|  |  |
| --- | --- |
| **Author** | **Hang Duong** |
| **Version** | **1.0** |

**BIDV ATM FUNDING MIGRATION**

**RUNBOOK**

Table of Contents

[1. Abstract: 1](#_Toc128643840)

[1.1. Objective: 1](#_Toc128643841)

[1.2. Person in charged: 1](#_Toc128643842)

[2. Assessment provided by AWS Team: 2](#_Toc128643843)

[2.1. Migration advantage: 2](#_Toc128643844)

[2.2. Cost: 2](#_Toc128643845)

[3. Prerequisite: 3](#_Toc128643846)

[4. Deployment: 3](#_Toc128643847)

[4.1. Architecture overview: 3](#_Toc128643848)

[4.2. Implement component 3 (S3 bucket): 4](#_Toc128643849)

[4.3. Implement component 4 (Amazon ECR) : 11](#_Toc128643850)

[4.4. Implement component 2 (SageMaker Processing Job): 22](#_Toc128643851)

[4.5. Implement component 1 (Eventbridge Scheduler): 22](#_Toc128643852)

[4.6. Clean up un-used resouce: 32](#_Toc128643853)

[5. Further improvement: 33](#_Toc128643854)

## 1. Abstract:

### 1.1. Objective:

BIDV Data team currently runs ATM cash prediction algorithm on premise and wants to migrate to AWS Cloud.

### 1.2. Person in charged:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Name** | **Role** | **Phone** | **Email** | **Note** |
| 1 | Hang Duong | Demo the migration and prep docs. | +84788383533 | hangdtt@amazon.com |  |
| 2 | Huy Tran |  |  |  |  |
| 3 | Hung Hoang |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

## 2. Assessment provided by AWS Team:

### 2.1. Migration advantage:

|  |  |  |
| --- | --- | --- |
|  | On-premise | AWS Cloud |
| Time processing | ~ 2 days | ~ < 30 mins. (can be optimized to ~15mins,  e.g: use ml.c5.18xlarge with multi-thread processing, or upgrade quota as in section 3) |
| Auto Scheduler | Not yet | Yes |
| Infrastructure | Lack of infrastructure | Broad and numerous infrastructures |
| ML tool | Not available, limited | Available, diverse |

### 2.2. Cost:

**Total Cost (SageMaker cost + DataSync cost + Amazon S3):**

$21.2/month (option 1), $14.7/month (option 2)

#### 2.2.1. SageMaker Cost:

|  |  |  |
| --- | --- | --- |
| Env | Price (region: Singapore) | Details |
| Dev(Test) | $11.09 / month  ($0.063 per hour instance) | SageMaker notebook instance:  For : 1 Data Scientist,  1 notebook ml.t3.medium (2vCPU, 4GiB) , 8 hours/day,  22 days/month |
| Prod | Option 1:  $5.29/run/week  $21.16/4 runs/month  Option 2:  $3.67/run/week  $14.68/ 4 runs/month | SageMaker Processing job:  Option 1: (no need to refactor R code)  Use instance ml.t3.large(2vCPU, 8 GB ram, $0.127/hour/1 instance)  100 instances per run (each instance runs one R script)    Option 2: (need to refactor R code)  Use instance: ml.c5.18xlarge(72 vCPU, 144 GB ram, $4.234/hour/1 instance), 4 instances per run (~ 50 parallell R scripts per instance, 13 mins per instance per run 4 jobs per month) |

#### 2.2.2. DataSync Cost:

Suppose you move 1GB from on-premise to Amazon S3 per month -> Cost: $0.0125/month

(price: 0.0125 USD/1GB)

#### 2.2.3 Amazon S3 cost:

Suppose you have 1GB in Amazon S3 per month, and run SageMaker Processing Job as above configuration: -> Cost $0.03/month (details: <https://calculator.aws/#/estimate?id=1b454fa422db8e5c740250cccf60778241684470> )

## 3. Prerequisite:

- AWS account (that will run ATM cash prediction algorithm), and an AWS IAM user with the right permission (Eventbridge full access, Sagemaker full access, S3 full access, ECR full access).

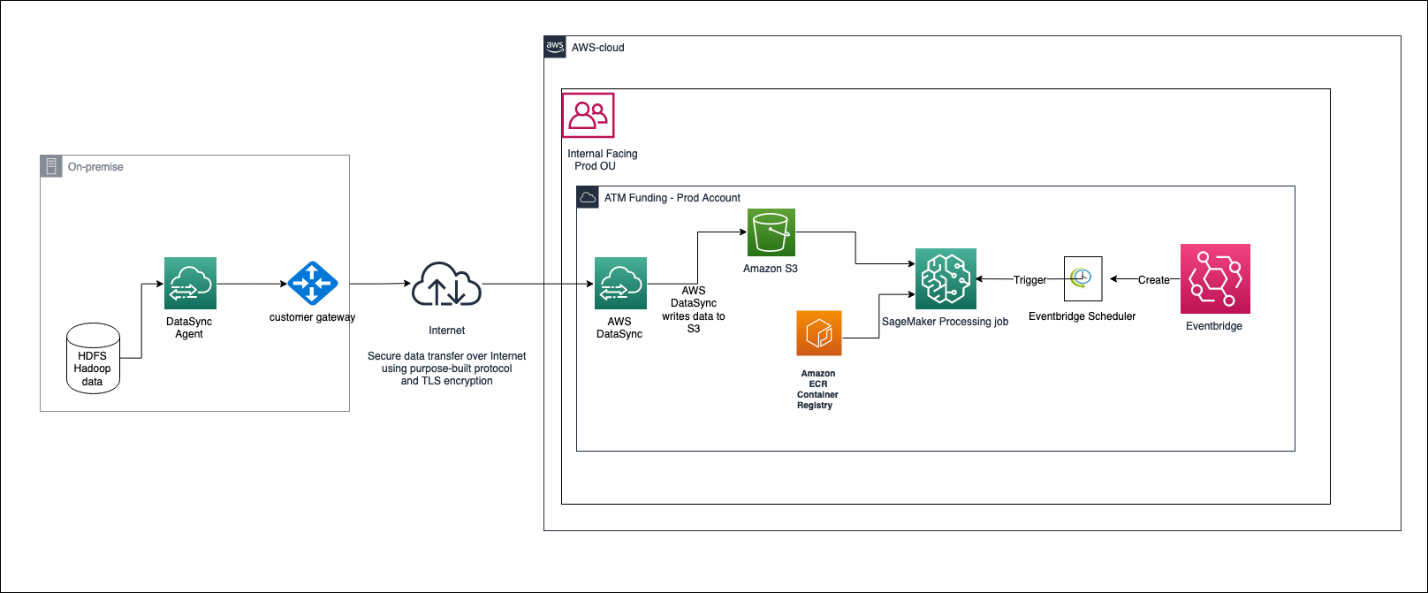
* Download folder “Migration” along with this documentation.
* Upgrade your SageMaker processing job ml.t3.large quota to 100. (or 200 if you want the time processing to be ~ 15mins).
* Upgrade your SageMaker Number of instances across all processing jobs quota to 100.

## 4. Deployment:

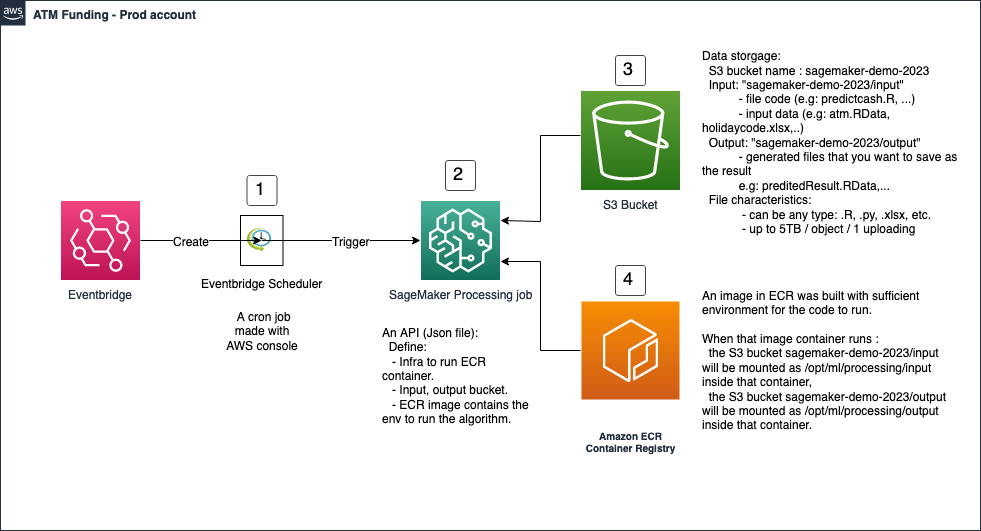
There are some ways to implement the migration, this deployment shows one of them.

### 4.1. Architecture overview:

General architecture:



Details architecture on AWS Cloud for ATM Funding – Prod account :



From General architecture, you can see that if you want to move data from your on-premise Hadoop to Amazon S3, you need to use AWS DataSync, follow this documentation to implement it: <https://aws.amazon.com/blogs/storage/using-aws-datasync-to-move-data-from-hadoop-to-amazon-s3/> .

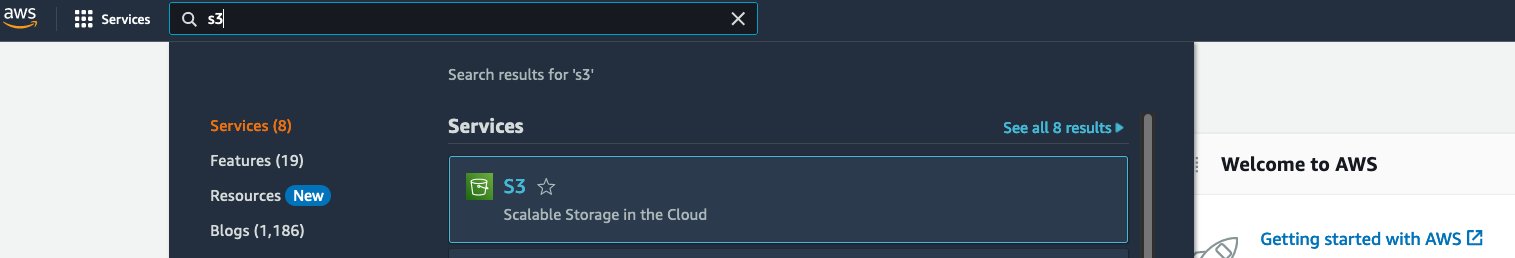
This solution below assumes you already have data in S3 (or on local computer), and we will create SageMaker processing job (that runs the algorithm) every Sunday at 7AM.

We will implement in following order: component 3 (S3 bucket) → component 4 (Amazon ECR) → component 2 (SageMaker Processing Job) → component 1 (Eventbridge Scheduler).

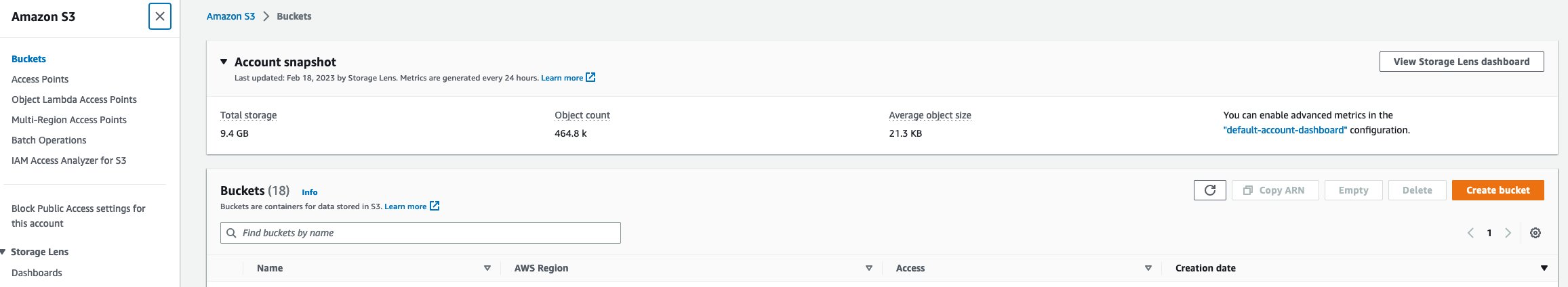
### 4.2. Implement component 3 (S3 bucket):

Create S3 Bucket:

