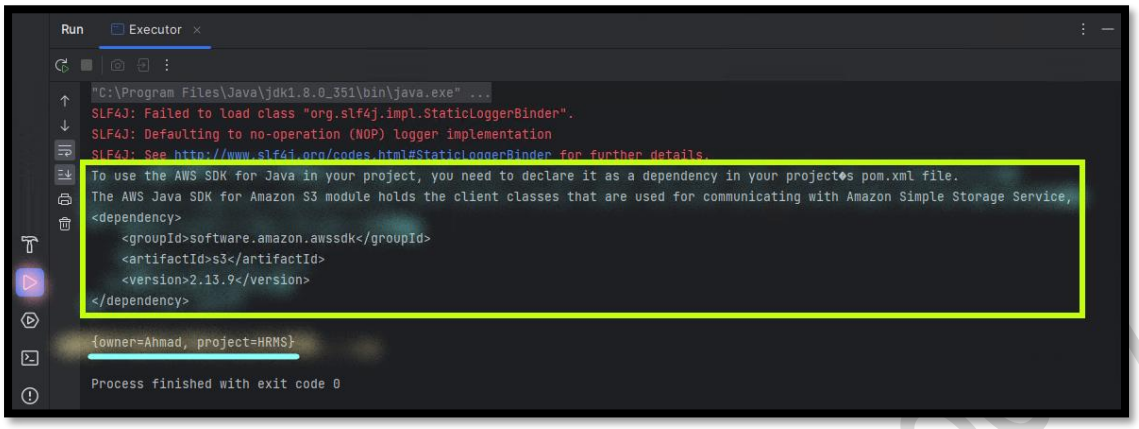
a. **Replace** the ***uploadObjectWithMetadata*** method with the **getObject** method in the ***line no. 20***.

b. **Replace** the **File01**.txt with the **File02**.txt, object name in the ***line no. 78***.

i. **Execute** the **Run Executor**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Note: In the **Console**, you will see the **Object content** with **Metadata**.



Step 12: Update Metadata with the Existing Object (File01.txt)

38. **From** the ***Executor.java***.

    a. **Replace** the ***getObject*** method with the ***updateMetadata*** method in the ***line no. 20***.

    b. **Replace** the **BUCKET_NAME** with the **images-123**, bucket name (which you have created in the previous step) in the ***line no. 68***.

Note: In the **code**, we have already mentioned file name (**File01.txt**) and the **Project metadata**.

i. **Execute** the **Run Executor**.

Note: If Object uploaded succesfully, in the **Console**, you will see the **CopyObjectResponse**.

Step 13: Get (Read) Existing Object (File01.txt)

39. **From** the ***Executor.java***.

    a. **Replace** the ***updateMetadata*** method with the **getObject** method in the ***line no. 20***.

    b. **Replace** the **File02**.txt with the **File01**.txt, object name in the ***line no. 78***.

i. **Execute** the **Run Executor**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Note: In the **Console**, you will see the **Object content** with **Metadata**.

Step 14: Create Pre-Signed URL

40. **From** the ***Executor.java***.

    a. **Replace** the ***getObject*** method with the ***getPresignedUrl*** method in the ***line no. 20***.

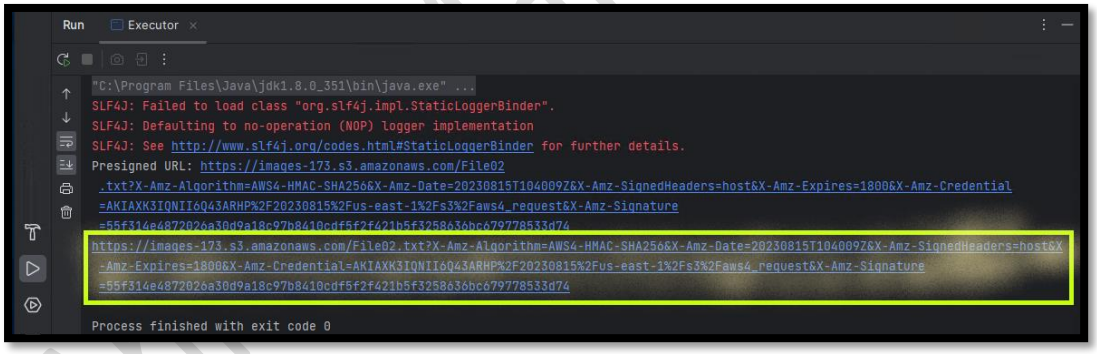
    b. **Replace** the **BUCKET_NAME** with the **images-123**, bucket name (which you have created in the previous step) in the ***line no. 86***.

