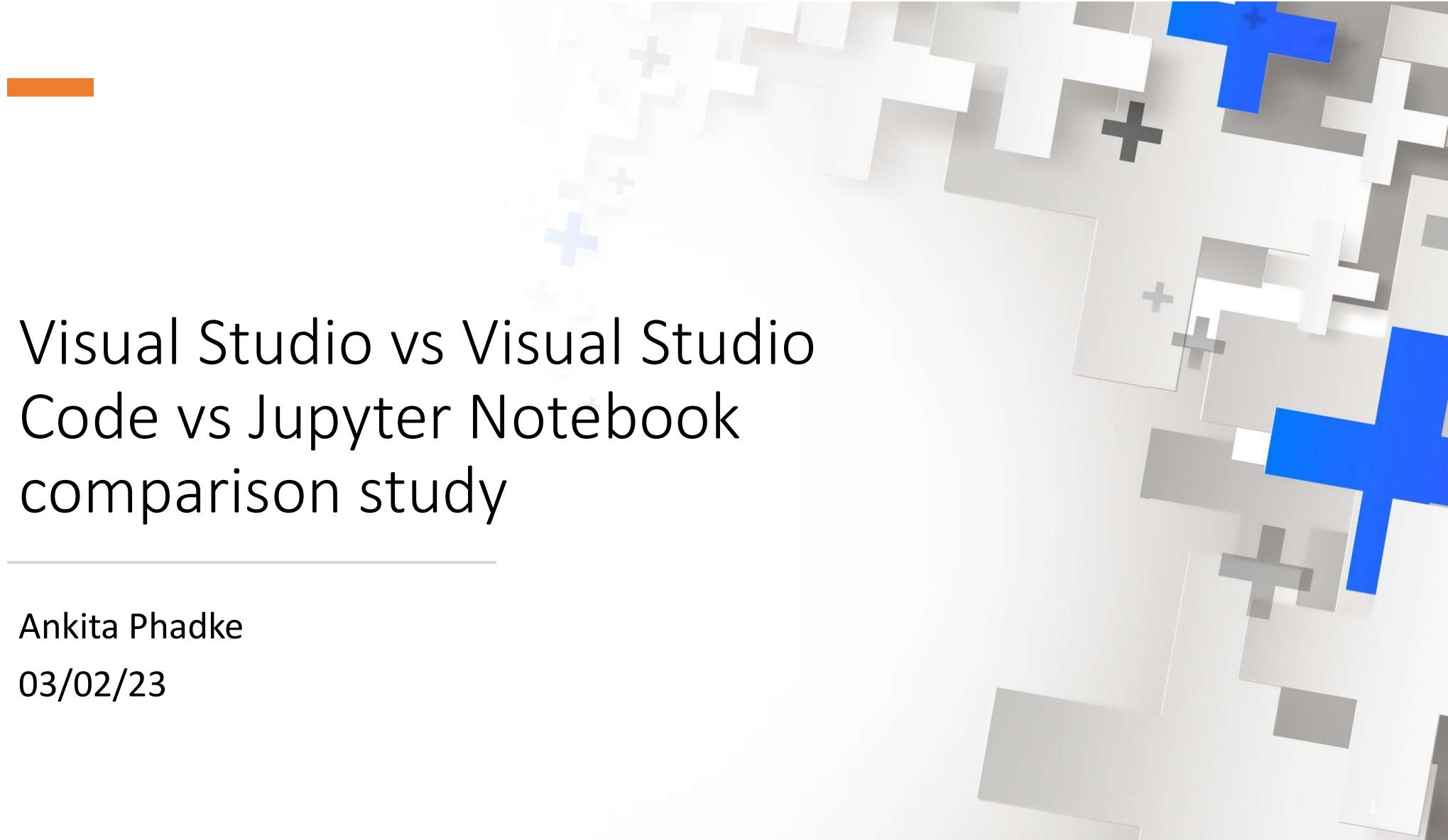




Visual Studio vs Visual Studio Code vs Jupyter Notebook comparison study

Ankita Phadke

03/02/23



Contents

Comparison based on Github options

Comparison based on AWS options

Comparison based on general points

Visual Studio vs. Visual Studio Code comparison

AWS connect with Visual Studio

Github connect with Visual Studio

AWS connect with Visual Studio Code

Github connect with Visual Studio Code

AWS connect with Jupyter Notebook

Github connect with Jupyter Notebook

DynamoDB connect with Jupyter Notebook

Conclusion

Comparison based on Github options



	Visual Studio	Visual Studio Code	Jupyter Notebook
Can they connect to github	Yes, steps are outlined in later slides	Yes, steps are outlined in later slides	Yes, steps are outlined in later slides (Can be directly done through AWS Management Console)
Can they do edit/test code before committing to github	Yes, Visual Studio has ability to setup Python projects and run Python code	Yes, the code can be run and tested in Visual Code	Yes, the code can be edited/tested in Jupyter Notebook

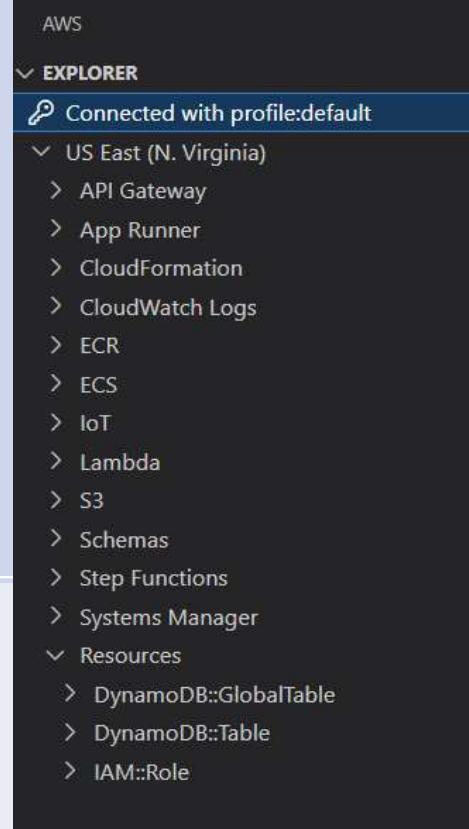


Comparison based on AWS options



	Visual Studio	
Can they connect to AWS through Python code/AWS CLI	<ul style="list-style-type: none">• No separate option available for AWS CLI through Visual Studio• IAM credentials can be used for logging into the toolkit and setup• All the AWS apps can be accessed directly through AWS Toolkit – this is available only on Windows, not available on MAC yet (https://aws.amazon.com/visualstudio/) (details on next slides)	A screenshot of the AWS Explorer in Visual Studio. It shows a list of AWS services on the right side, each with a small icon. The services listed are: Amazon CloudFront, Amazon CloudWatch, Amazon CodeArtifact, Amazon DynamoDB, Amazon EC2, Amazon Elastic Container Service, Amazon RDS, Amazon S3, Amazon SimpleDB, Amazon SNS, Amazon SQS, Amazon VPC, AWS CloudFormation, AWS Elastic Beanstalk, AWS Identity and Access Management, and AWS Lambda. At the top, there are dropdown menus for 'Credentials' set to 'Profile:IMADemo1' and 'Region' set to 'US East (N. Virginia)'. There are also icons for adding, editing, and deleting items.
AWS DynamoDB support	All operations are available as part of the AWS Toolkit, can create, edit, change tables (details on next slides)	

Comparison based on AWS options

	Visual Studio Code	
Can they connect to AWS through Python code/AWS CLI	<ul style="list-style-type: none">Plugin available for AWS CLI through Visual Studio CodeIAM credentials can be used for logging into the toolkit and setupAWS apps can be accessed directly through AWS Toolkit – this is available on all the OS: Windows, Linux and MAC OS (https://aws.amazon.com/visualstudiocode/) (details on next slides)Supports Python SDK (https://docs.aws.amazon.com/toolkit-for-vscode/latest/userguide/setup-toolkit.html)	
AWS DynamoDB support	<ul style="list-style-type: none">Available in preview/read-only modeUnable to make changes to the tables/create tables	

Comparison based on AWS options

	Jupyter Notebook
Can they connect to AWS through Python code/AWS CLI	<ul style="list-style-type: none">• Command Line Interface is available in Jupyter Notebook• AWS connection to Jupyter Notebook can be done through AWS Sagemaker Studio Notebook• Provides one click Jupyter Notebook deployment• Comes with Pre-installed Sagemaker Python SDK (https://docs.aws.amazon.com/sagemaker/latest/dg/ex1-prepare.html) <p>(details on next slides)</p>
AWS DynamoDB support	<ul style="list-style-type: none">• AWS DynamoDB can be accessed through Jupyter Notebook available through Amazon Sagemaker Studio• All operations on DynamoDB tables can be done through Python scripts/boto3 <p>(details on next slides)</p>

Comparison based on general points

	Visual Studio	Visual Studio Code	Jupyter Notebook
Which are more commonly used by the industry (rank if you can)	More suitable for debugging/application development	Most used in data science application, has in-built Jupyter Notebook support (https://code.visualstudio.com/docs/datascience/jupyter-notebooks)	Better than Visual Studio code for data science/machine learning operations
Which are more commonly used for CI/CD (rank if you can)	In-built Azure DevOps services support, can be more suitable for CI/CD operations https://learn.microsoft.com/en-us/azure/devops/organizations/accounts/set-up-vs?view=azure-devops	Azure tools plugin/extensions available https://learn.microsoft.com/en-us/previous-versions/azure/devops/all/java/vscode-extension?view=tfs-2017	CI/CD operations for DevOps can be done through Amazon Sagemaker and interaction with other services https://guyernest.medium.com/using-jupyter-notebooks-as-your-cloud-ide-for-devops-a49c03e5cb3f
Can they be used both on PC and Mac/Apple	Available on Windows and MAC OS (not on Linux)	Available on Windows, Linux and MAC OS	Available on Windows, Linux and MAC OS

Comparison based on general points

	Visual Studio	Visual Studio Code	Jupyter Notebook
Fees – is there any fees?	Free download available for Visual Studio Community version	Completely free of cost for all OS	Completely free of cost for all OS
Pros and cons	<p>Pro:</p> <ul style="list-style-type: none">• Cloud connectivity• Feature-rich• Github support <p>Cons:</p> <ul style="list-style-type: none">• Slower at times• Version updates needed, older versions work slower (https://visualstudio.microsoft.com/vs/professional/)	<p>Pro:</p> <ul style="list-style-type: none">• Free editor• Supports all languages• Github support <p>Cons:</p> <ul style="list-style-type: none">• Needs extra plugins support• Comes with only base functions (https://code.visualstudio.com/docs/editor/whyvscode)	<p>Pro:</p> <ul style="list-style-type: none">• Easy to use• Lightweight• Keeps the data secured• Github support <p>Cons:</p> <ul style="list-style-type: none">• Can't work easily if multiple developers are collaborating <p>(https://betterprogramming.pub/pros-and-cons-for-jupyter-notebooks-as-your-editor-for-data-science-work-tip-pycharm-is-probably-40e88f7827cb)</p>