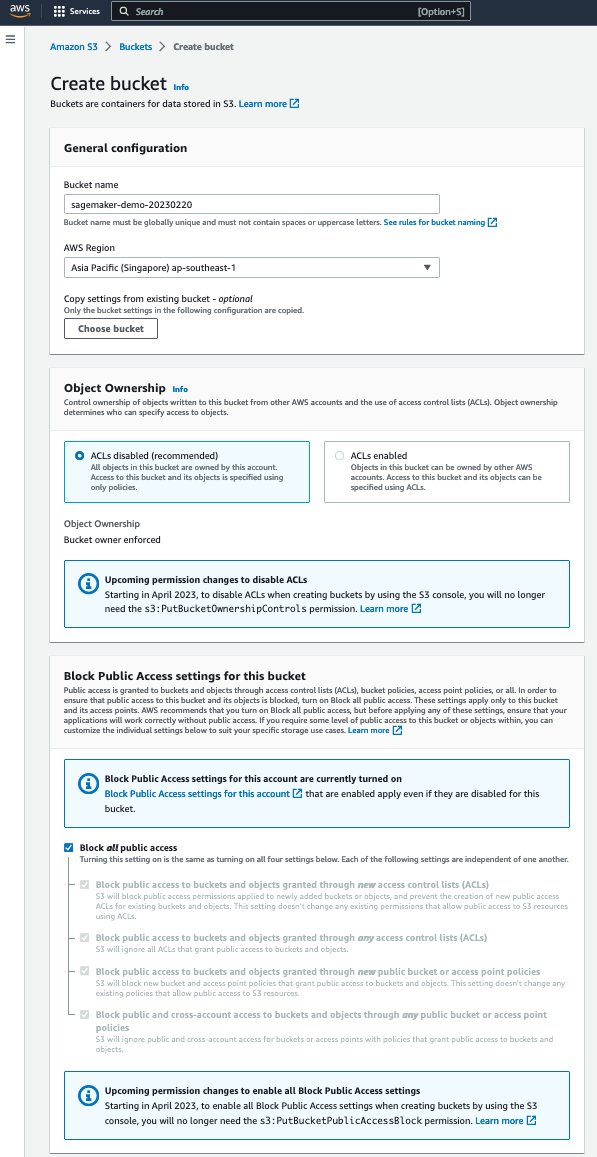
Type “s3” in the search bar and Click “S3” in Services section results:

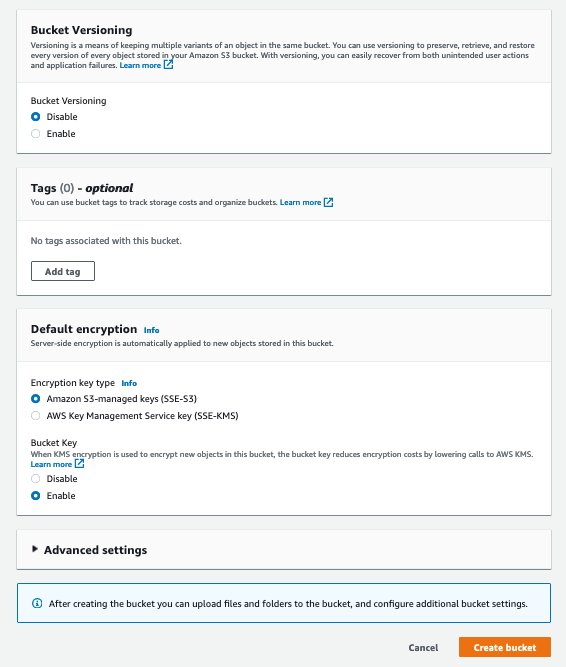


You’ll see Amazon S3 page, click the orange button “Create bucket”:



Fill the information as bellow:

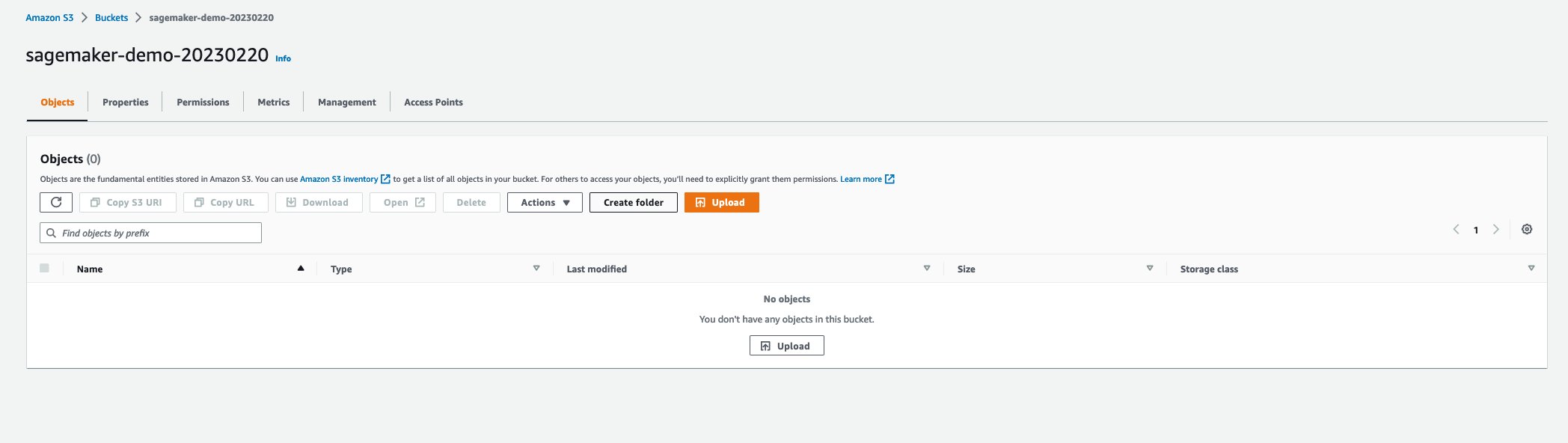




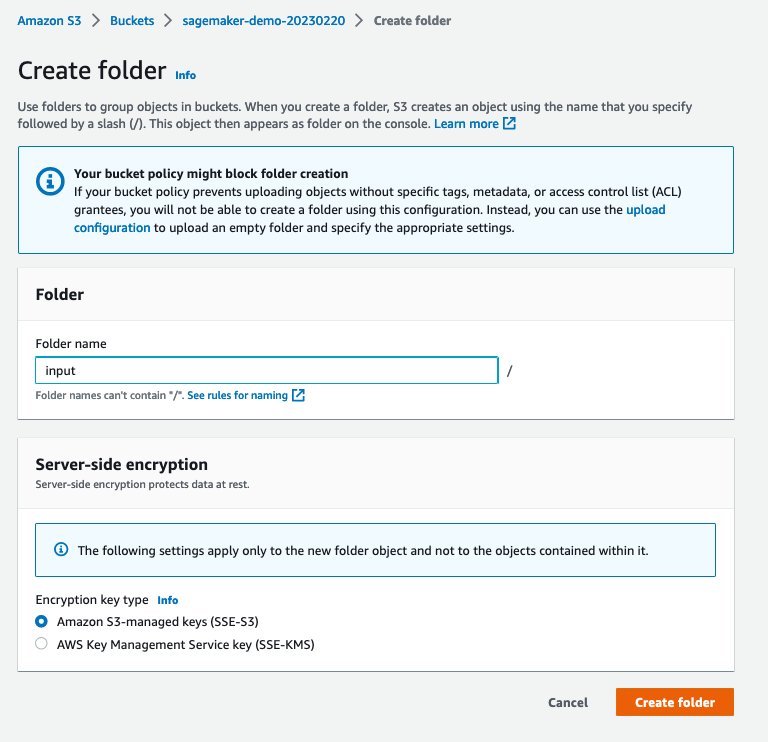


Click “Create bucket” button.

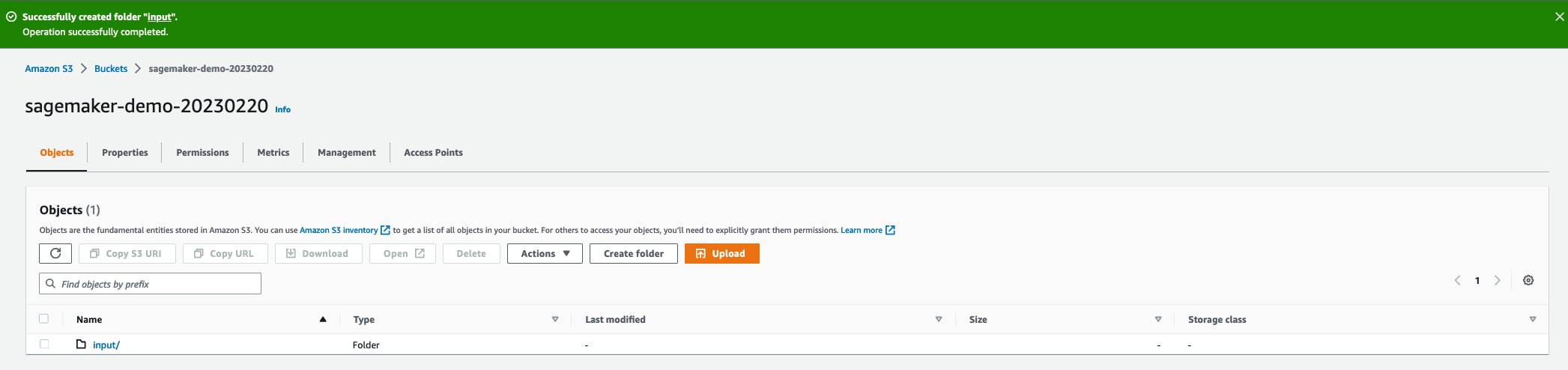
You will see “sagemaker-demo-20230220” bucket:



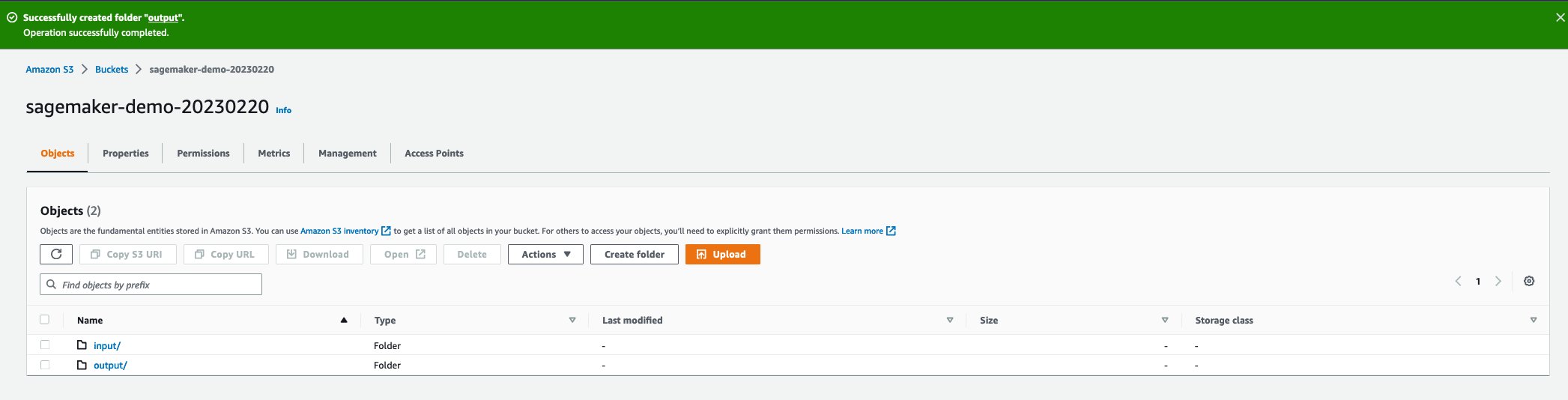
Click “Create folder” and fill the information as bellow:



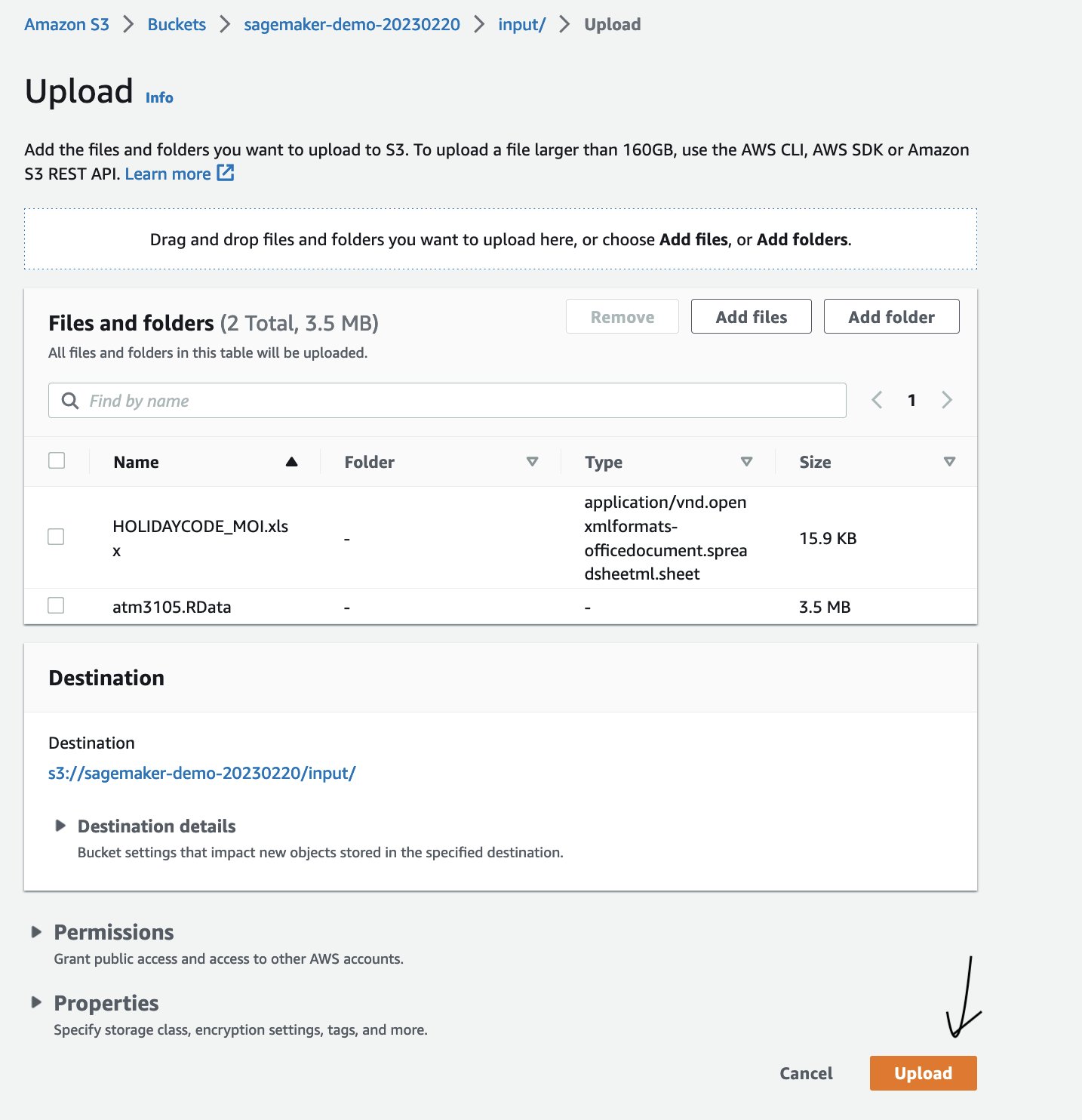
Click “Create folder”:



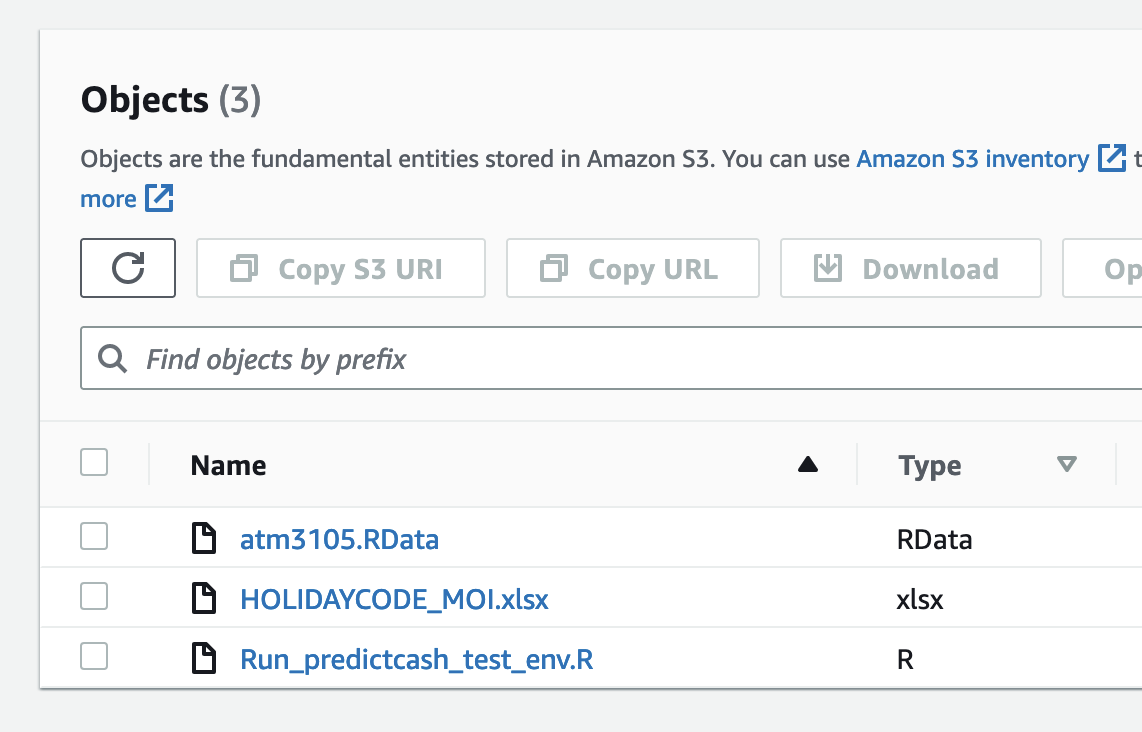
Repeat the process to make another folder named “output”, after all, you’ll have 2 folders as bellow:



Click “input/” → Click “Upload” orange button → Click “Add files” button → Select “HOLIDAYCODE\_MOI.xlsx” and “atm3105.RData“ → Click ”Upload“ orange button :

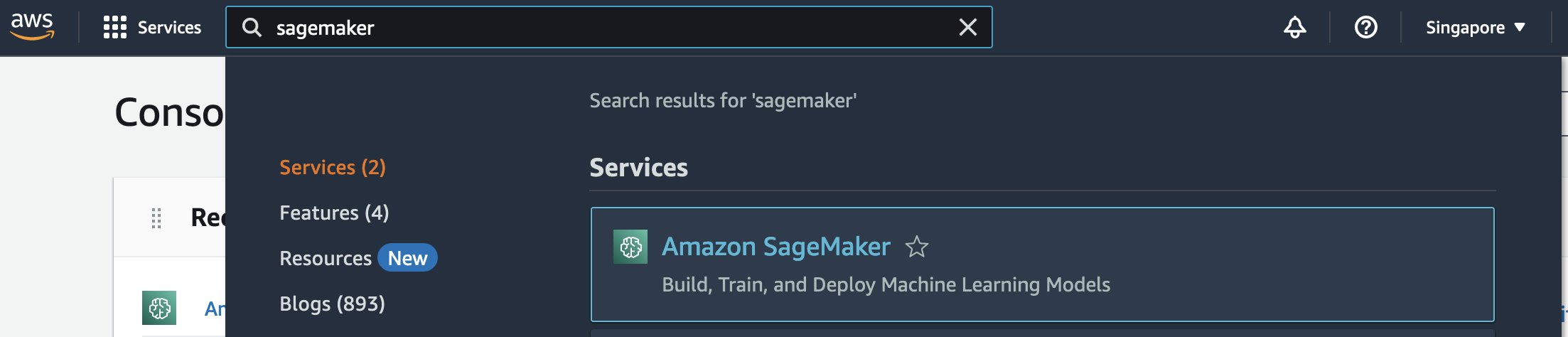


Wait a second and you’ll see:

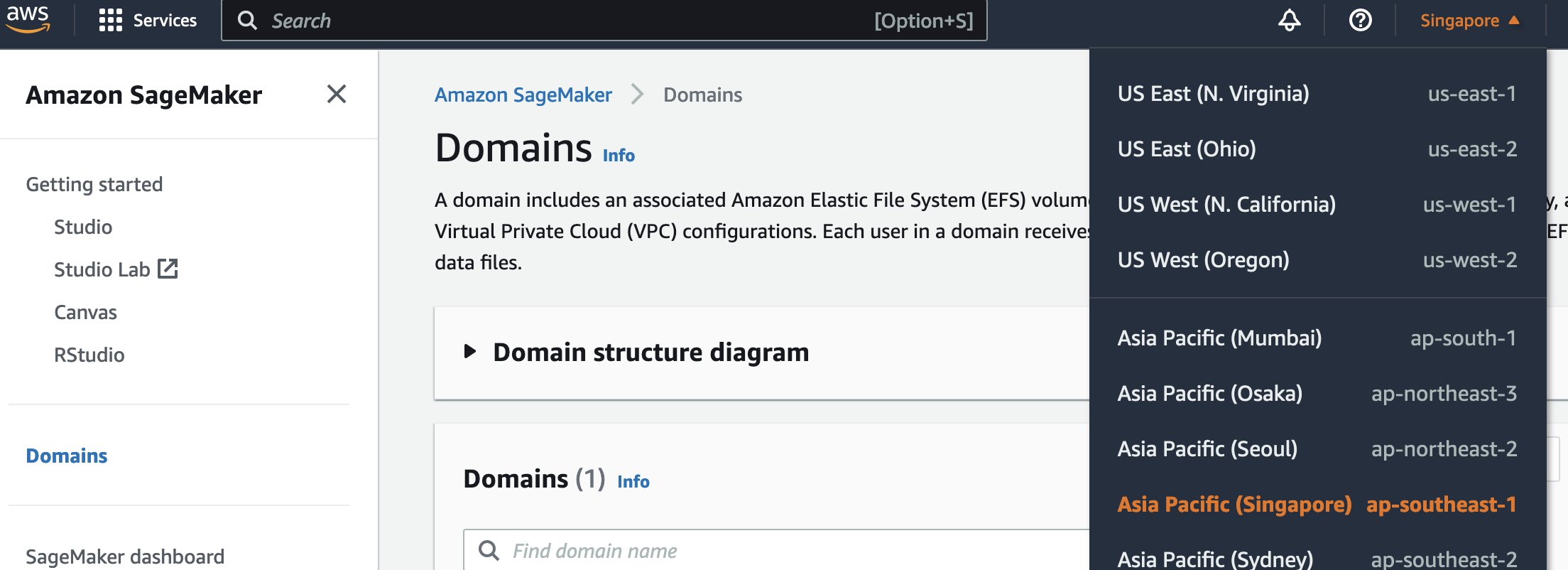
Similarity, upload the code algorithm (e.g: Run\_predictcash\_test\_env.R) :

### 4.3. Implement component 4 (Amazon ECR) :

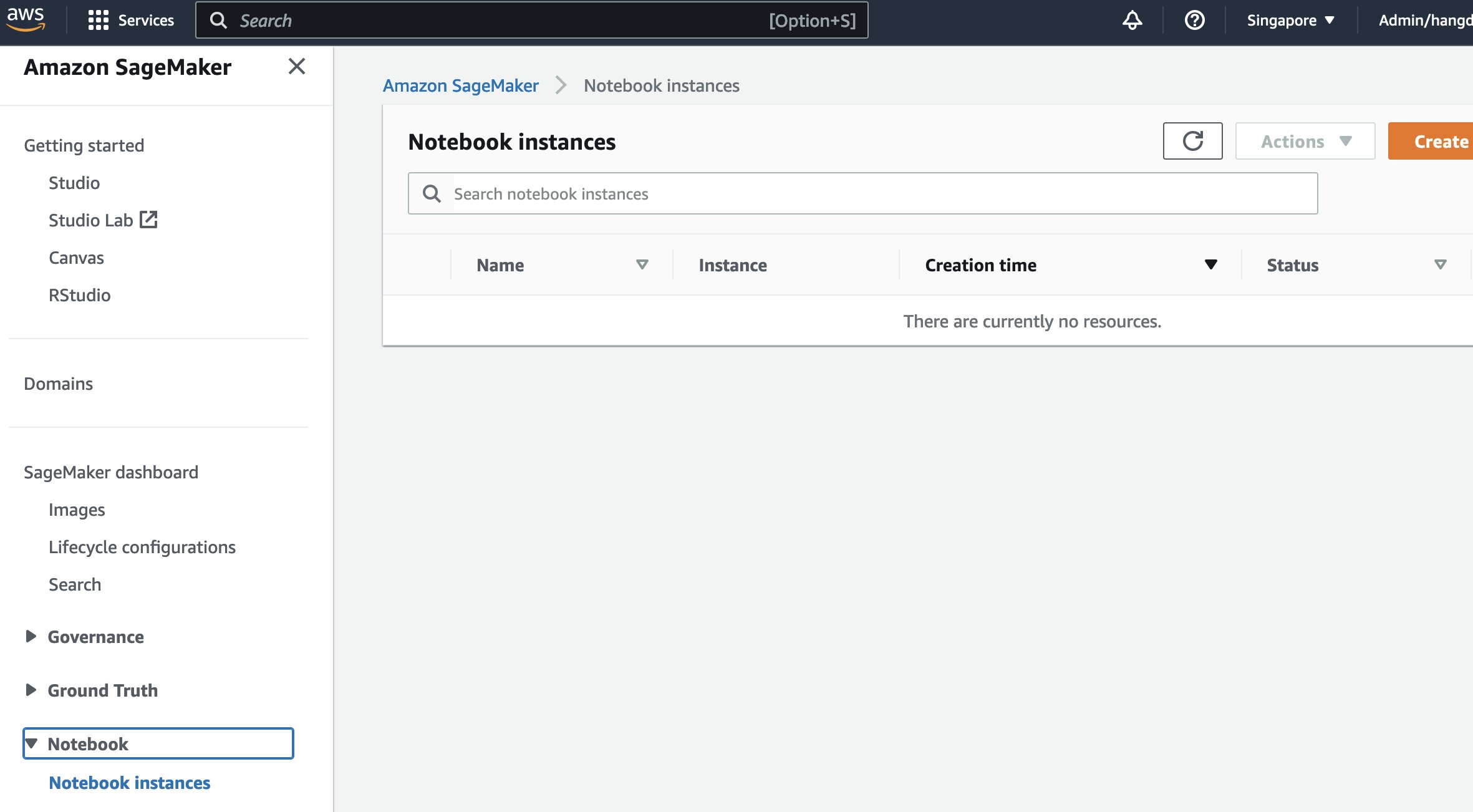
Search for “Amazon Sagemaker” service:



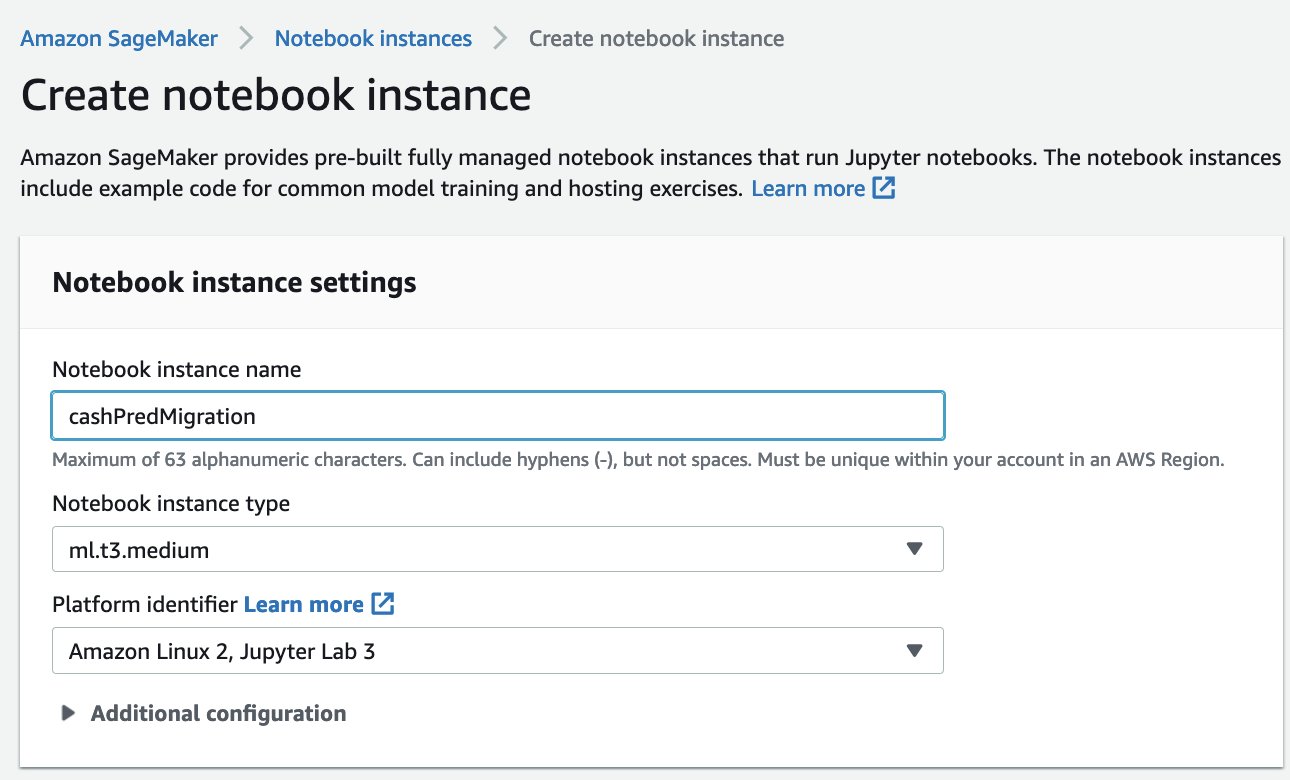
Select region: “Singapore”:



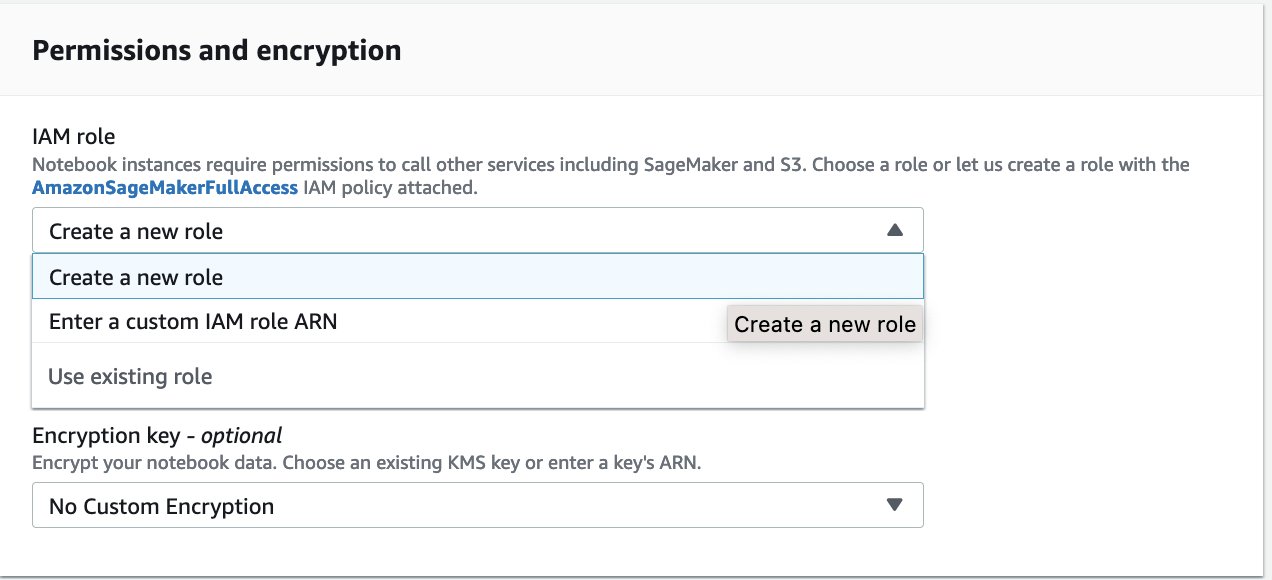
Select “Notebook” → “Notebook instances”



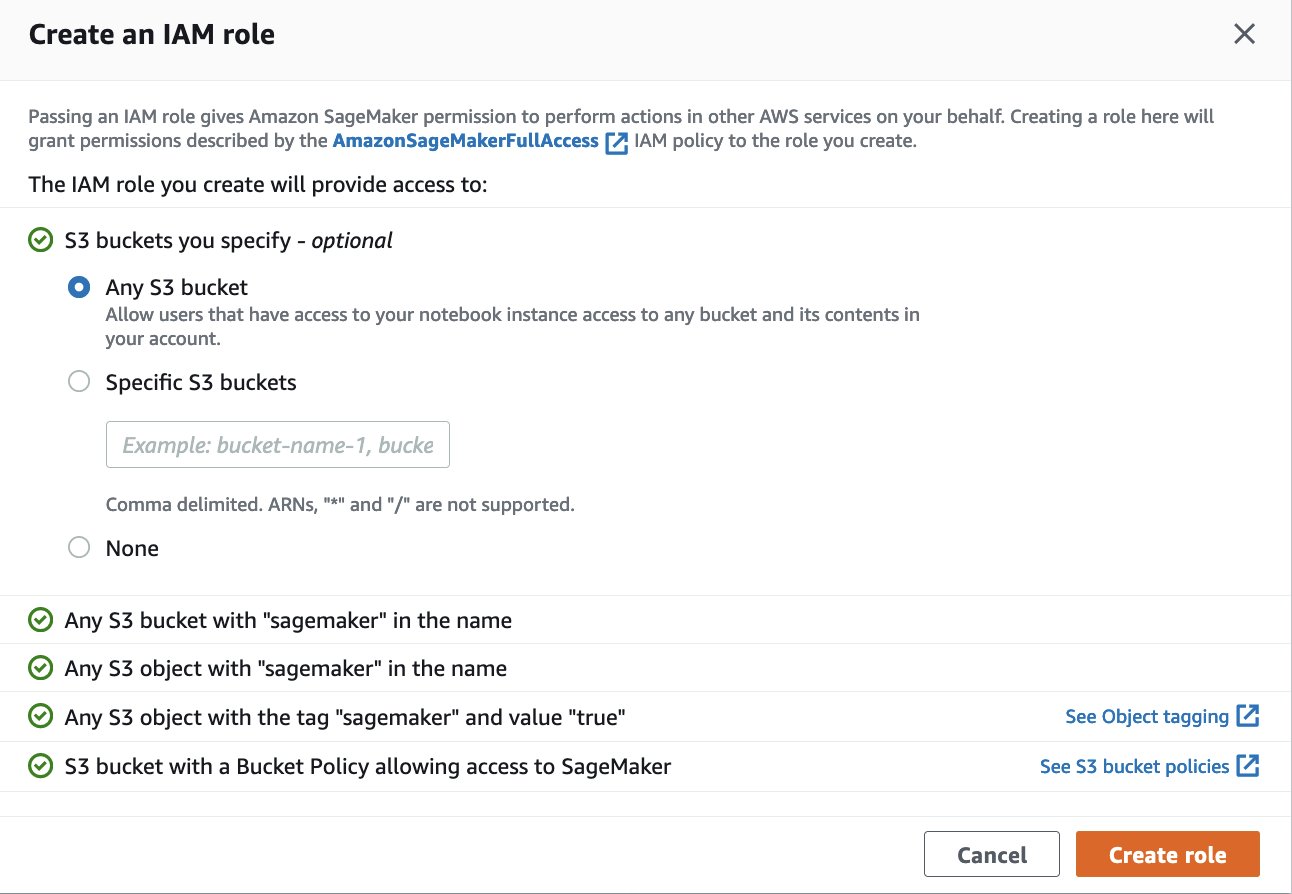
Select “Create notebook instance” and fill information as bellow:



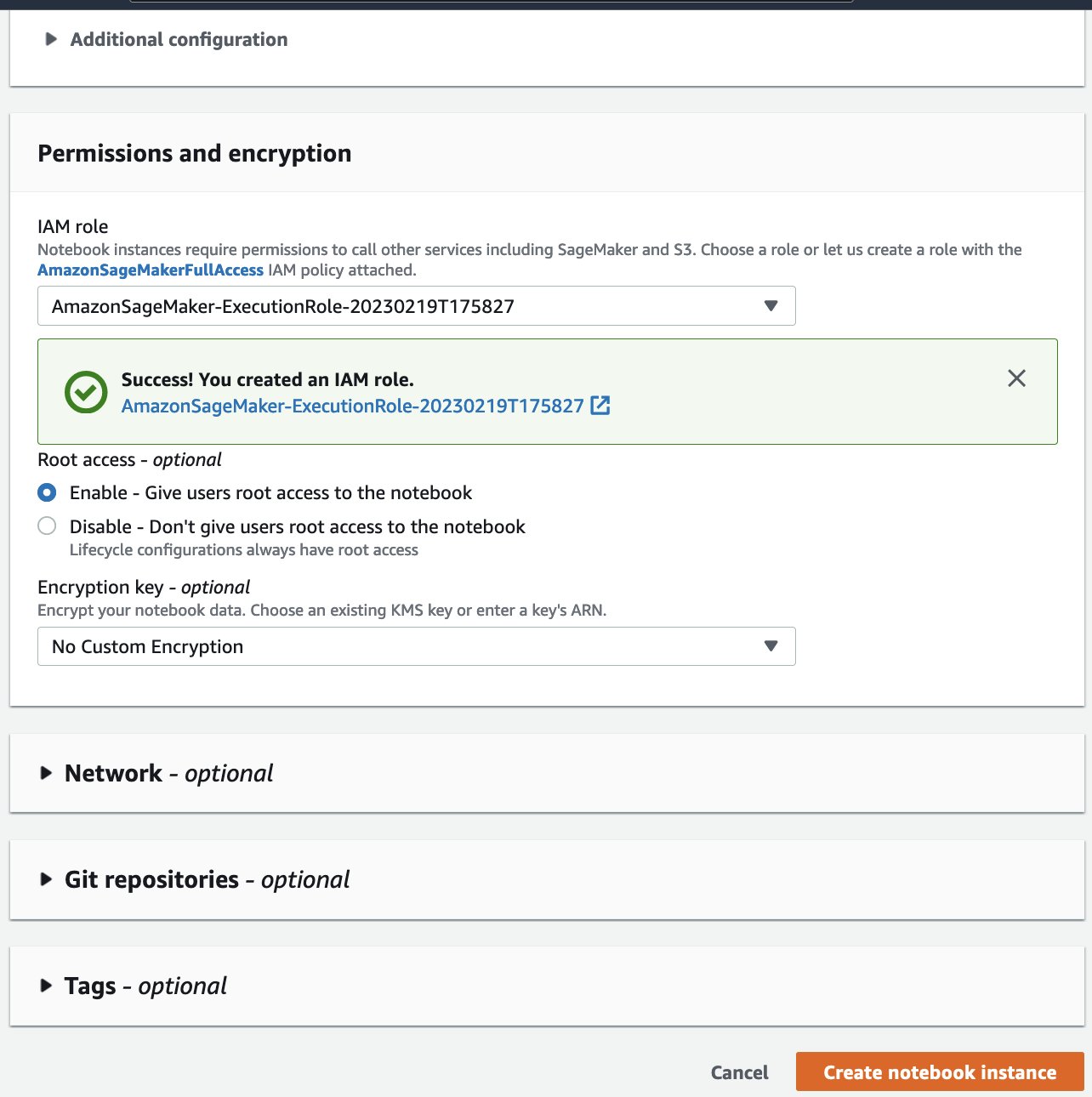
Select “Create a new role”:



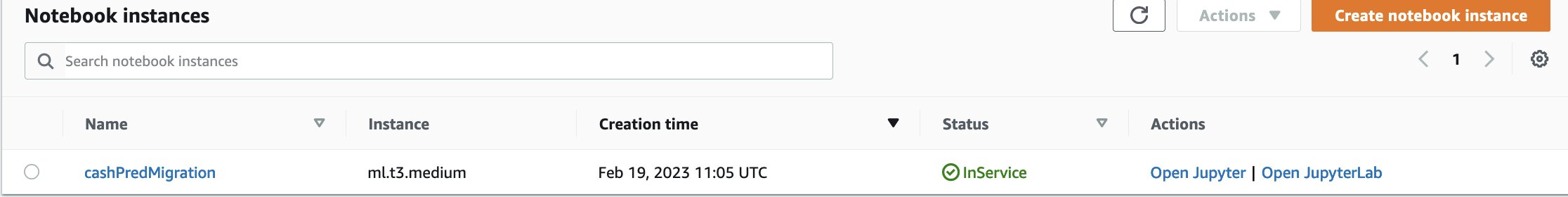
Leave everything as default as bellow image and select “Create role”:



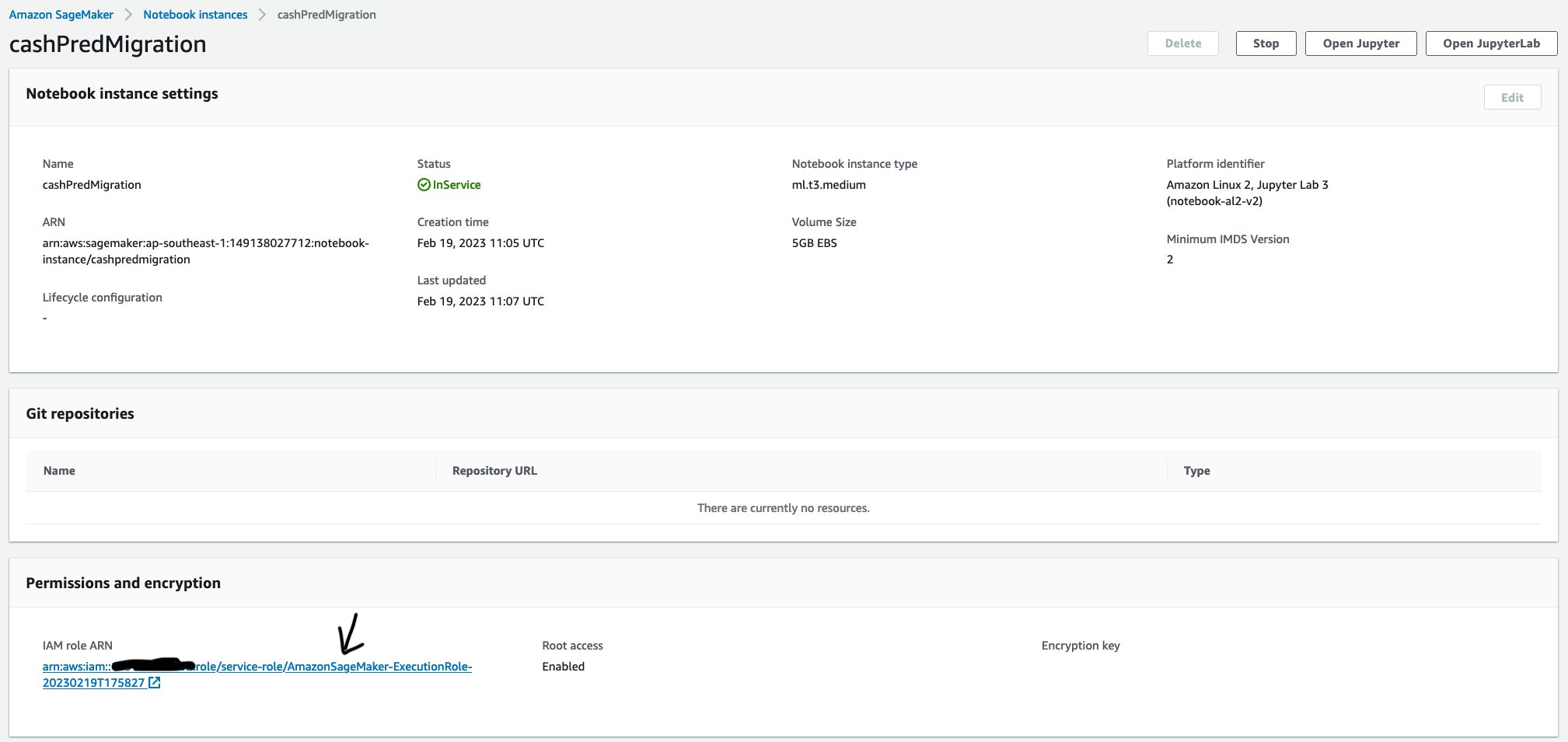
Leave everything as default as bellow image and select “Create notebook instance”:



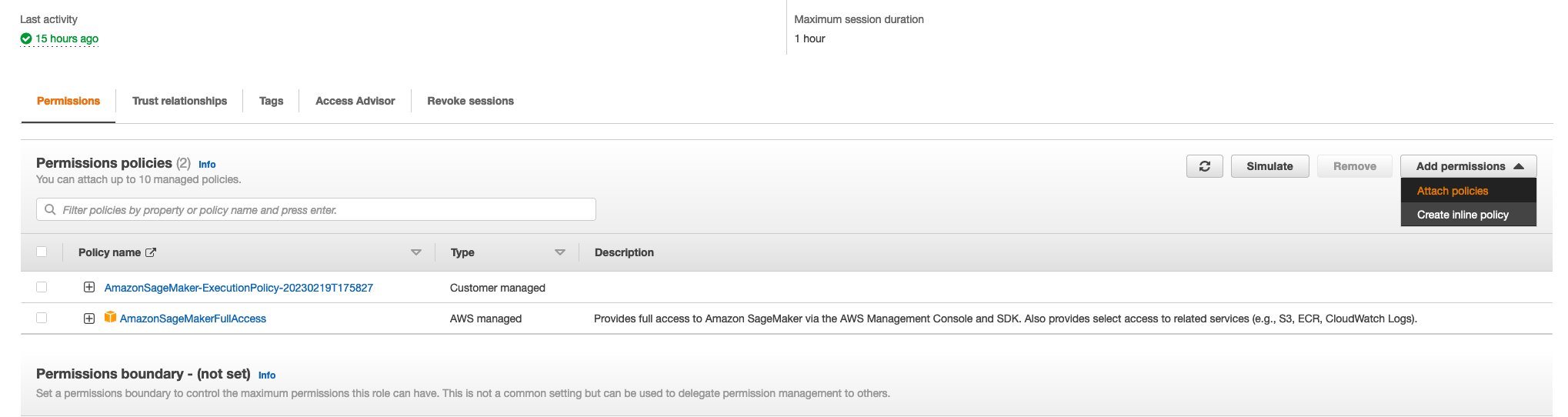
You’ll wait (~minute) until the “Status” shows “InService” (Click the Refresh button to see change):



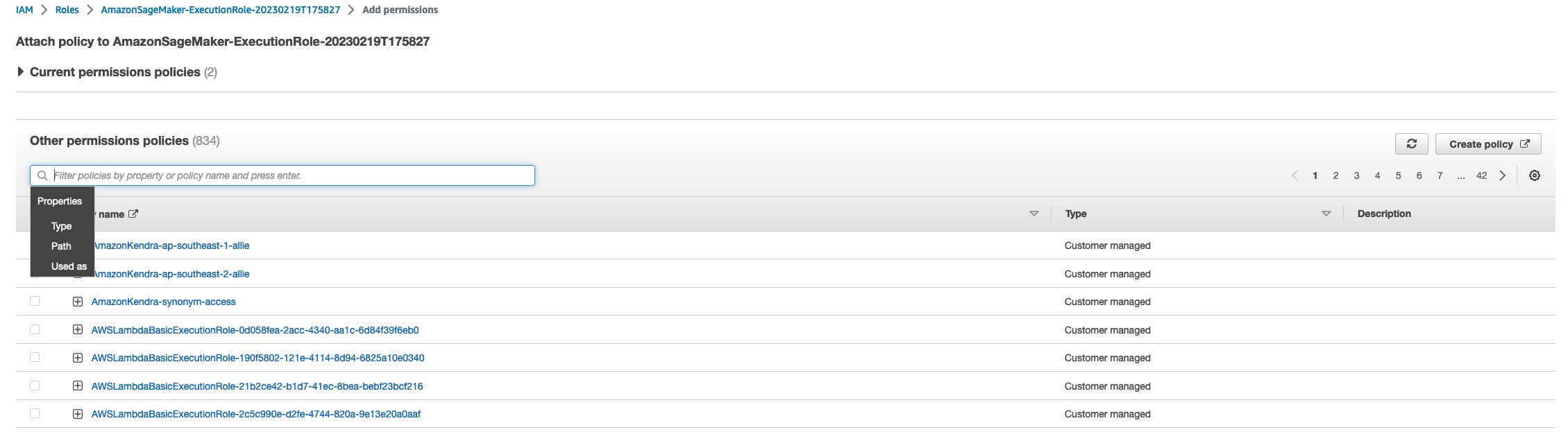
Click “cashPredMigration” and Copy the IAM role ARN link (which is highlighted with blue color as bellow image) to your notepad (we will use it latter), then Click that link :



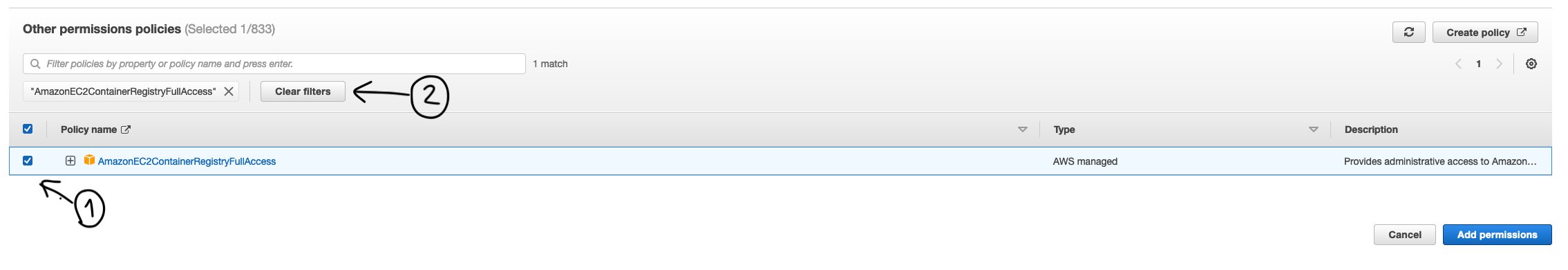
A new tab will be opened as bellow, Click “Add permissions” → “Attach policies”:

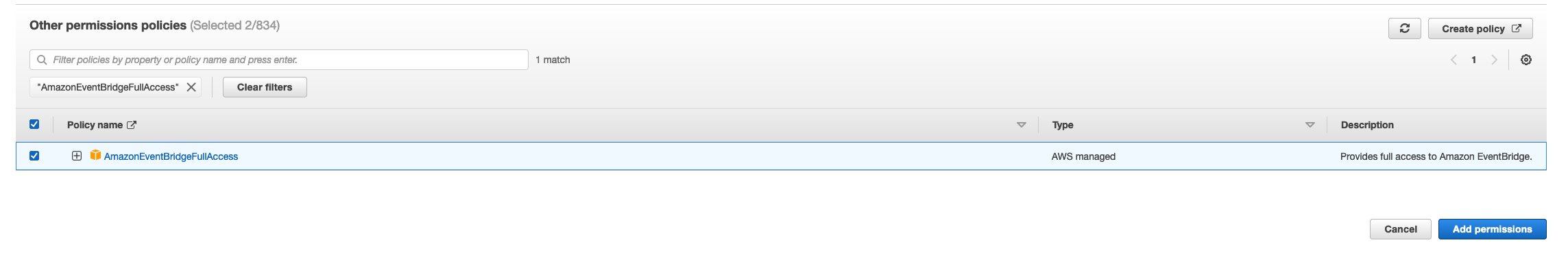


You will see a screen of Policy name as following:

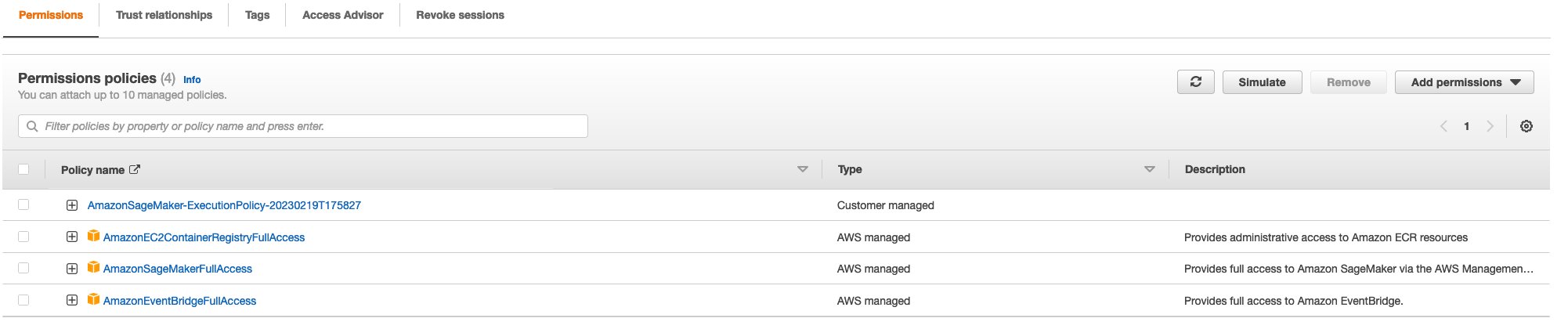


Type: “AmazonEC2ContainerRegistryFullAccess" with no quote (in the blank bellow Other permissions policies)→ Enter → Tick the box → Click “Clear filters” → Type: [“AmazonEventBridgeFullAccess](https://us-east-1.console.aws.amazon.com/iam/home#/policies/arn:aws:iam::aws:policy/AmazonEventBridgeFullAccess)“ with no quote (in the blank bellow Other permissions policies)→ Enter → Tick the box →  Click Add permissions (as bellow image):