Note: In the **code**, we have already mentioned file name (**File02.txt**).

i. **Execute** the **Run Executor**.

Note: In the **Console**, you will see the **Pre-signed URL**.

Note: **Copy** the **Pre-signed URL** in the **Notepad**.



Step 15: Get (Read) Object using Pre-Signed URL

41. **From** the ***Executor.java***.

    a. **Replace** the ***getPresignedUrl*** method with the ***getPresignedUrlData*** method in the ***line no. 20***.

    b. **Replace** the **PRE-SIGNED_URL** with the ***Pre-Signed URL*** which you have created in the previous step in the ***line no. 93***.

i. **Execute** the **Run Executor**.

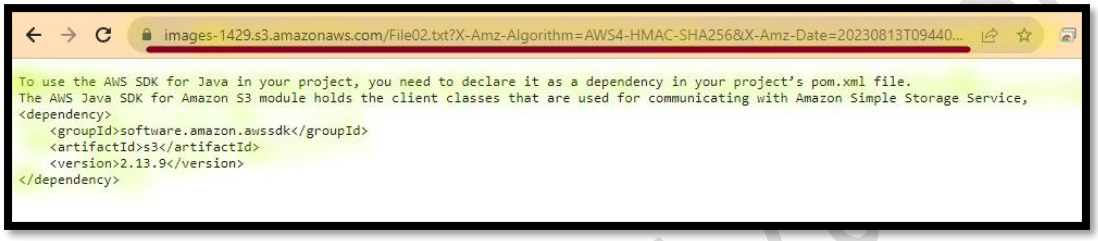
© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Note: In the **Console**, you will see the **Object content**.

Step 16: Access the Object from Browser

42. **From** your **Local Desktop/ Laptop**, open the **Browser**, write **Pre-Signed_URL** of **File02.txt**, to access the ***Object content***.

Note: You can see the **File02.txt** Object content.



Task 5: Install AWS Toolkit for IntelliJ

Step 1: Close the Maven Project

43. **From** the **IntelliJ IDE**.

    a. Select the **File**.

        i. Select **Close Project**.

Step 2: Install the AWS Toolkit

44. From the **IntelliJ IDE**.

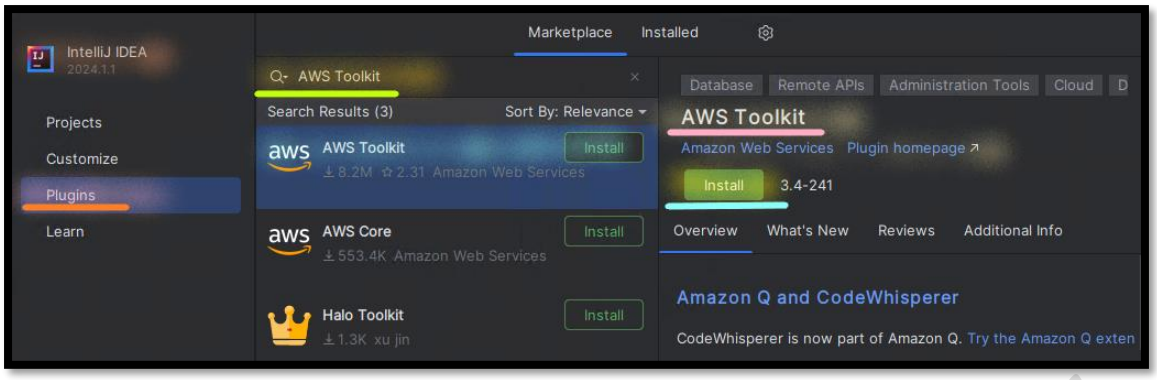
    a. Select **Plugins**.

        i. In the **Search box**, type **AWS Toolkit**.