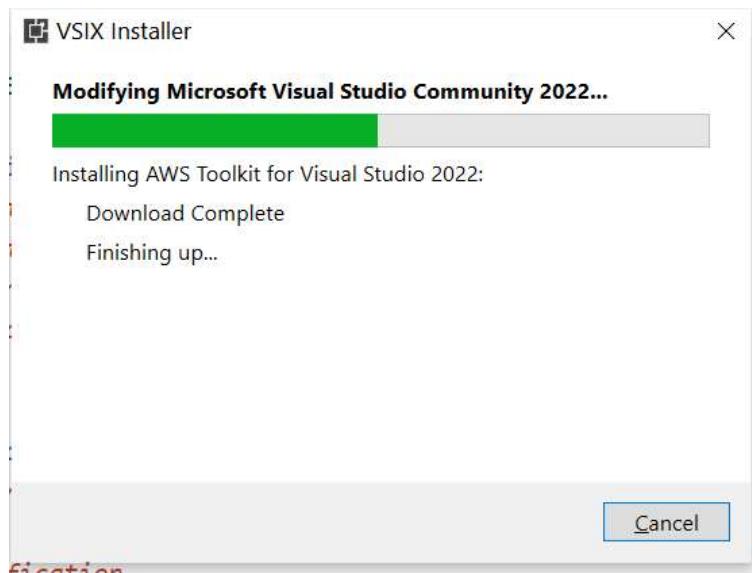
Visual Studio vs. Visual Studio Code comparison

Visual Studio	Visual Studio Code
Product by Microsoft	Product by Microsoft
Complete IDE suite	More of a code editor with many languages support
More useful for collaborative inputs from multiple developers	More popular in data science community
Better in-built functionalities	Provides better plugins support
Sometimes works slower, heavier software, feature-rich	Faster than Visual Studio
Paid versions are expensive	Provides better pricing options

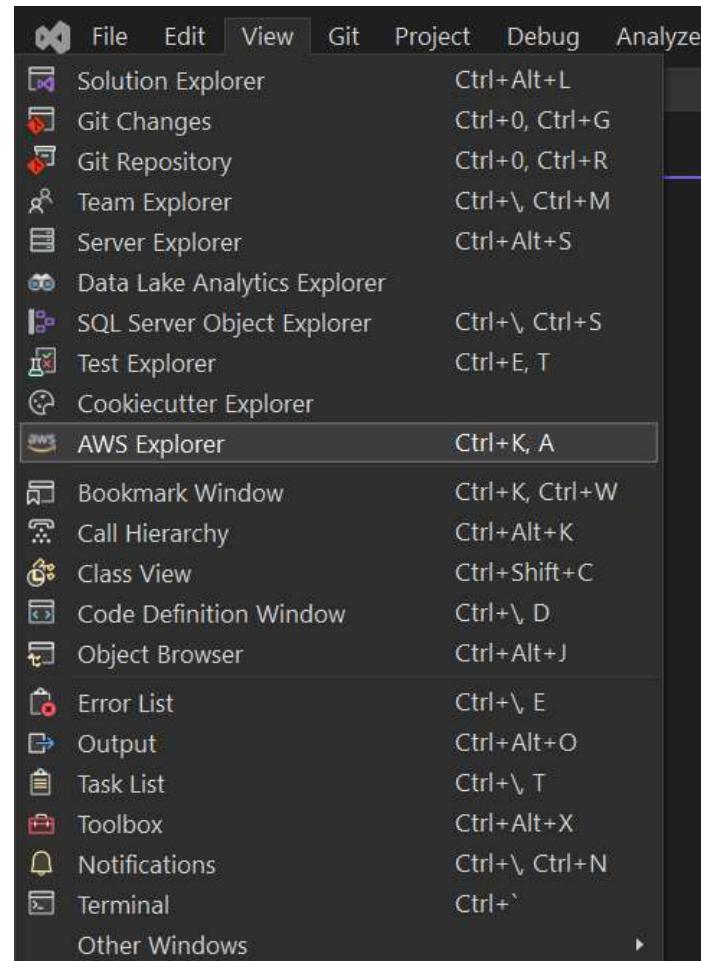
References:

- <https://www.infoworld.com/article/3436860/visual-studio-vs-visual-studio-code-how-to-choose.html>
- <https://www.turing.com/kb/ultimate-guide-visual-studio-vs-visual-studio-code>

AWS connect with Visual Studio

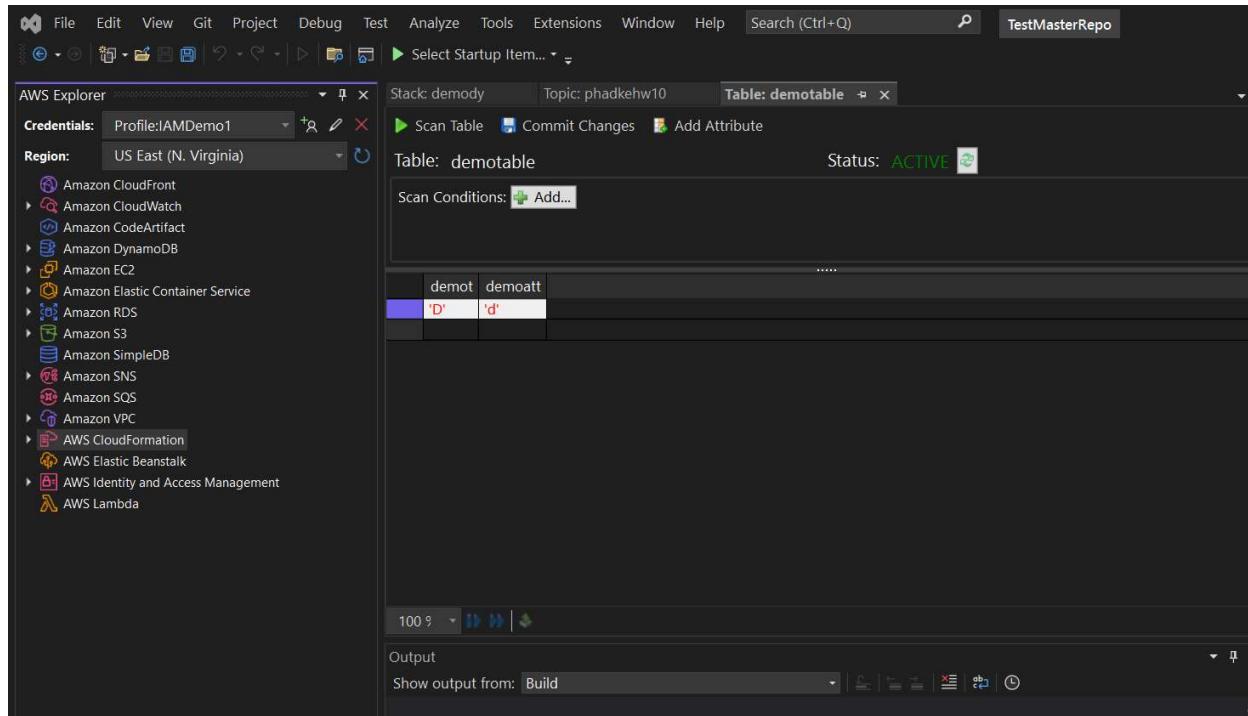


1. Install AWS Toolkit from Marketplace
2. AWS Explorer is added to the menu
3. AWS functions are added to the Explorer and available to use

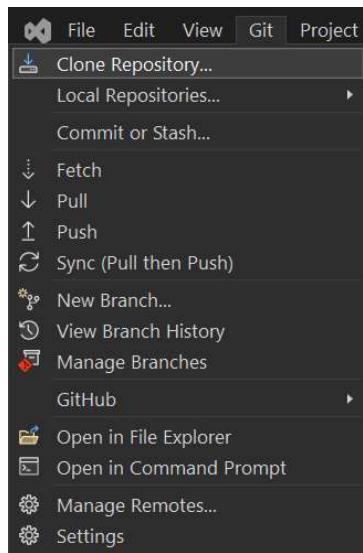


AWS DynamoDB with Visual Studio

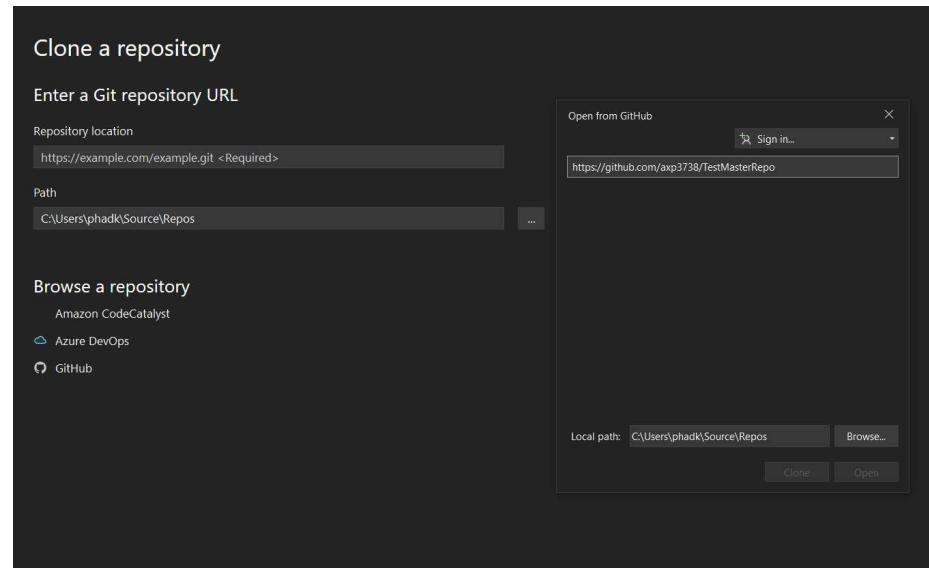
DynamoDB is available in editable mode, and changes committed are reflected in DynamoDB
E.g. demotable, below