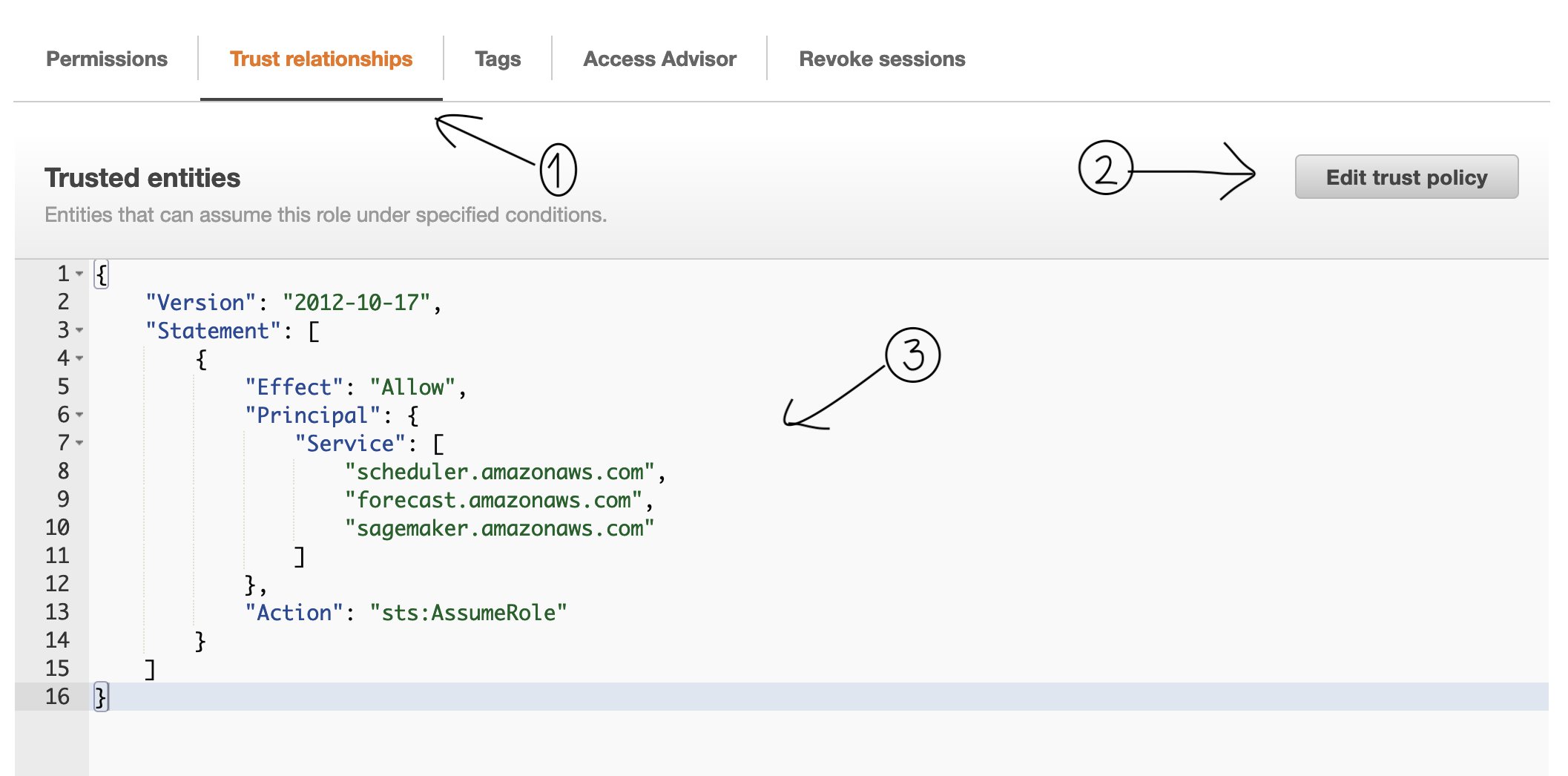




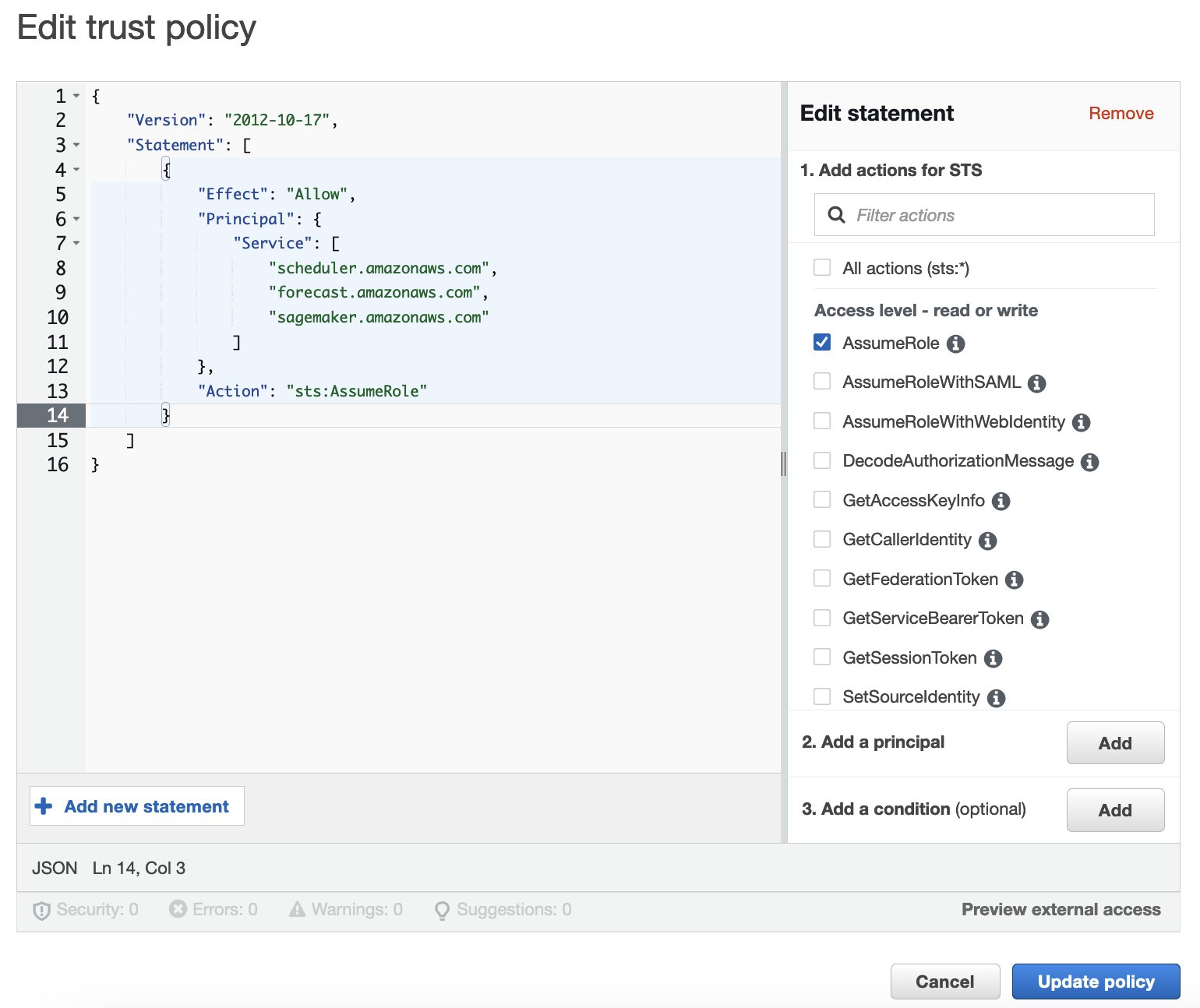
After all, you will see 4 policies as bellow:



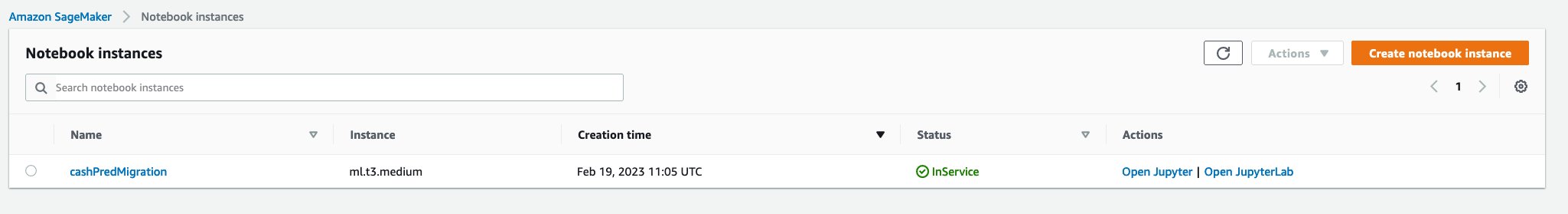
Click the tab “Trust relationships” → “Edit trust policy” → then you edit the policy as bellow image:



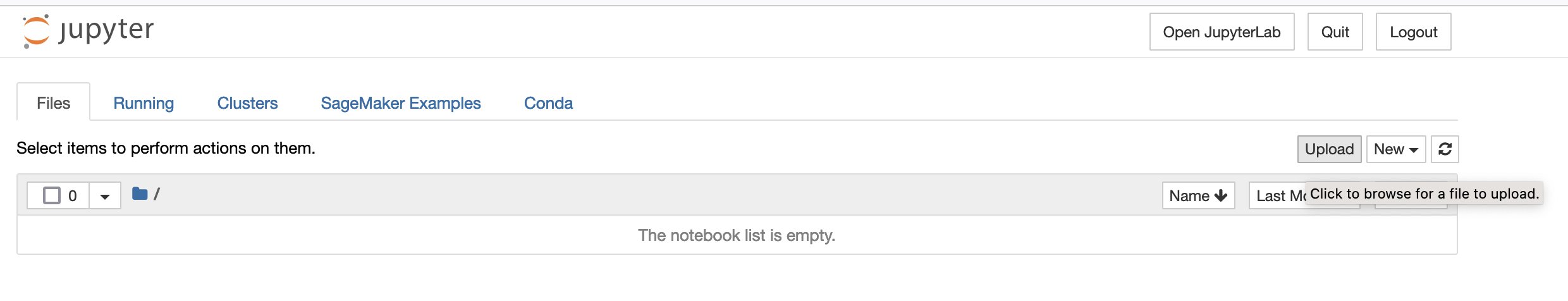
After you finish editing → Click “Update policy” :



Close this tab, and come back the Notebook instances tab:



Click “Open Jupyter” (in Actions column) and a new tab will be opened as follow:



Click “Upload” and select “Deployment.ipynb”, “atm3105.RData”, “Dockerfile”, “HOLIDAYCODE\_MOI.xlsx”, “usedLibraries.R” and then click **“Upload” (blue button)** to upload :