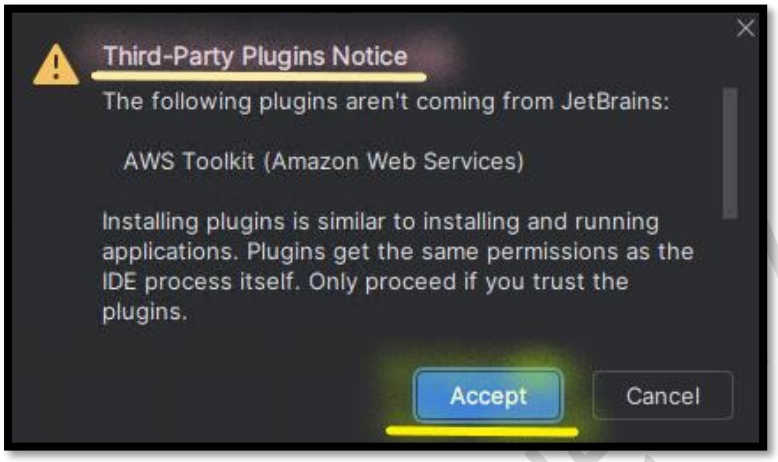
Note: You can see the **AWS Toolkit**.

a) Select **Install**.

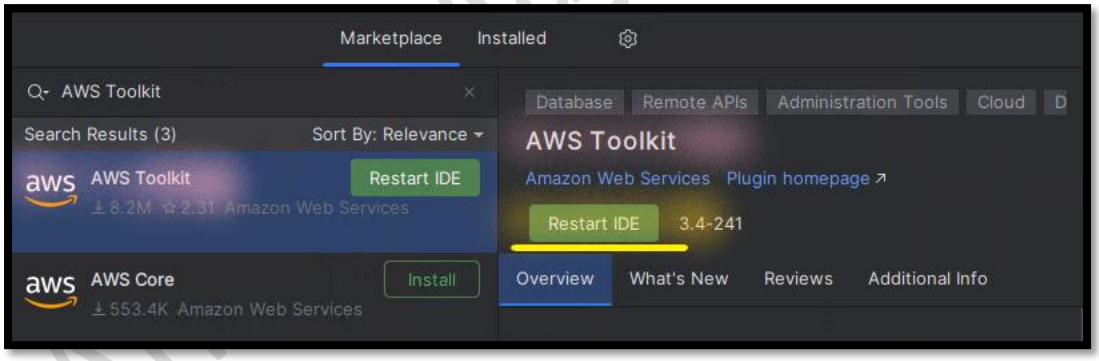
© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.



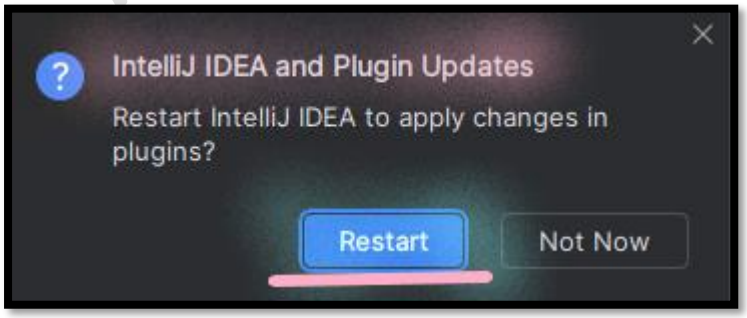
1) **Third-Party Plugins Notice**: Select **Accept**.



2) Select **Restart IDE**.



I. Select **Restart**.



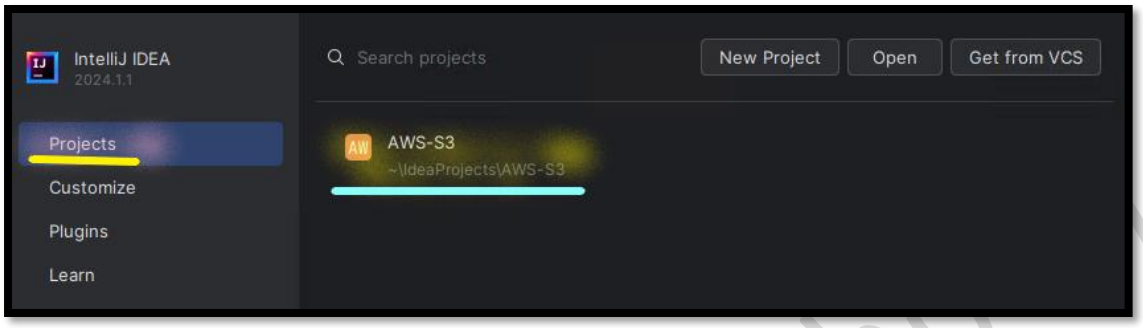
© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Step 3: Install the AWS Toolkit