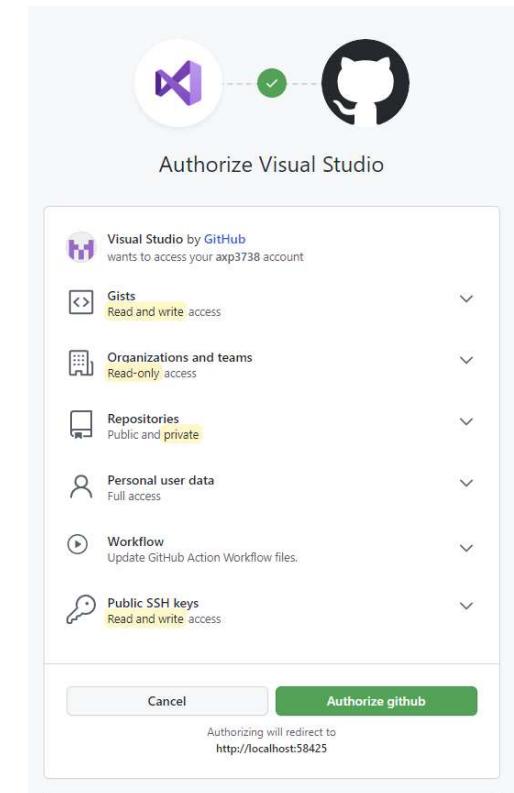
Github connect with Visual Studio



1. Menu: Git -> Clone Repository

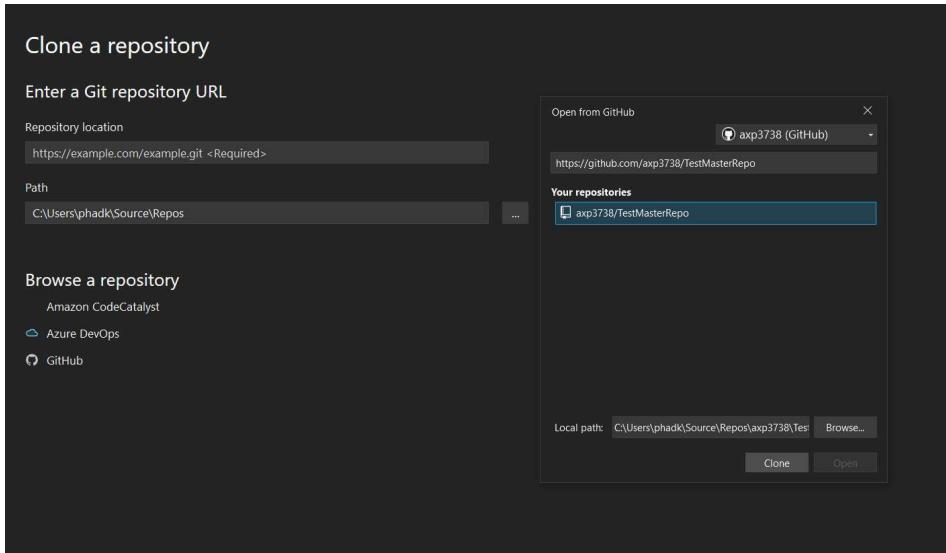


2. Enter repository URL

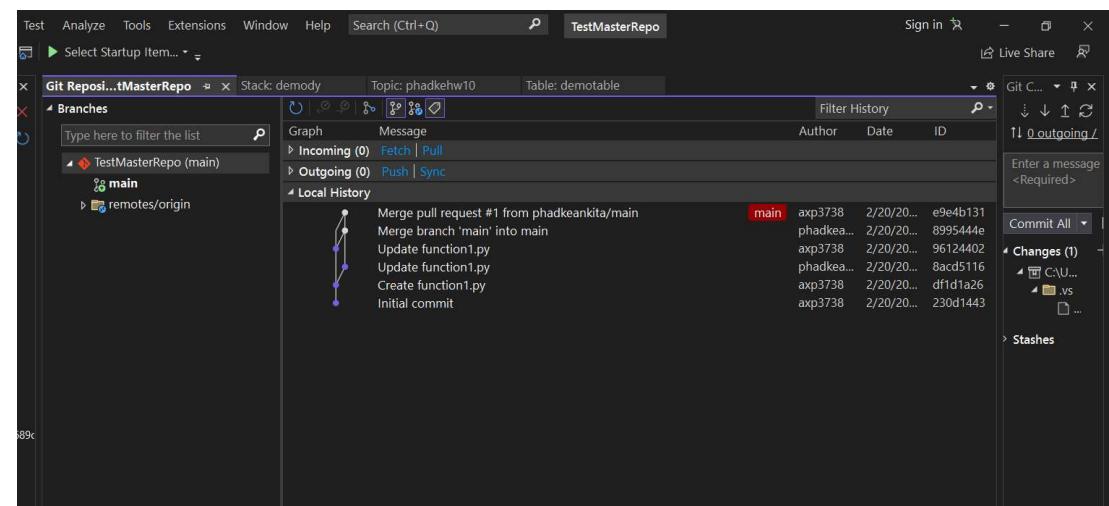


3. Authorize Visual Studio

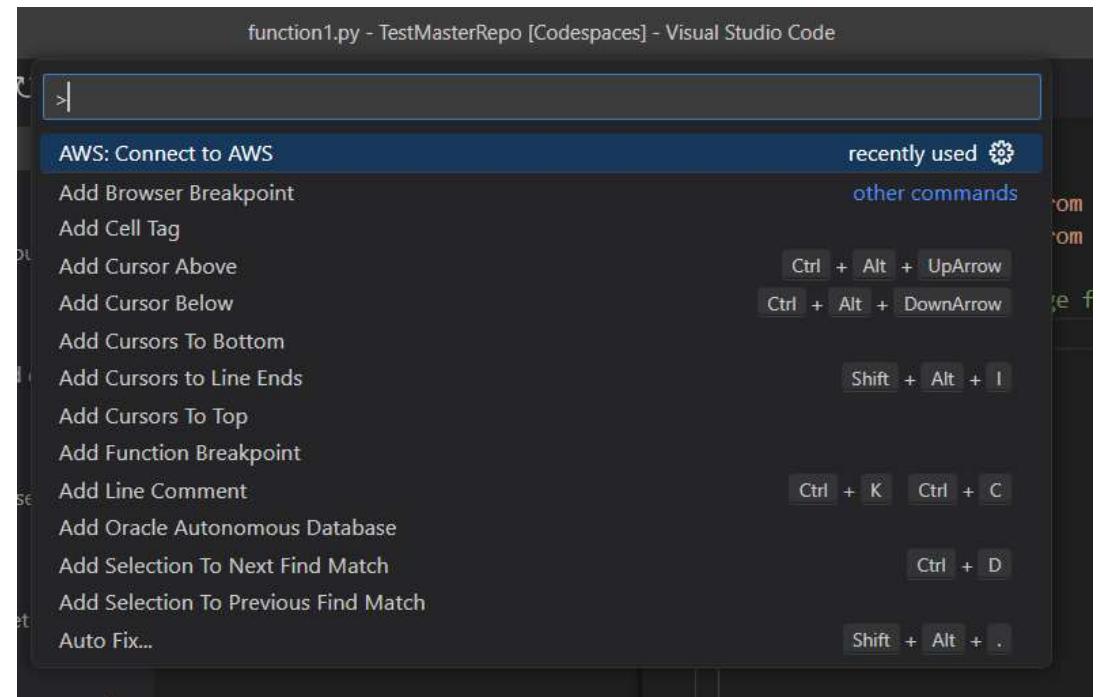
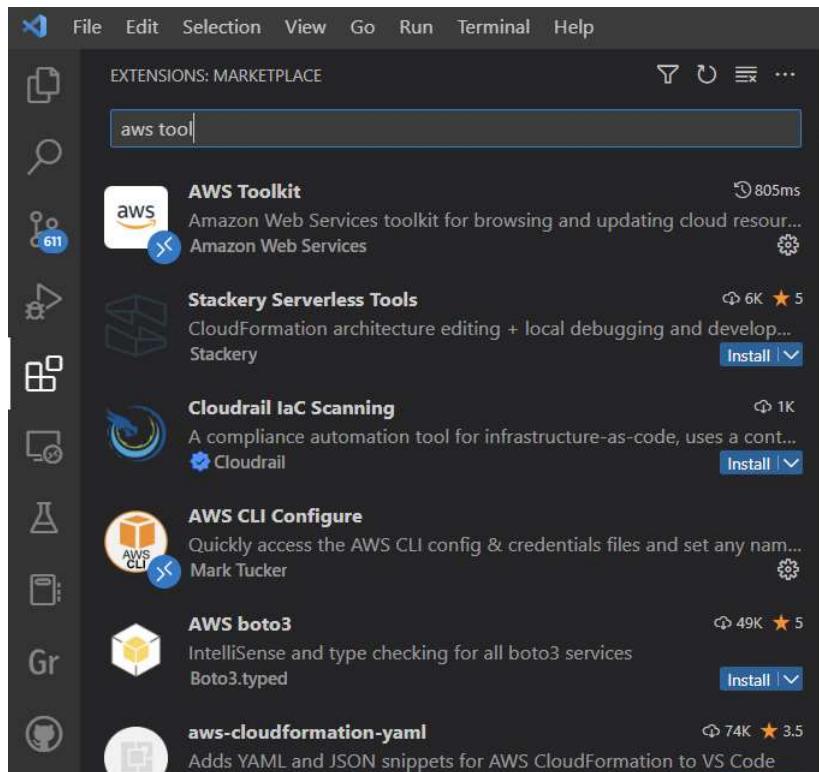
Github connect with Visual Studio



4. Create local repository



AWS connect with Visual Studio Code

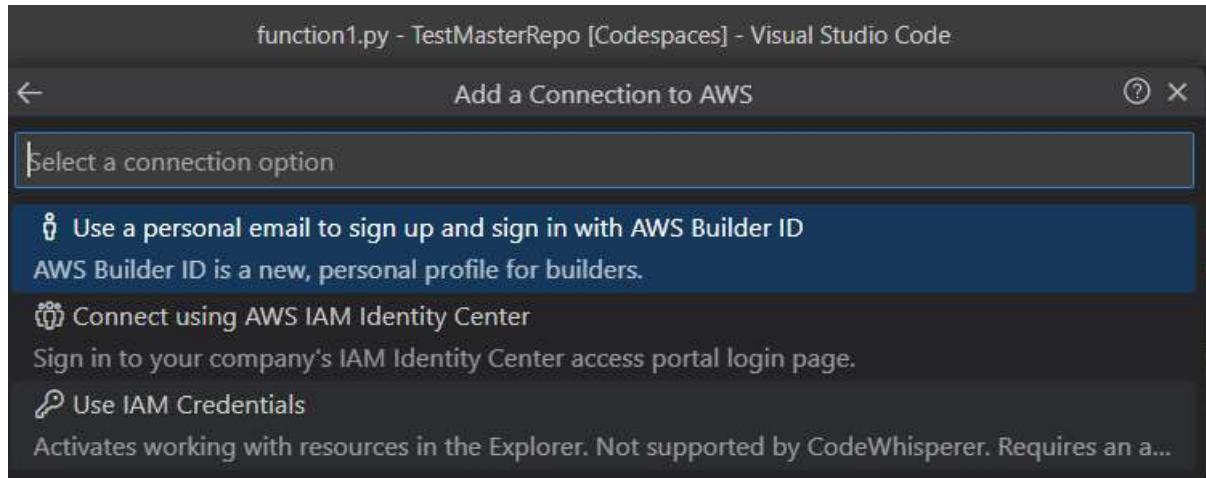


2. Open Connect to AWS option to configure the login

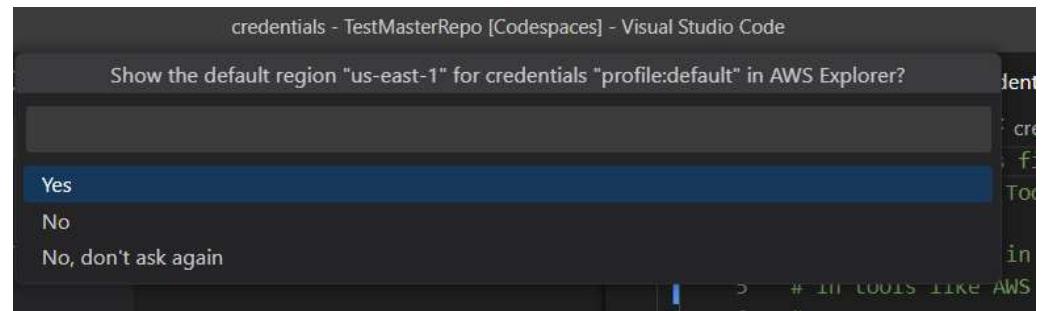
1. Install AWS Toolkit extension from Marketplace