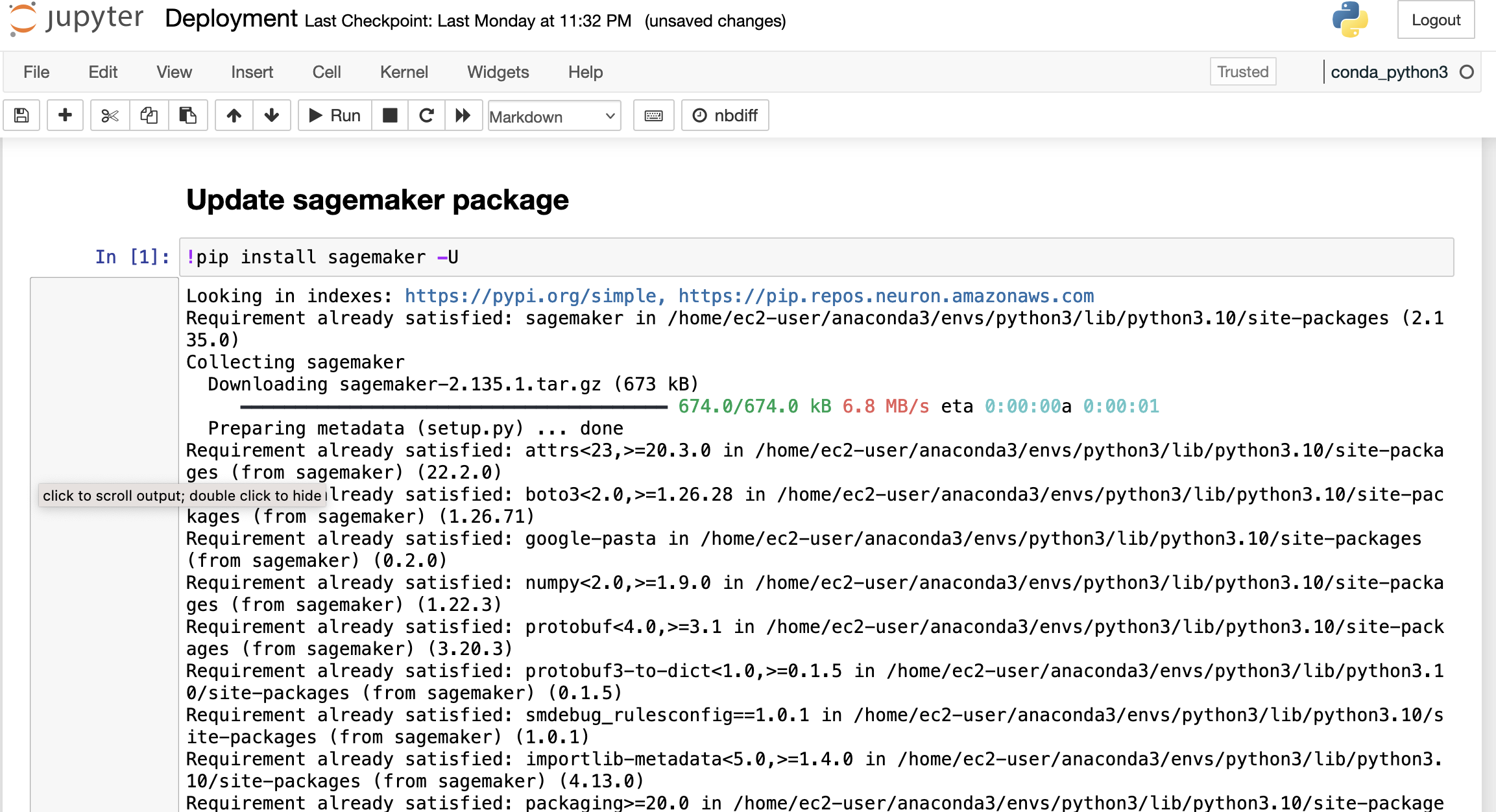


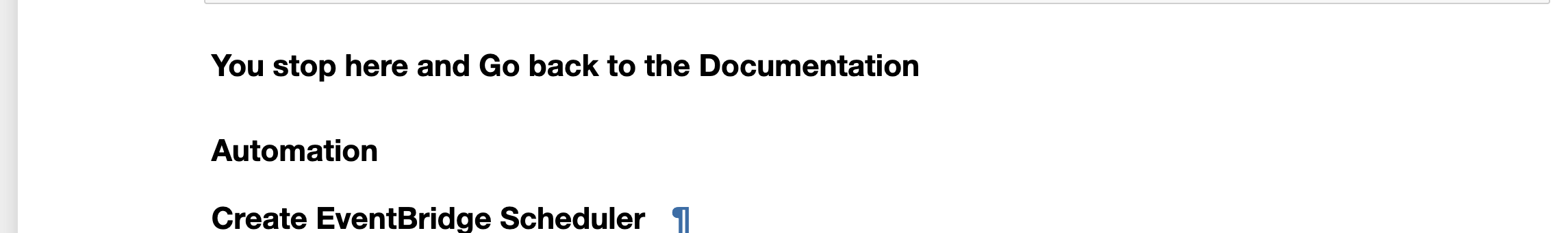
Wait until you see all files are uploaded successfully as bellow:





Click the “Deployment.ipynb” file to open it:

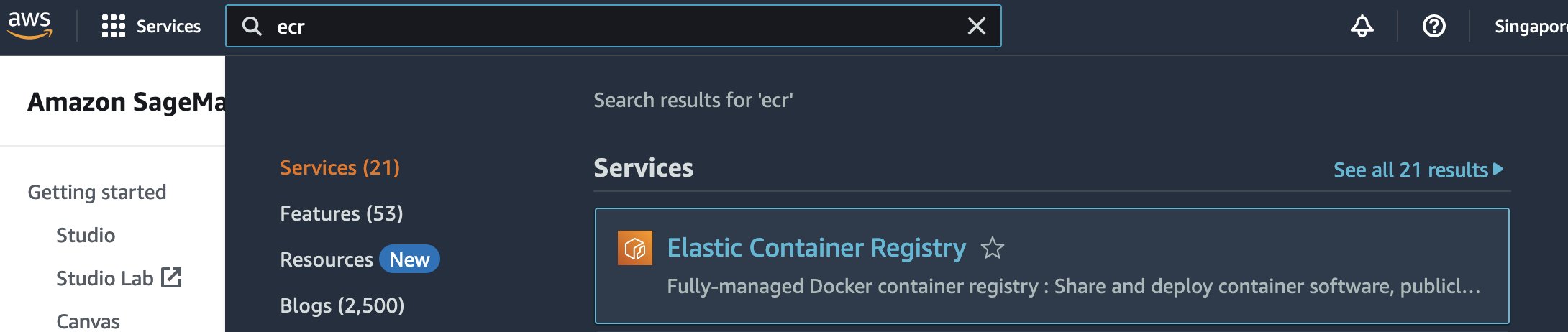


Run every cell by click on each cell and press “Shift+Enter”, until your see “You stop here…” and wait until your tab icon show the notebook as below:

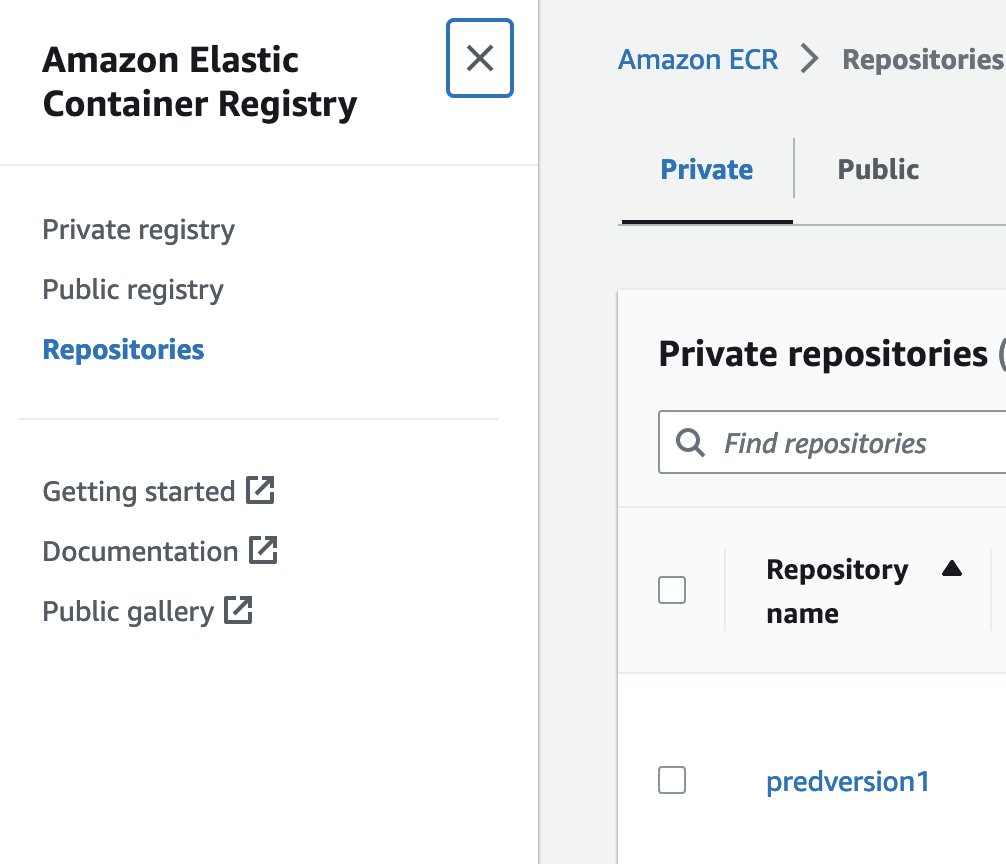


(you need to wait about ~25mins to run all those above cells successfully)

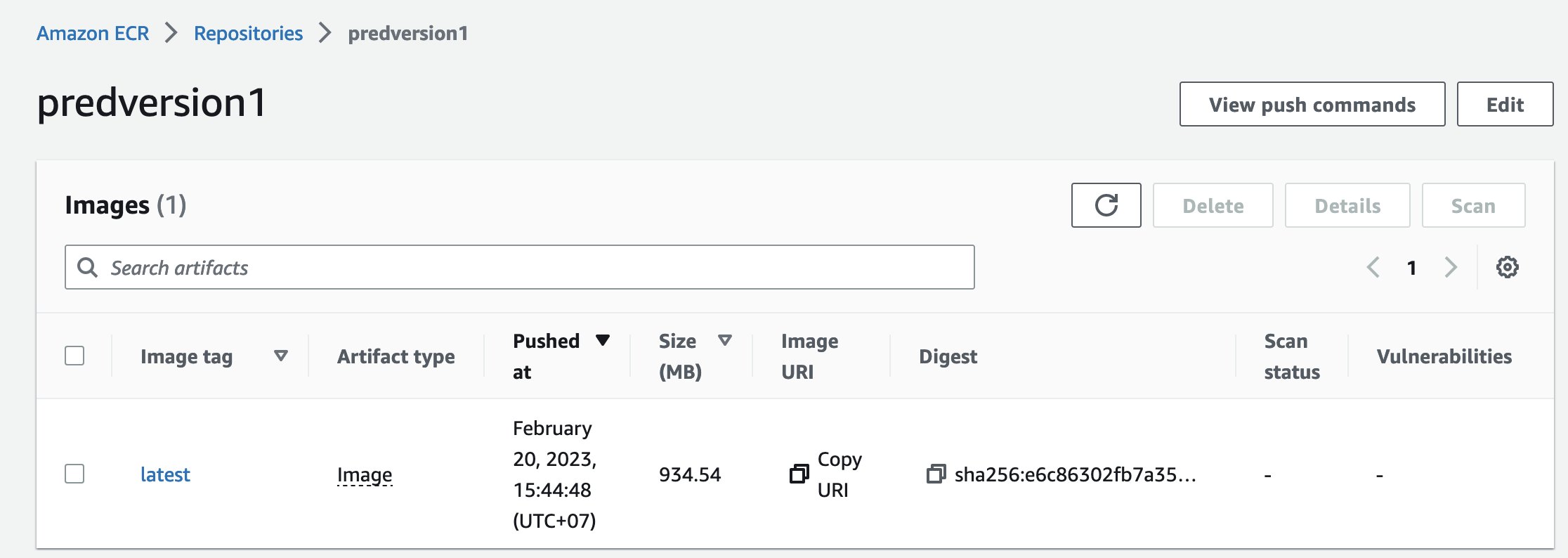
open the AWS Console and type “ecr” in search bar, click “Elastic Container Registry” :



Open “Repositories” tab, you’ll see “predversion1” in Private repositories as following:



Click “predversion1” :



Click the icon near “Copy URI” to copy the image URI, paste your URI in “ImageUri” in the following sagemaker\_processing\_job\_api.json file:

***sagemaker\_processing\_job\_api.json***

{

"AppSpecification": {

"ImageUri": " your\_image\_URI\_goes\_here "

},

"ProcessingInputs": [

{

"AppManaged": false,

"InputName": "input",

"S3Input": {

"LocalPath": "/opt/ml/processing/input/",

"S3DataType": "S3Prefix",

"S3InputMode": "File",

"S3Uri": "s3://sagemaker-demo-20230220/input/"

}

}

],

"ProcessingJobName": "predictCashATM-0301-19",

"ProcessingOutputConfig": {

"Outputs": [

{

"AppManaged": false,

"OutputName": "output",

"S3Output": {

"LocalPath": "/opt/ml/processing/output/",

"S3UploadMode": "EndOfJob",

"S3Uri": "s3://sagemaker-demo-20230220/output/"

}

}

]

},

"ProcessingResources": {

"ClusterConfig": {

"InstanceCount": 1,

"InstanceType": "ml.t3.large",

"VolumeSizeInGB": 1

}

},

"Environment" : {

"SIR": "990035" # Replace your ATM ids list here !!!!

},

"RoleArn": " your\_cashPredMigration\_IAM\_role\_ARN\_goes\_here "

}

Change the “RoleArn” value to your “cashPredMigration” IAM role ARN (which you saved in your notepad before).

Replace your ATM id in “Environment” (you choose one ATM id just for demo)

Save your sagemaker\_processing\_job\_api.json file.

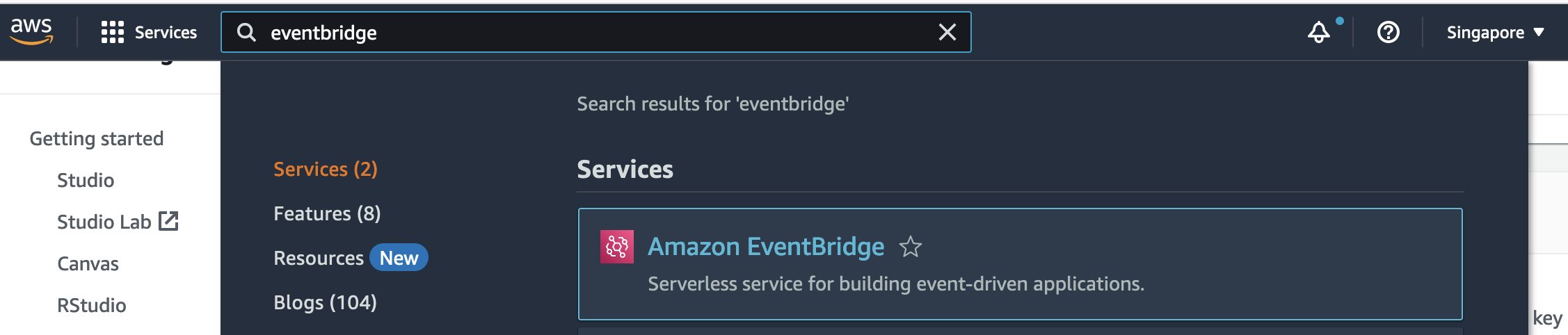
### 4.4. Implement component 2 (SageMaker Processing Job):

You’ve just created *sagemaker\_processing\_job\_api.json* filein the above step and that’s also the SageMaker Processing Job API.