45. From the **IntelliJ IDE**.

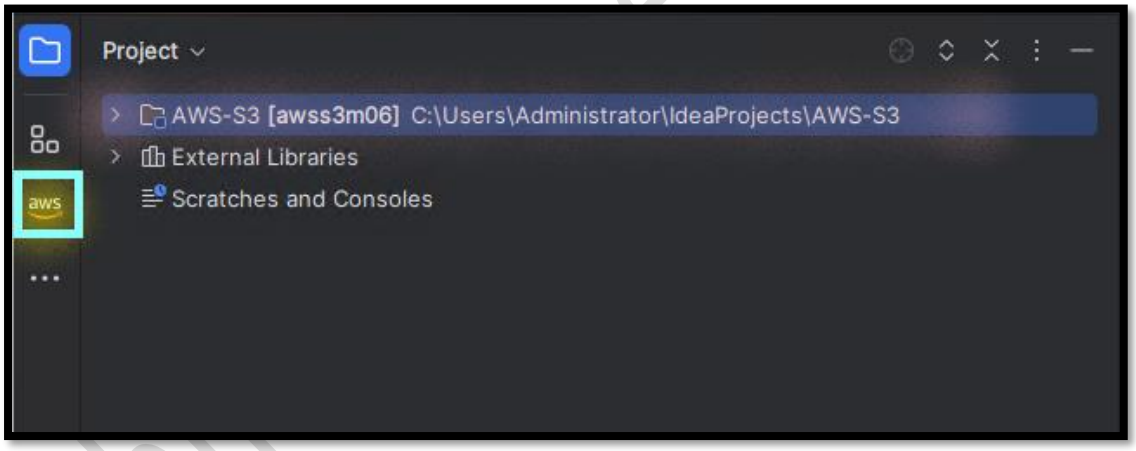
    a. Select **Project**.

        i. Select **AWS-S3**.



Note: You can see the **AWS-S3** project.

a) Select **AWS Toolkit**.



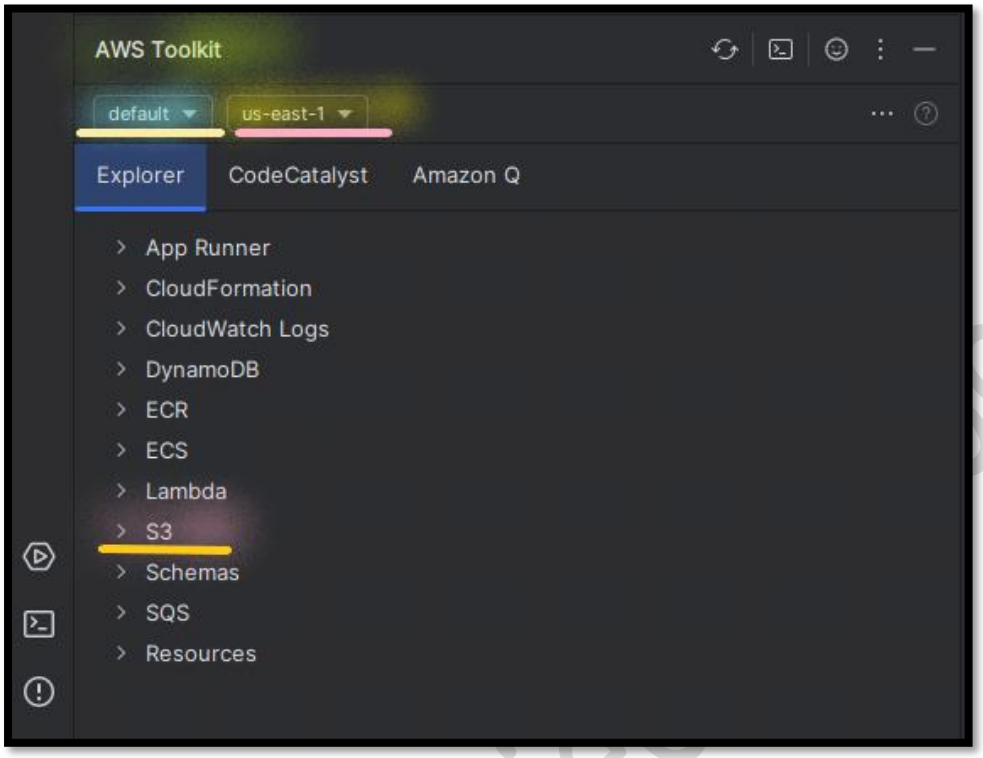
Note: You can see the **AWS Toolkit**.