14

AWS connect with Visual Studio Code

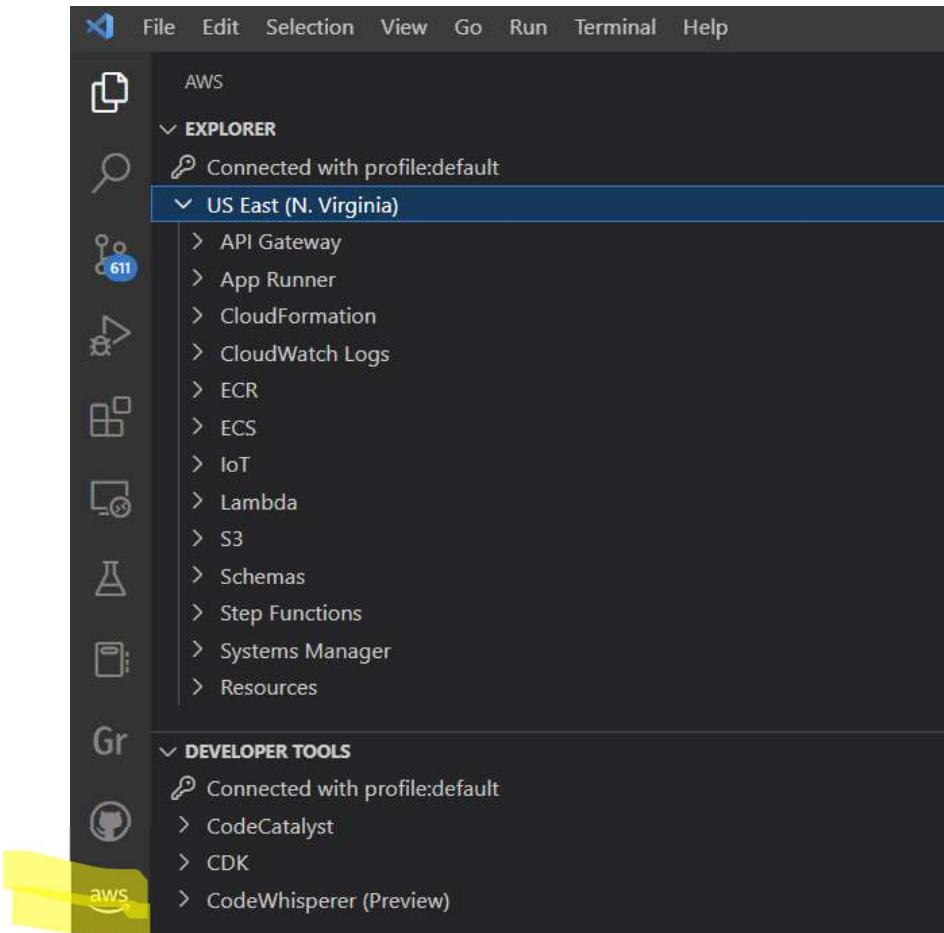


3. Select the option 'Use IAM Credentials'



4. Put the access key & secret access key and continue

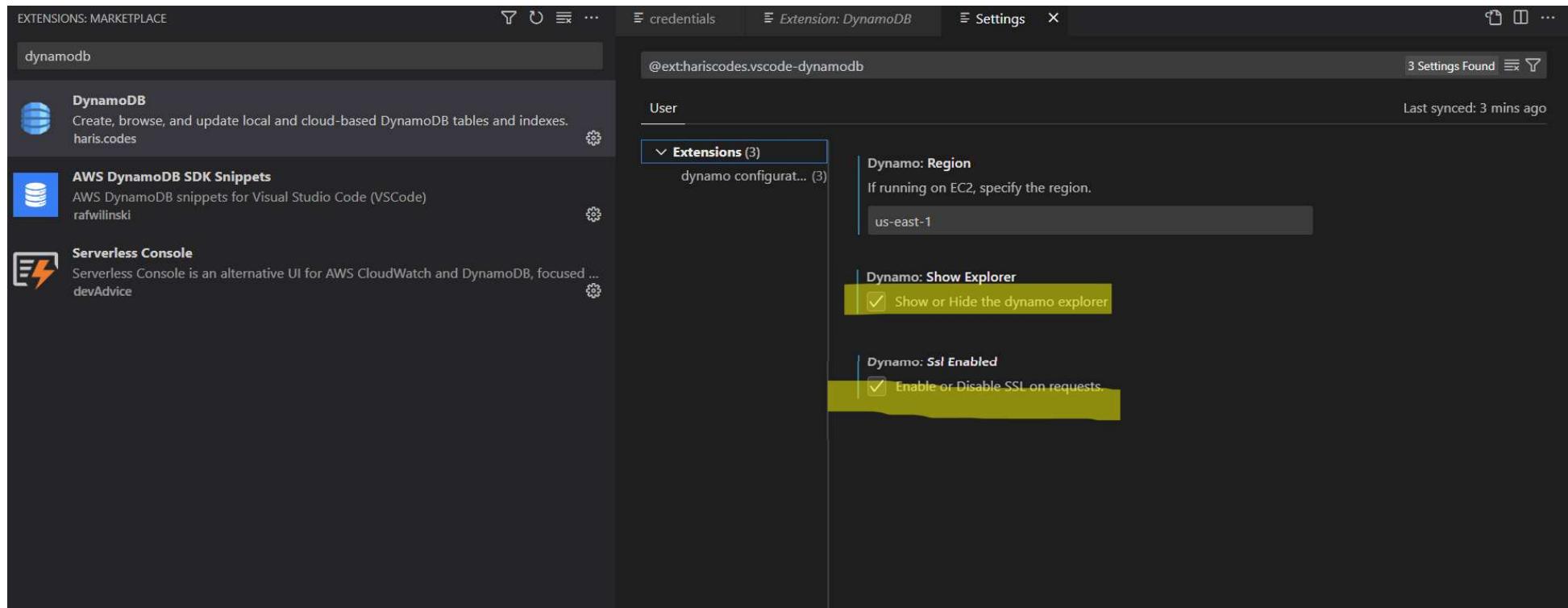
AWS connect with Visual Studio Code



5. AWS options are added in Visual Studio Code, through AWS Toolkit

AWS DynamoDB with Visual Studio Code

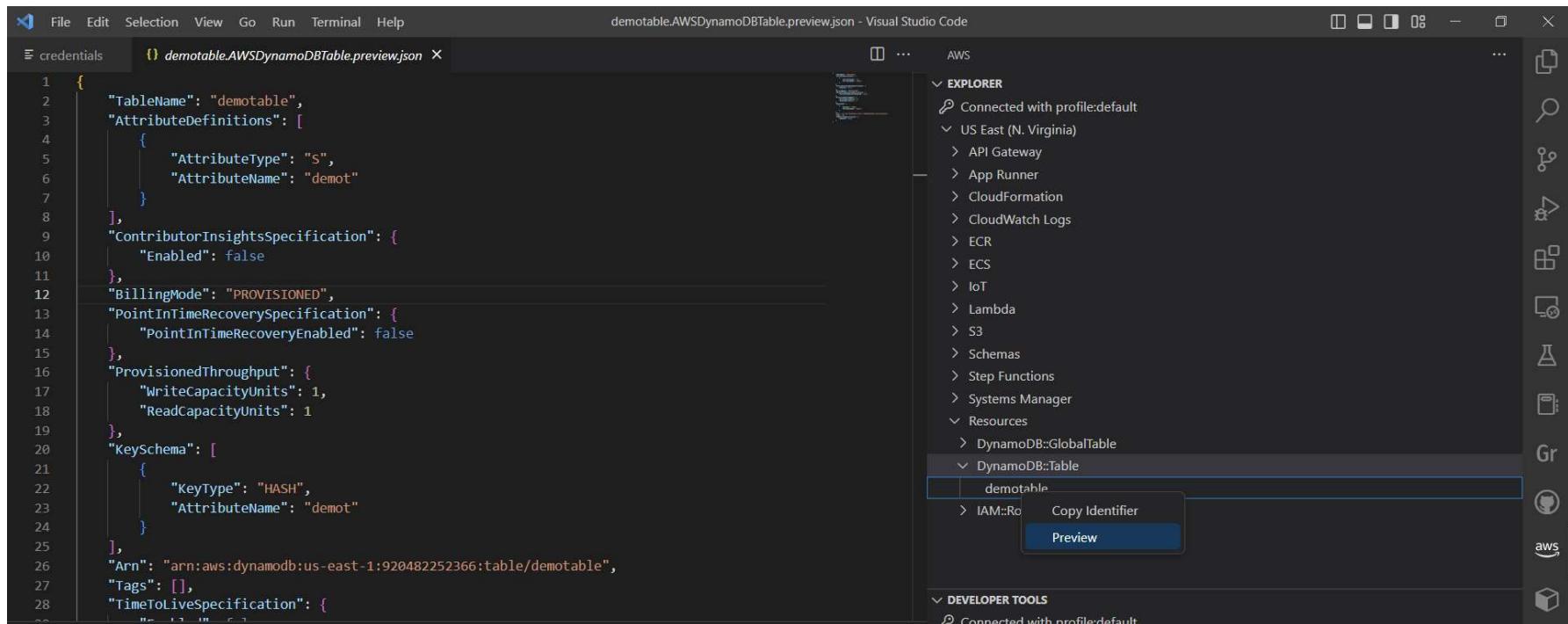
DynamoDB related options can be enabled through the relevant plugins, add those and enable the options



AWS DynamoDB with Visual Studio Code

DynamoDB option is only available in view-only mode (preview only)

Changes couldn't be done to the DynamoDB tables, or new tables couldn't be created



The screenshot shows the Visual Studio Code interface with the AWS extension installed. The left side features a code editor with a JSON file named 'demutable.AWDynamoDBTable.preview.json'. The file contains configuration for a DynamoDB table named 'demutable'. The right side shows the AWS Explorer sidebar, which is connected to the 'US East (N. Virginia)' region. Under the 'DynamoDB:Table' section, the 'demutable' table is listed. A context menu is open over this table entry, with the 'Preview' option highlighted.

```
1 {  
2     "TableName": "demutable",  
3     "AttributeDefinitions": [  
4         {  
5             "AttributeType": "S",  
6             "AttributeName": "demot"  
7         }  
8     ],  
9     "ContributorInsightsSpecification": {  
10        "Enabled": false  
11    },  
12    "BillingMode": "PROVISIONED",  
13    "PointInTimeRecoverySpecification": {  
14        "PointInTimeRecoveryEnabled": false  
15    },  
16    "ProvisionedThroughput": {  
17        "WriteCapacityUnits": 1,  
18        "ReadCapacityUnits": 1  
19    },  
20    "KeySchema": [  
21        {  
22            "KeyType": "HASH",  
23            "AttributeName": "demot"  
24        }  
25    ],  
26    "Arn": "arn:aws:dynamodb:us-east-1:920482252366:table/demutable",  
27    "Tags": [],  
28    "TimeToLiveSpecification": {  
29        "TimeToLive": null  
30    }  
31}
```

AWS website lists VS Code as a typically used Toolkit for Python

- <https://aws.amazon.com/developer/language/python/>
- <https://aws.amazon.com/visualstudiocode/>

Tools

Download the tools needed to run Python applications on AWS



SDK for Python

Simplifies use of AWS services by providing a set of libraries that are consistent and familiar for Python developers.

[Download SDK »](#)



AWS IDE Toolkits

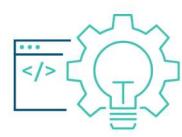
Use popular Integrated Development Environments (IDEs) to author, debug, and deploy your code on AWS.

[Get the PyCharm Toolkit »](#)

[Get the IntelliJ Toolkit »](#)

[Get the VS Code Toolkit »](#)

[Get Amazon CodeWhisperer »](#)



AWS CDK for Python

Use the AWS Cloud Development Kit (CDK) for your Infrastructure as Code with Python.

[Download CDK »](#)

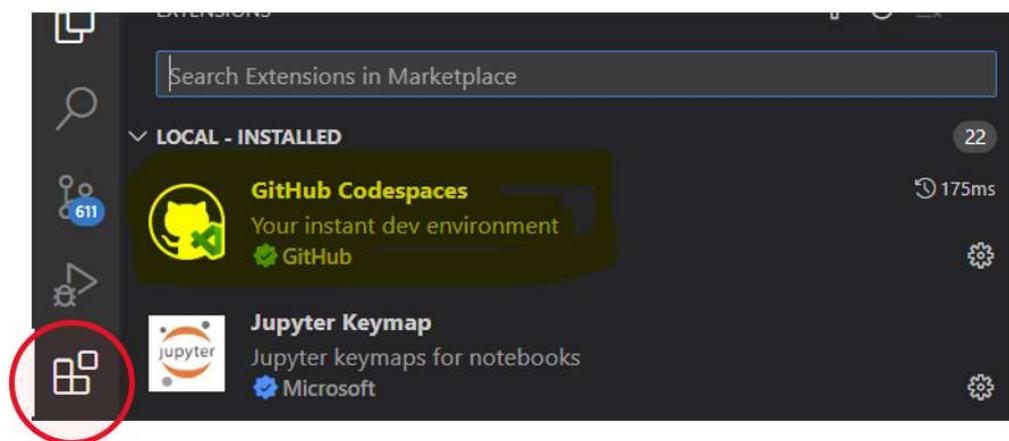


AWS IoT SDK for Python

Access AWS IoT using MQTT or MQTT over the WebSocket protocol from Python.

[Clone on Github »](#)

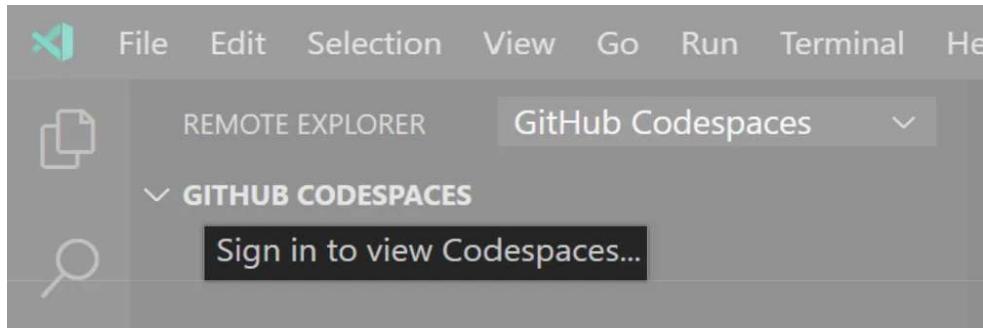
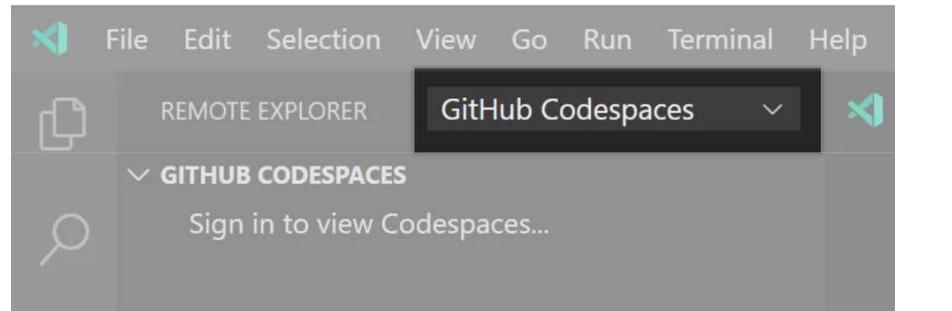
Github connect with Visual Studio Code



1. Download visual studio code, install codespace

2. Above symbol is added in visual studio code's left bar menu:
'Remote Explorer'

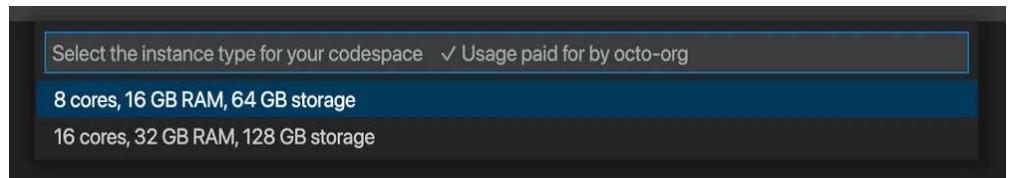
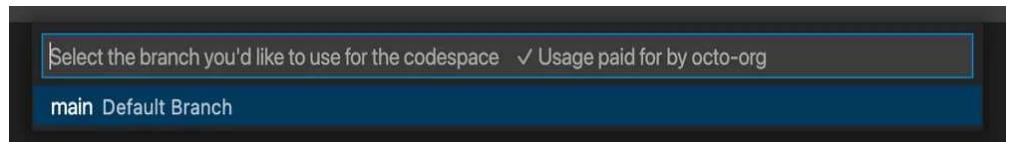
Github connect with Visual Studio Code



3. Click on remote explorer and go to codespaces option and try to select GitHub branch of interest

4. To authorize VS Code to access account on GitHub, click Allow

5. Back in Visual Studio Code, select main branch and storage options as below



Github connect with Visual Studio Code

6. Files present in your GitHub repository can be seen in Visual Studio Code

The screenshot shows the Visual Studio Code interface with a GitHub Codespace open. The left sidebar displays the 'REMOTE EXPLORER' and 'GITHUB CODESPACES' sections, which include details about the repository 'axp3738/TestMasterRepo'. The main workspace contains two code editors showing the file 'function1.py'. Both editors show the same Python code:

```
function1.py - TestMasterRepo [Codespaces] - Visual Studio Code
function1.py
function1.py > message
function1.py > message
def message():
    print('Hello from phadkeanita')
    print('Hello from axp3738')
# call the message function
message()

function1.py
function1.py > message
function1.py > message
def message():
    print('Hello from phadkeanita')
    print('Hello from axp3738')
# call the message function
message()
```

The bottom right corner of the terminal window shows the number '22'.

AWS connect with Jupyter Notebook

Can be done using Sagemake studio

(<https://docs.aws.amazon.com/sagemaker/latest/dg/ex1-prepare.html>)

The screenshot shows the AWS Management Console with the 'Amazon SageMaker' service selected. The main content area displays the 'MACHINE LEARNING' section for Amazon SageMaker, featuring a large heading 'Amazon SageMaker' and the subtext 'Build, train, and deploy machine learning models at scale'. Below this, there's a 'New to SageMaker?' section with a 'Get Started' button, and a 'Documentation' sidebar with links like 'Getting started', 'Tutorials', and 'Developer Resources'. On the left, a navigation pane lists options such as 'Getting started', 'Studio', 'Canvas', 'RStudio', 'Domains', 'SageMaker dashboard', 'JumpStart' (which is expanded to show 'Foundation models', 'Computer vision models', and 'Natural language processing models'), and 'Search'.

1. AWS Management Console -> Open Amazon Sagemaker

This screenshot shows the same Amazon SageMaker service page as above, but with the 'Notebook instances' link under the 'Documentation' sidebar highlighted in blue, indicating it has been clicked. The rest of the interface remains the same, showing the 'JumpStart' section expanded to include 'Foundation models' (marked as NEW), 'Computer vision models', and 'Natural language processing models'.

2. Open Notebook option &
Navigate to Notebook instances

AWS connect with Jupyter Notebook

The screenshot shows the 'Create notebook instance' page in the AWS SageMaker console. At the top, there's a breadcrumb navigation: 'Amazon SageMaker > Notebook instances > Create notebook instance'. Below it, the main title is 'Create notebook instance'. A descriptive text states: 'Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises.' A 'Learn more' link is provided. The 'Notebook instance settings' section contains the following fields:

- Notebook instance name:** DemoJupyter
- Notebook instance type:** ml.t3.medium
- Elastic Inference:** Learn more
- Platform identifier:** Learn more
- Additional configuration:** (button)

The screenshot shows the 'Permissions and encryption' configuration page in the AWS SageMaker console. The title is 'Permissions and encryption'. It includes the following sections:

- IAM role:** A dropdown menu for 'Enter a custom IAM role ARN' is open, showing 'arn:aws:iam::920482252366:user/IAMDemo1'.
- Custom IAM role ARN:** A text input field containing 'arn:aws:iam::920482252366:user/IAMDemo1'.
- Create role using the role creation wizard:** A button with a checkmark icon.
- Root access - optional:**
 - Enable - Give users root access to the notebook
 - Disable - Don't give users root access to the notebook
Lifecycle configurations always have root access
- Encryption key - optional:** A dropdown menu set to 'No Custom Encryption'.
- Network - optional:** (button)
- Git repositories - optional:** (button)

3. Select Create Notebook instance
(IAM user linked should have AmazonSageMakerFullAccess IAM policy attached.)
(Selected option to let AWS create the custom IAM role/ARN)

AWS connect with Jupyter Notebook

The screenshot shows the 'Permissions and encryption' step of creating a new notebook instance. It includes fields for selecting an IAM role (AmazonSageMaker-ExecutionRole-20230301T175899), enabling root access (selected), and choosing an encryption key (No Custom Encryption). Below these are sections for Network, Git repositories, and Tags.

IAM role
Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the [AmazonSageMakerFullAccess](#) IAM policy attached.
AmazonSageMaker-ExecutionRole-20230301T175899

Success! You created an IAM role.
AmazonSageMaker-ExecutionRole-20230301T175899

Create role using the role creation wizard [\[View\]](#)

Root access - optional

Enable - Give users root access to the notebook
 Disable - Don't give users root access to the notebook
Lifecycle configurations always have root access

Encryption key - optional
Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.
No Custom Encryption

Network - optional

Git repositories - optional

Tags - optional

The screenshot shows the 'Notebook instances' page after creation. A success message indicates the instance is being created. The table lists the newly created instance: Name: DemoJupyter, Instance: ml.t3.medium, Creation time: Mar 01, 2023 23:59 UTC, Status: Pending.