Note: You can see the **Default credentials profile**.

Note: You can see the **Region**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Note: You can see the **AWS Services**.



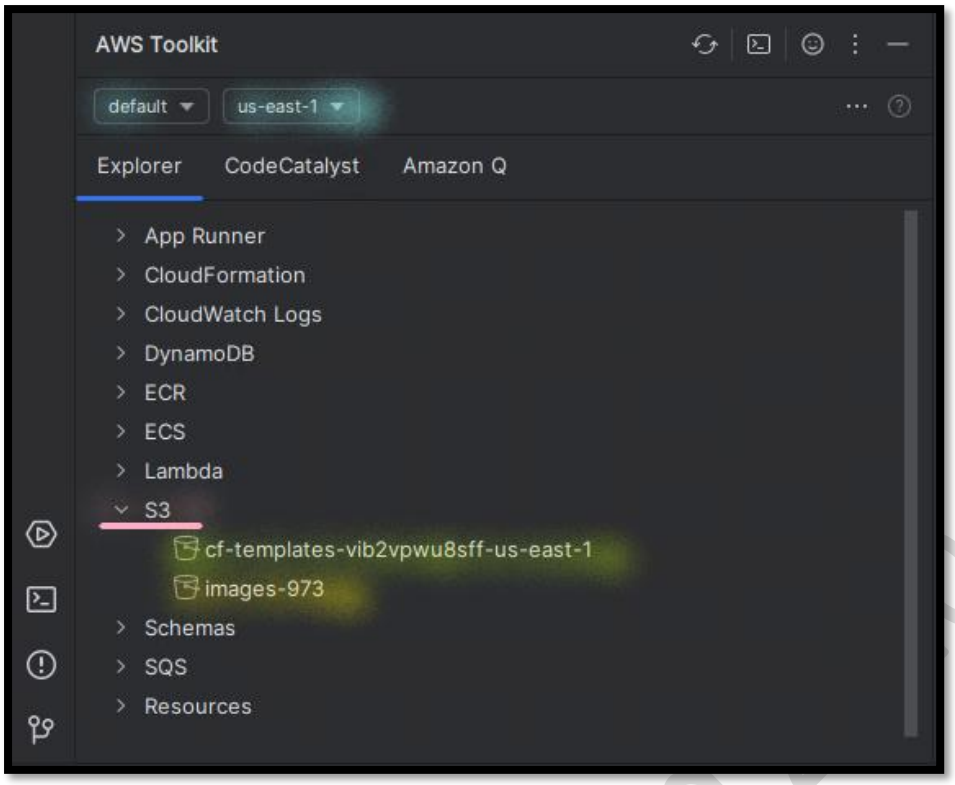
b) Expand **S3**.

Note: You can see the **Buckets**.

Note: You can see the **images** bucket.

Note: You can also see the **images** bucket and **another bucket** created via **clouformation**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.



Note: You can also create and upload new objects from AWS Toolkit.

Task 6: Close the Project

Step 1: Close the Maven Project

46. **From** the **IntelliJ IDE**.
    a. Select the **File**.
        i. Select **Close Project**.

Step 2: Delete the Stack

47. In the **AWS Management Console**, on the **Services** menu, search and select **CloudFormation**.
48. Select **Stack**.
    a. Select **Dev-Instance-JV**.
        i. Select **Delete**.
            i. Select **Delete stack**.

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.

Step 2: Delete the Buckets

49. In the **AWS Management Console**, on the **Services** menu, click **S3**.

50. Select **Buckets**.

Delete Images-123 Bucket

a. Select **images-123** bucket.

   i. Select **Empty**.

      a) **Type** **permanently delete** to delete all the objects.

      b) Select **Empty**.

      c) Select **Exit**.

b. Select **images-123** bucket.

   i. Select **Delete**.

      a) Type **images-123** bucket name to delete bucket.

      b) Select **Delete bucket**.

Delete Website-123 Bucket

c. Select **images-123** bucket.

   i. Select **Empty**.

      a) **Type** **permanently delete** to delete all the objects.

      b) Select **Empty**.

      c) Select **Exit**.

d. Select **images-123** bucket.

   i. Select **Delete**.

      a) Type **images-123** bucket name to delete bucket.

      b) Select **Delete bucket**.

Ahmad Majeed Zahoory

© No part of this manual, may be reproduced in whole or in part in any manner without the permission of the copyright owner.