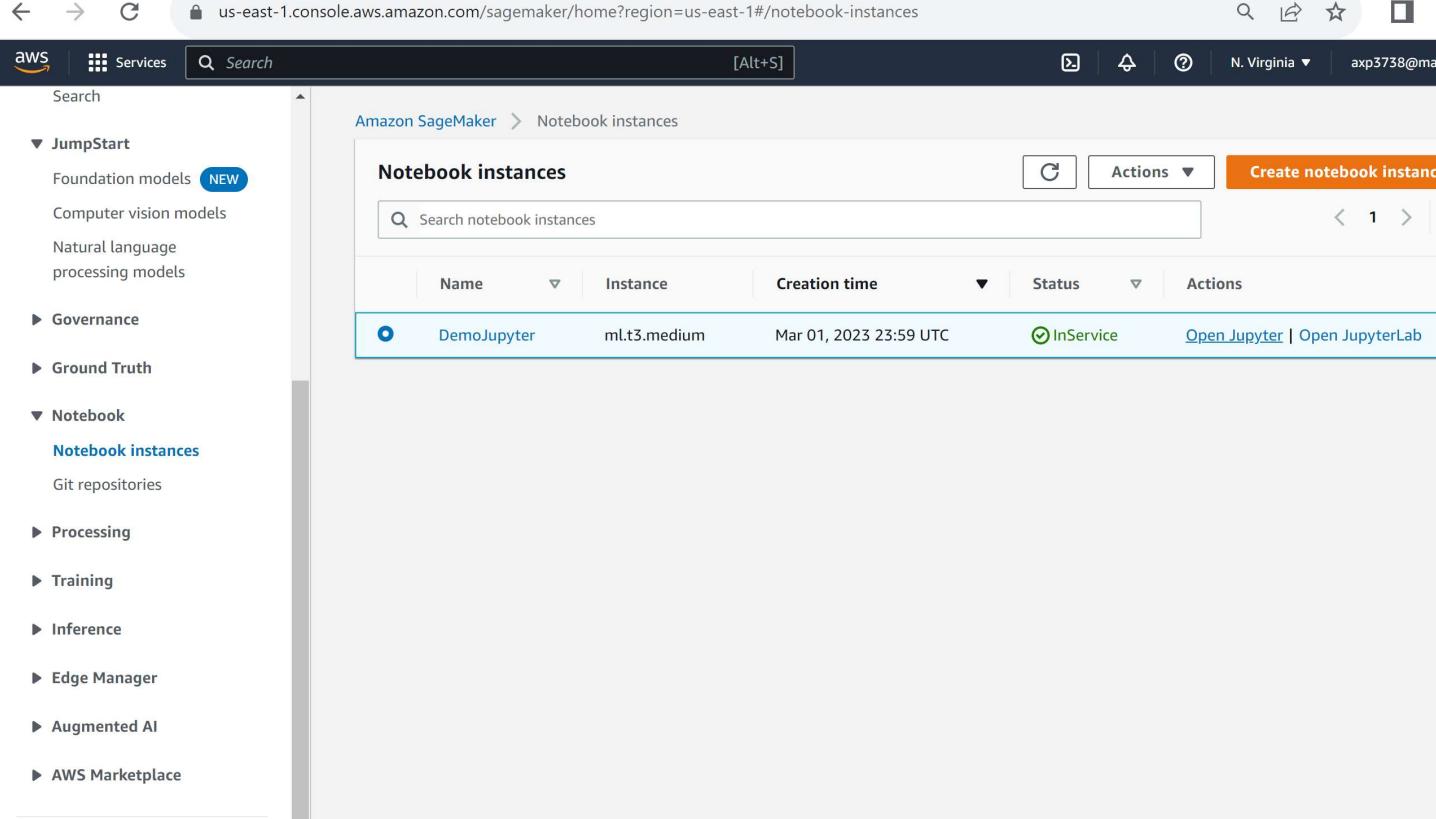
Notebook instances

Success! Your notebook instance is being created.
Open the notebook instance when status is InService and open a template notebook to get started.

Name	Instance	Creation time	Status	Actions
DemoJupyter	ml.t3.medium	Mar 01, 2023 23:59 UTC	Pending	-

4. AWS creates the custom IAM role, and the notebook instance as required

AWS connect with Jupyter Notebook

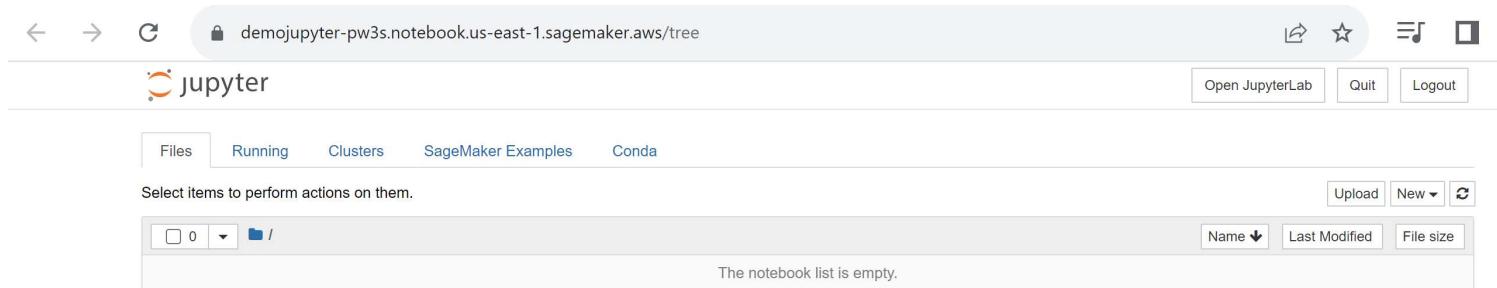


The screenshot shows the AWS SageMaker console interface. The left sidebar contains a navigation menu with sections like JumpStart, Governance, Ground Truth, Notebook, Processing, Training, Inference, Edge Manager, Augmented AI, and AWS Marketplace. The main content area is titled "Notebook instances" and displays a table with one row. The table columns are Name, Instance, Creation time, Status, and Actions. The single entry is "DemoJupyter" (ml.t3.medium), created on Mar 01, 2023 23:59 UTC, and is marked as "InService". The "Actions" column for this row contains links labeled "Open Jupyter" and "Open JupyterLab". The top navigation bar includes the AWS logo, services menu, search bar, and user information (N. Virginia, axp3738@mav).

Name	Instance	Creation time	Status	Actions
DemoJupyter	ml.t3.medium	Mar 01, 2023 23:59 UTC	InService	Open Jupyter Open JupyterLab

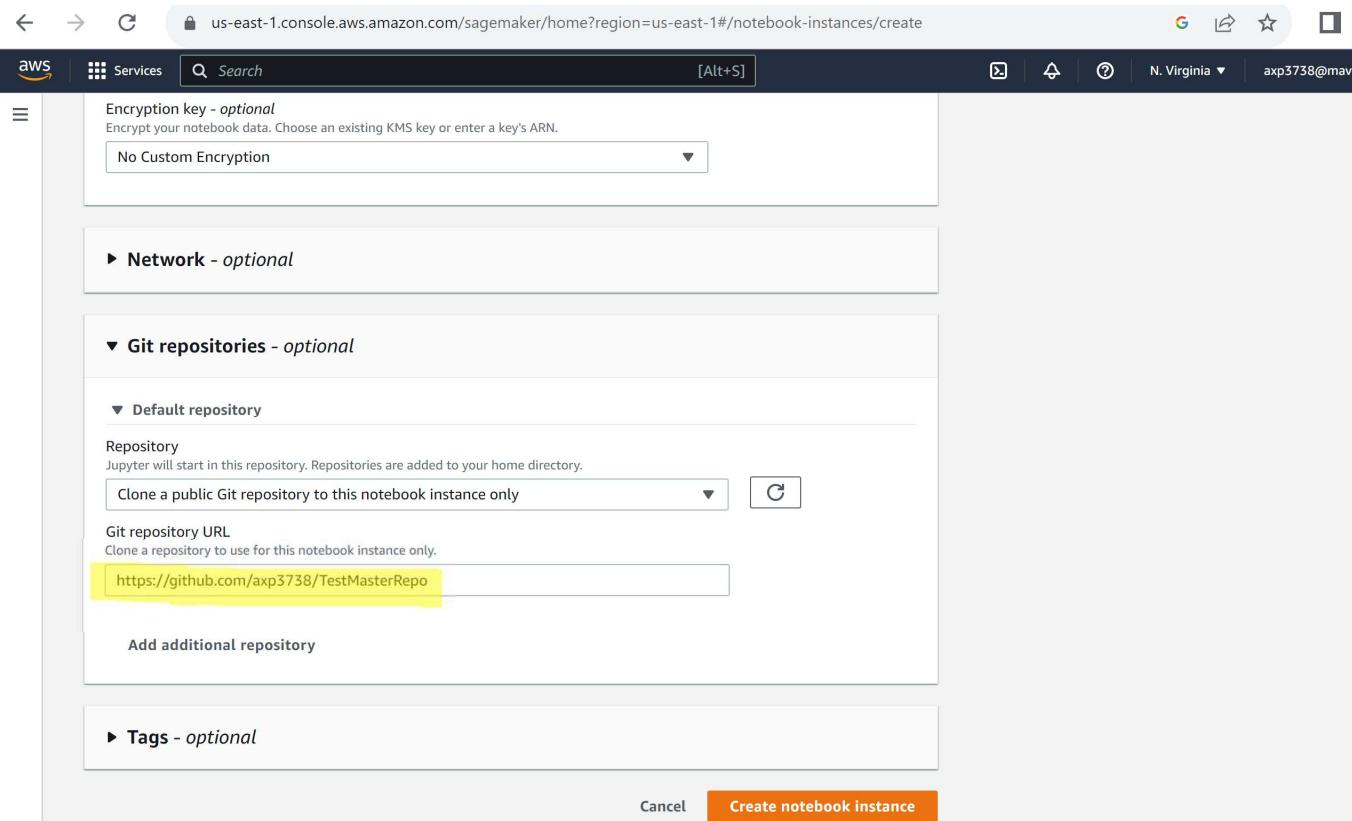
4. Notebook can be launched using Open Jupyter link, after creation

AWS connect with Jupyter Notebook



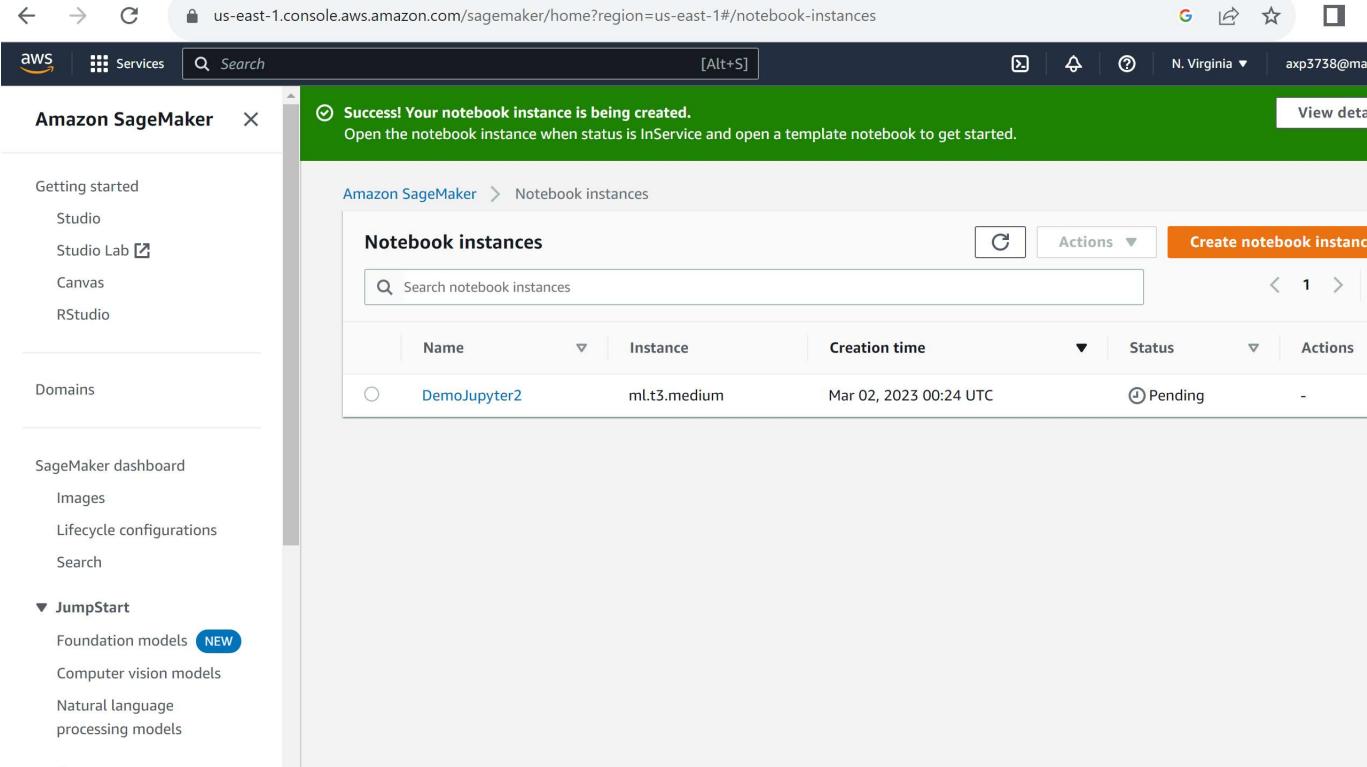
5. Notebook can be accessed as above

Github connect with Jupyter Notebook



1. Can be done using Amazon Sagemake studio, through AWS Management Console, while creating Jupyter Notebook: by adding the Github repository name

Github connect with Jupyter Notebook



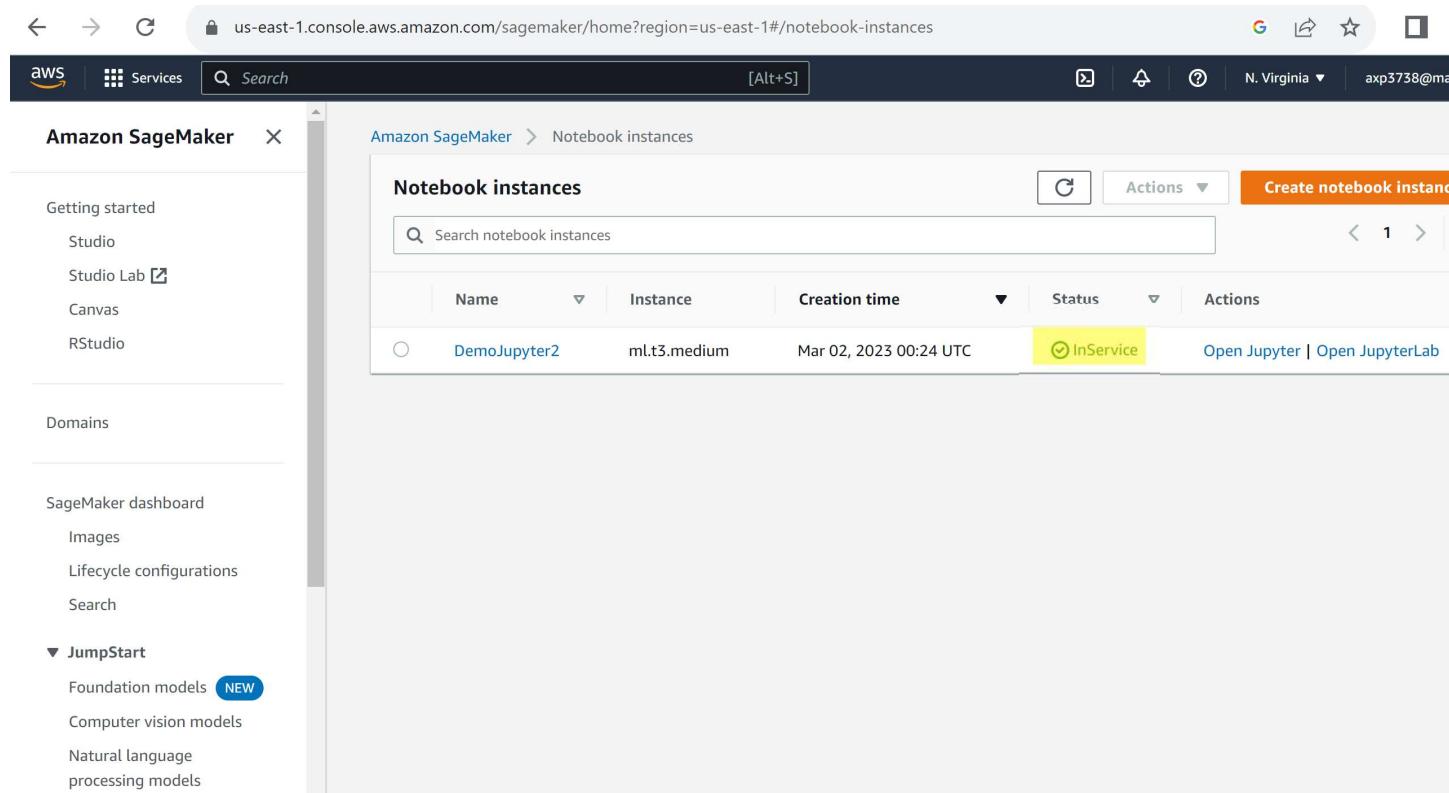
The screenshot shows the Amazon SageMaker console interface. At the top, there is a navigation bar with back, forward, and search icons, followed by the URL "us-east-1.console.aws.amazon.com/sagemaker/home?region=us-east-1#/notebook-instances". Below the URL is a header with AWS services, a search bar, and user information "N. Virginia" and "axp3738@mav". A green success message banner at the top right says "Success! Your notebook instance is being created. Open the notebook instance when status is InService and open a template notebook to get started." and includes a "View detail" link. On the left, a sidebar menu lists "Getting started" (Studio, Studio Lab, Canvas, RStudio), "Domains", "SageMaker dashboard" (Images, Lifecycle configurations, Search), and "JumpStart" (Foundation models, Computer vision models, Natural language processing models). The main content area is titled "Notebook instances" and shows a table with one row:

Name	Instance	Creation time	Status	Actions
DemoJupyter2	ml.t3.medium	Mar 02, 2023 00:24 UTC	Pending	[Actions]

A large orange "Create notebook instance" button is located at the top right of the table. To the right of the table, the number "2." is displayed.

2. Once instance is submitted, Notebook gets created

Github connect with Jupyter Notebook

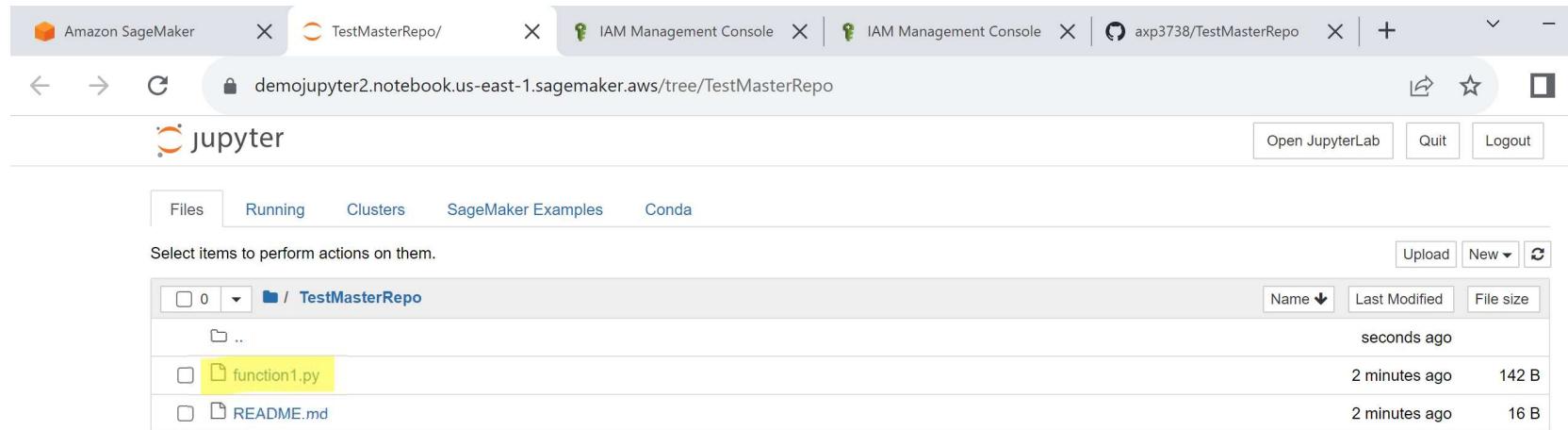


The screenshot shows the Amazon SageMaker console interface. On the left, there is a navigation sidebar with sections like 'Getting started', 'Studio', 'Studio Lab', 'Canvas', 'RStudio', 'Domains', 'SageMaker dashboard', 'Images', 'Lifecycle configurations', 'Search', and 'JumpStart' (which includes 'Foundation models' marked as NEW, 'Computer vision models', and 'Natural language processing models'). The main content area is titled 'Notebook instances' and shows a table with one row. The table columns are 'Name', 'Instance', 'Creation time', and 'Status'. The row contains 'DemoJupyter2', 'ml.t3.medium', 'Mar 02, 2023 00:24 UTC', and 'InService' (with a checkmark icon). To the right of the status, there are two buttons: 'Open Jupyter' and 'Open JupyterLab'. The top of the page has a search bar, a 'Create notebook instance' button, and various AWS navigation icons.

Name	Instance	Creation time	Status	Actions
DemoJupyter2	ml.t3.medium	Mar 02, 2023 00:24 UTC	InService	Open Jupyter Open JupyterLab

3. After the Notebook is ready, it can be accessed through Open Jupyter link

Github connect with Jupyter Notebook



4. The Notebook shows the code from chosen github repository

DynamoDB connect with Jupyter Notebook

1. Can be done through Amazon Sagemaker and Jupyter notebook
2. Open the IAM policies, select the policy to allow DynamoDB access, click on Attach

The screenshot shows the AWS IAM Policies page. On the left, there's a sidebar with navigation links like Dashboard, Access management, Policies (which is currently selected), and Access reports. The main area displays a table of policies. A search bar at the top right shows the filter "dynamodb". The table has columns for Policy name, Type, and Used as. One row is highlighted with a yellow background: "AmazonDynamoDBFullAccess" (Type: AWS managed, Used as: Permissions policy). To the right of the table is a context menu with options: Actions (with Attach highlighted in yellow), Detach, and Delete.

Policy name	Type	Used as
AmazonDynamoDBFullAccess	AWS managed	Permissions policy
AWSLambdaDynamoDBExecutionRole	AWS managed	None
AmazonDynamoDBReadOnlyAccess	AWS managed	None
DynamoDBReplicationServiceRolePolicy	AWS managed	None
AWSApplicationAutoscalingDynamoDBTablePolicy	AWS managed	Permissions policy
AWSLambdaInvocation-DynamoDB	AWS managed	None
DynamoDBKinesisReplicationServiceRolePolicy	AWS managed	None
DynamoDBCloudWatchContributorInsightsServiceRolePolicy	AWS managed	None

DynamoDB connect with Jupyter Notebook

3. Select the appropriate IAM user used in Amazon Sagemaker, on the next screen and click on Attach Policy

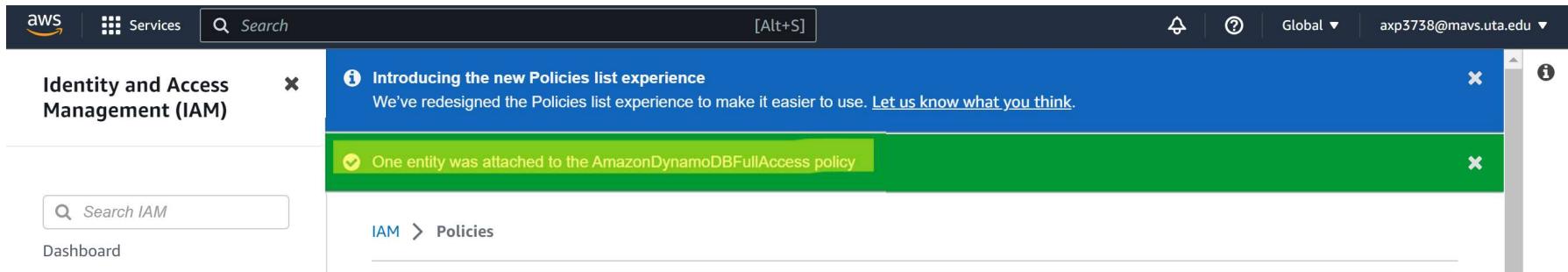
The screenshot shows the AWS IAM 'Attach policy' interface. At the top, there's a navigation bar with the AWS logo, 'Services' (selected), a search bar, and user information ('axp3738@mavs.uta.edu'). Below the navigation is a title 'Attach policy' and a subtitle 'Attach the policy to users, groups, or roles in your account'. A filter bar allows filtering by 'Name' and 'Type'. The main area displays a table with 10 results, showing a list of IAM entities:

Name	Type
axp3738	User
IAMDemo1	User
IAM_user 1	User
AmazonSagemakerCanvasForecastRole-DemoUser	Role
phadkehw10-role-06k3ttm3	Role
phadkehw10-role-abzvnr5	Role
phadkehw10-role-y5kys5w0	Role
administrators	Group
Administrator_IAMUser	Group

The row for 'AmazonSagemakerCanvasForecastRole-DemoUser' has a checked checkbox in the 'Name' column and is highlighted with a yellow background. At the bottom right of the interface are 'Cancel' and 'Attach policy' buttons, with 'Attach policy' being the active button.

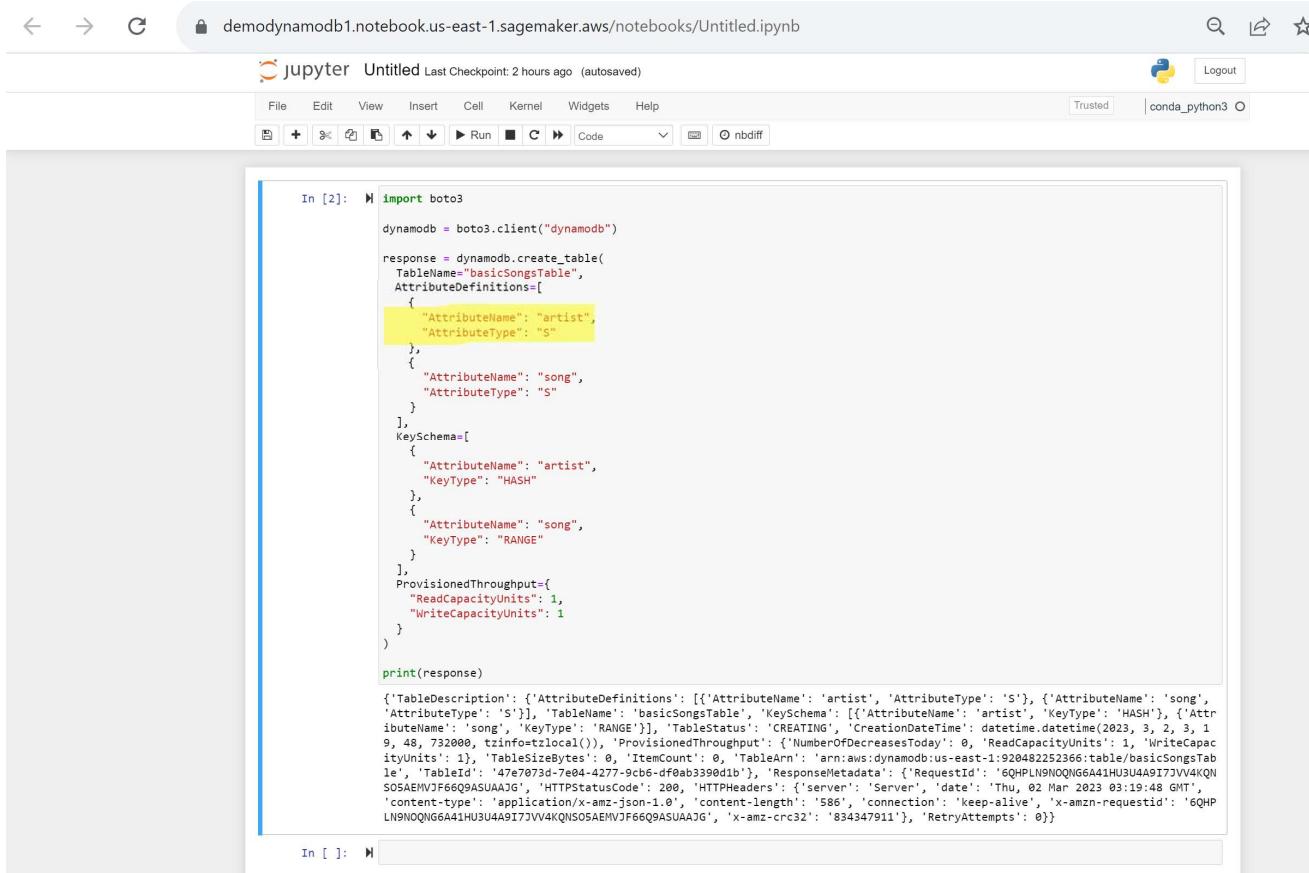
DynamoDB connect with Jupyter Notebook

4. After successful policy update, below confirmation is obtained



5. Launch Jupyter notebook through Amazon Sagemaker and open a blank notebook.
Write a sample script to create a new DynamoDB table

DynamoDB connect with Jupyter Notebook



The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** demodynamodb1.notebook.us-east-1.sagemaker.aws/notebooks/Untitled.ipynb
- Header:** jupyter Untitled Last Checkpoint: 2 hours ago (autosaved)
- Toolbar:** File, Edit, View, Insert, Cell, Kernel, Widgets, Help
- Cell Area:** In [2]:

```
import boto3

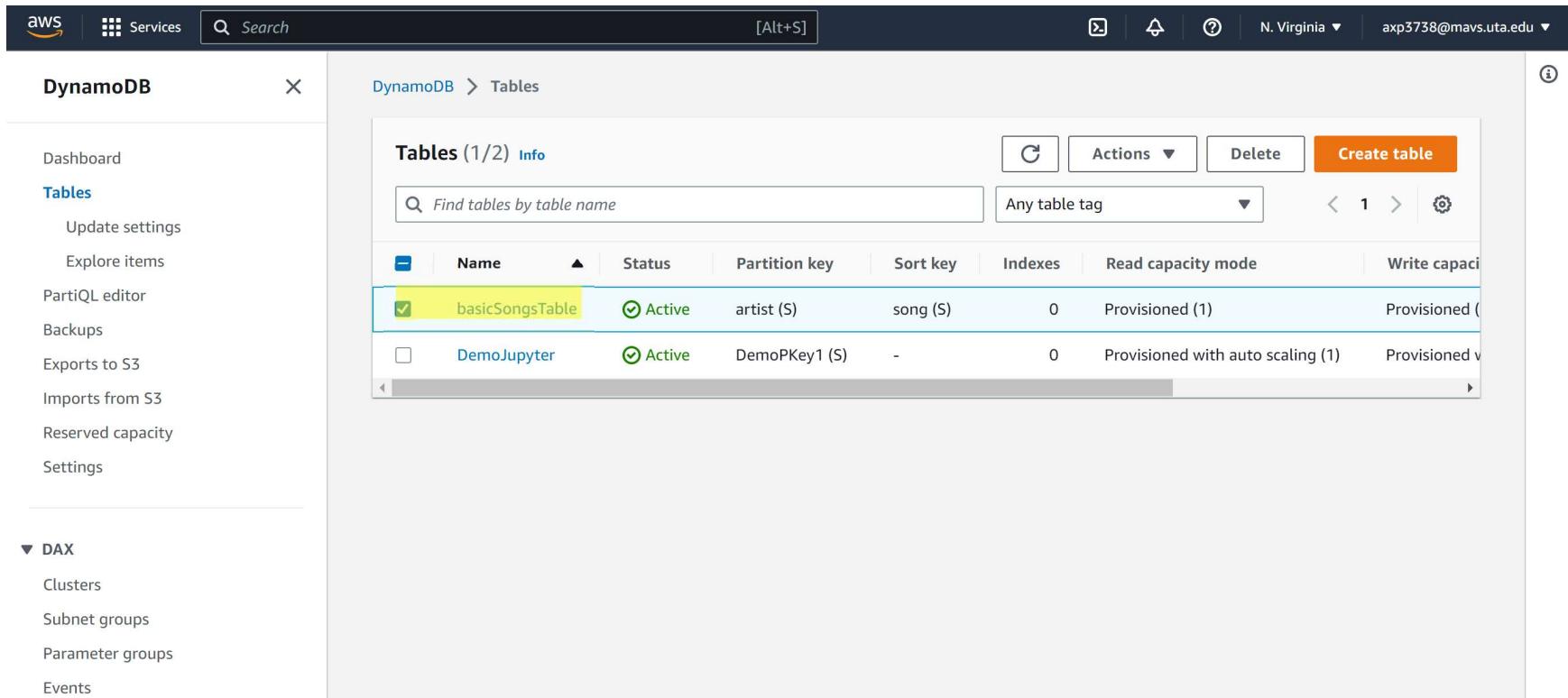
dynamodb = boto3.client("dynamodb")

response = dynamodb.create_table(
    TableName="basicSongsTable",
    AttributeDefinitions=[
        {
            "AttributeName": "artist",
            "AttributeType": "S"
        },
        {
            "AttributeName": "song",
            "AttributeType": "S"
        }
    ],
    KeySchema=[
        {
            "AttributeName": "artist",
            "KeyType": "HASH"
        },
        {
            "AttributeName": "song",
            "KeyType": "RANGE"
        }
    ],
    ProvisionedThroughput={
        "ReadCapacityUnits": 1,
        "WriteCapacityUnits": 1
    }
)
print(response)
```
- Output Area:** In []: (empty)
- Right Panel:** Trusted | conda_python3 ○ Logout

Reference: <https://www.fernandomc.com/posts/ten-examples-of-getting-data-from-dynamodb-with-python-and-boto3/>

DynamoDB connect with Jupyter Notebook

6. Check the table status in DynamoDB, table is created successfully.



The screenshot shows the AWS DynamoDB console interface. The left sidebar has 'DynamoDB' selected under 'Tables'. The main area shows a table titled 'Tables (1/2) Info' with two rows:

Name	Status	Partition key	Sort key	Indexes	Read capacity mode	Write capacity mode
basicSongsTable	Active	artist (S)	song (S)	0	Provisioned (1)	Provisioned (1)
DemoJupyter	Active	DemoPKey1 (S)	-	0	Provisioned with auto scaling (1)	Provisioned with auto scaling (1)

DynamoDB connect with Jupyter Notebook

The screenshot shows the AWS DynamoDB console interface. The left sidebar has a 'DynamoDB' tab selected, with options like Dashboard, Tables, Update settings, Explore items, PartiQL editor, Backups, Exports to S3, Imports from S3, Reserved capacity, and Settings. A 'DAX' section is also present. The main area shows the 'basicSongsTable' table details under 'Tables (2)'. The table name 'basicSongsTable' is highlighted in yellow. The 'Overview' tab is selected. A modal window titled 'Protect your DynamoDB table from accidental writes and deletes' explains Point-in-time recovery (PITR) and provides an 'Edit PITR' button. Below this, the 'General information' section displays the following details:

Partition key	Sort key
artist (String)	song (String)
Capacity mode	Table status
Provisioned	<input checked="" type="checkbox"/> Active
Alarms	Point-in-time recovery (PITR) Info
<input checked="" type="checkbox"/> No active alarms	<input type="checkbox"/> Off

Conclusion

- Jupyter notebook looks like the best option as it provides all the below:
 - AWS integration – through Amazon Sagemaker
 - Github integration – available through Amazon Sagemaker or standalone with Jupyter notebook
 - Code compilation, testing and editing options are available
 - In-built Python support with boto3
 - DynamoDB full access
 - No separate download/install is needed if done through Amazon Sagemaker
 - No extra added cost, other than the AWS services cost