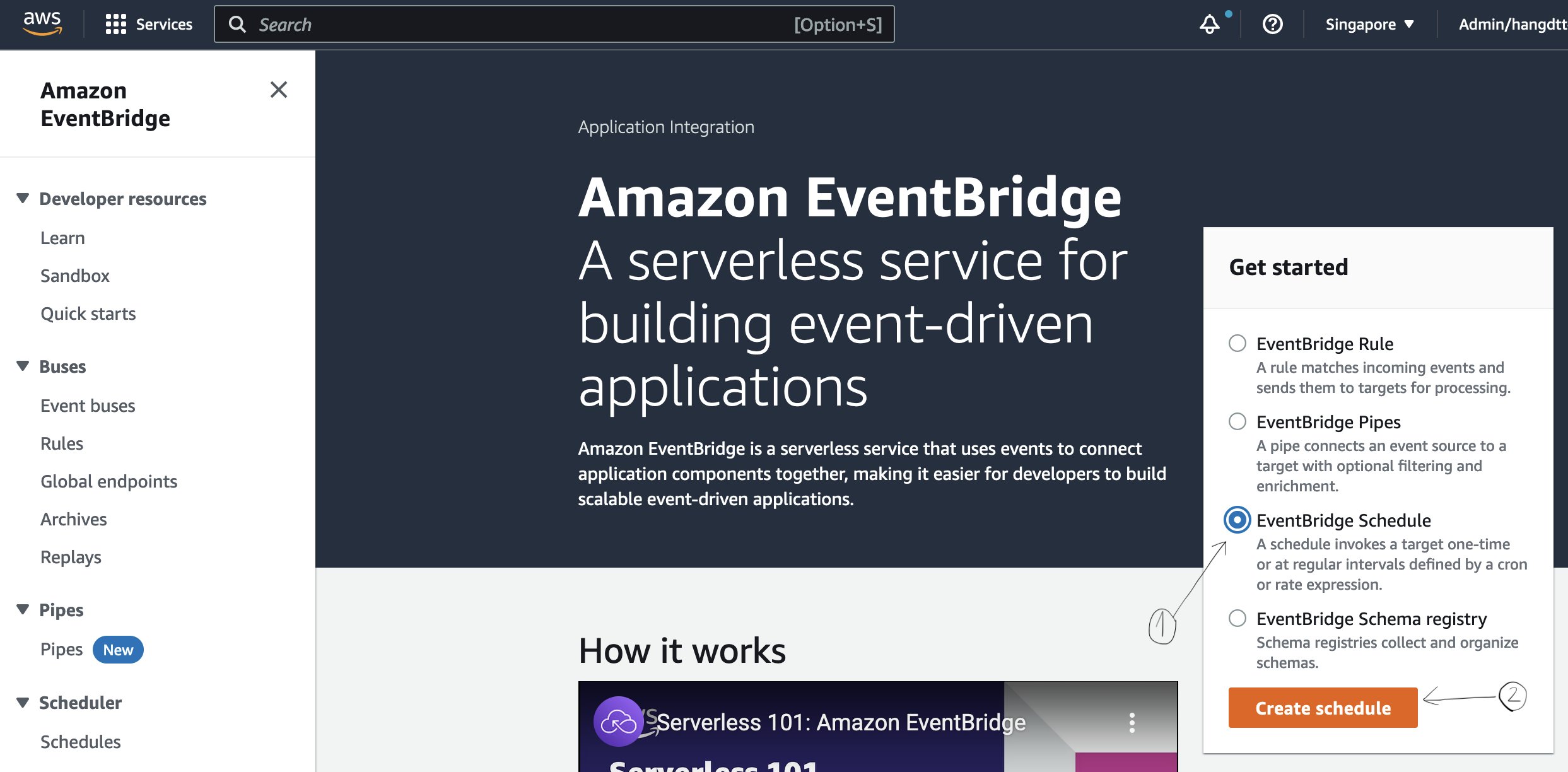
### 4.5. Implement component 1 (Eventbridge Scheduler):

This part shows the implementation by hand (just a portion of the whole job) or automation(full job), specifically, 4.5.1 is created manually and 4.5.2 is created automatically (by code). You need to follow section 4.5.2 to handle all 200 ATMs, while section 4.5.2 handles just 1 ATM but you will have more insight of the code in section 4.5.2.

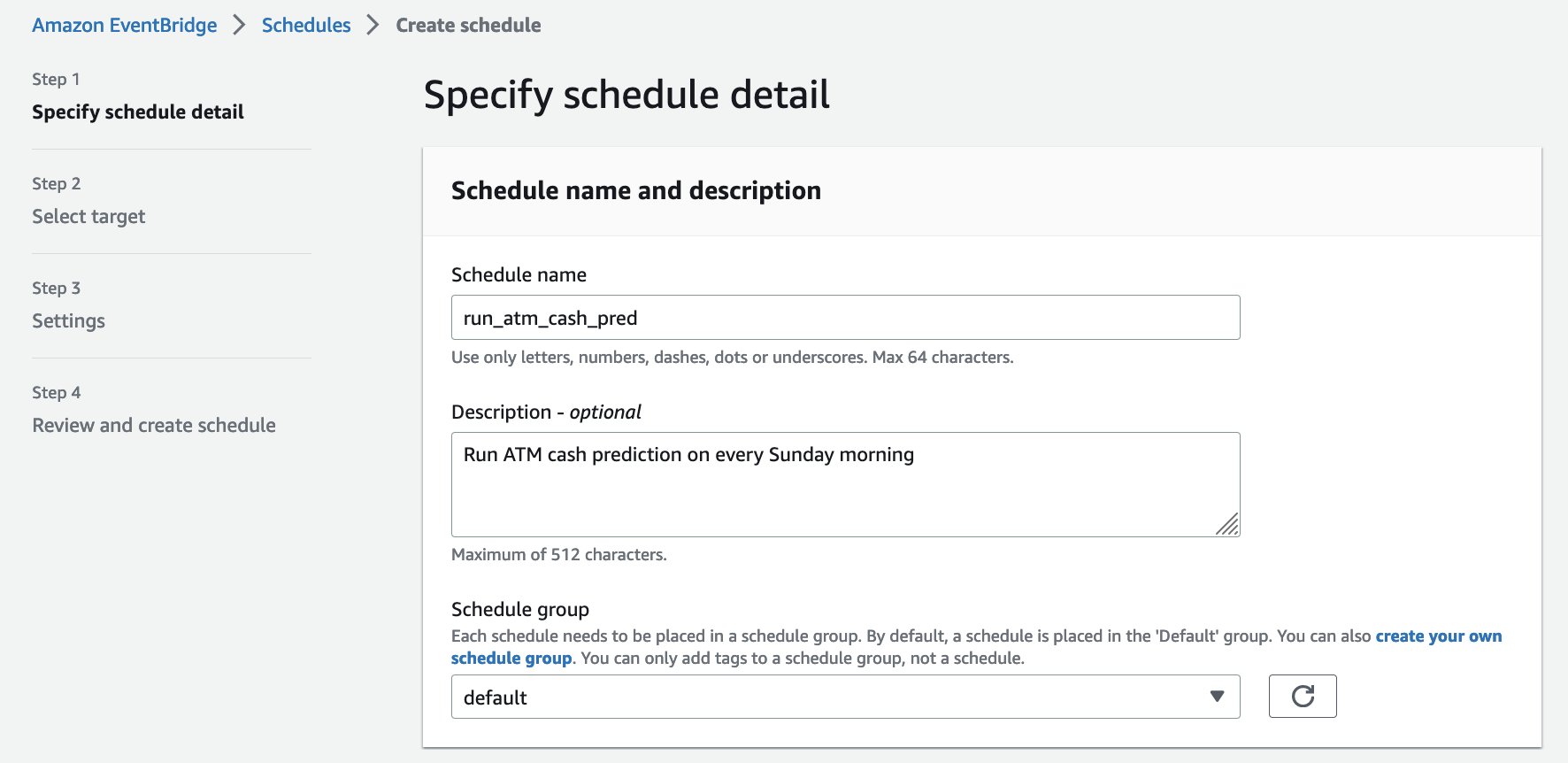
4.5.1. Implement manually:  
From your AWS console, search “eventbridge” and click “Amazon EventBridge” below Services section:



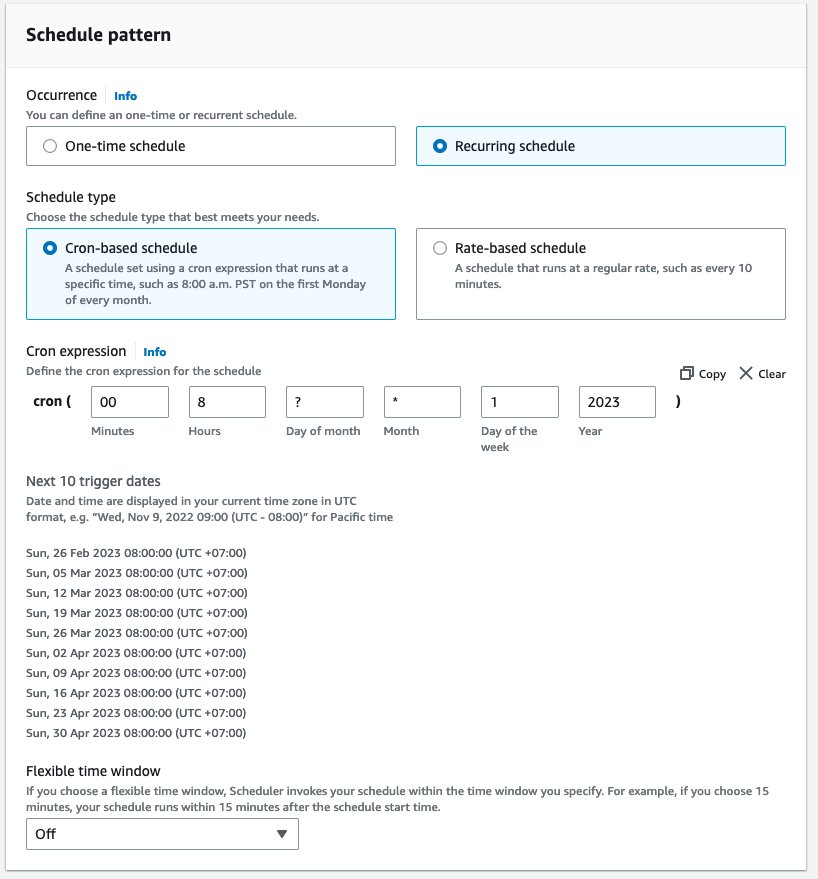
Now you’ll see “Amazon EventBridge” page, click button as shown in step 1 & step 2 as bellow image:



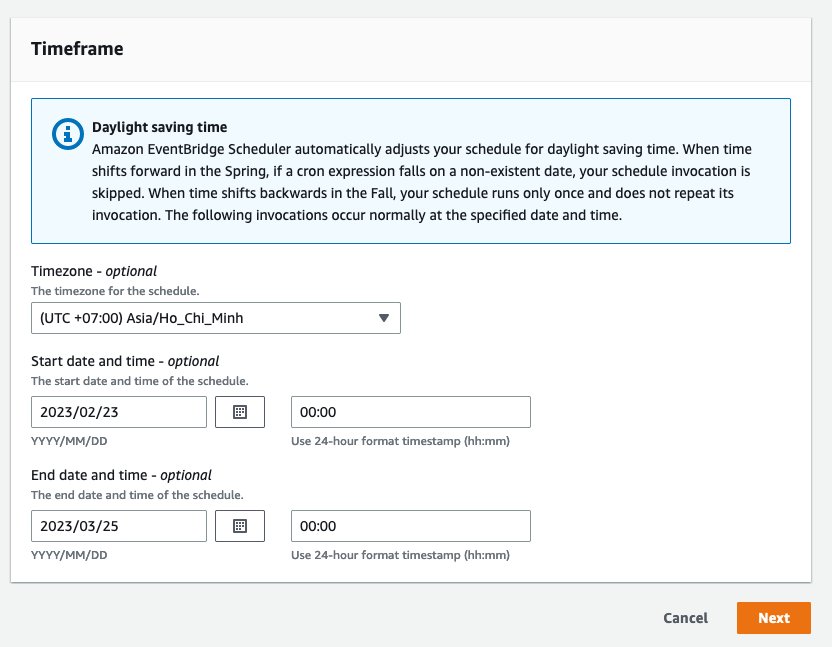
Fill the information as following images:



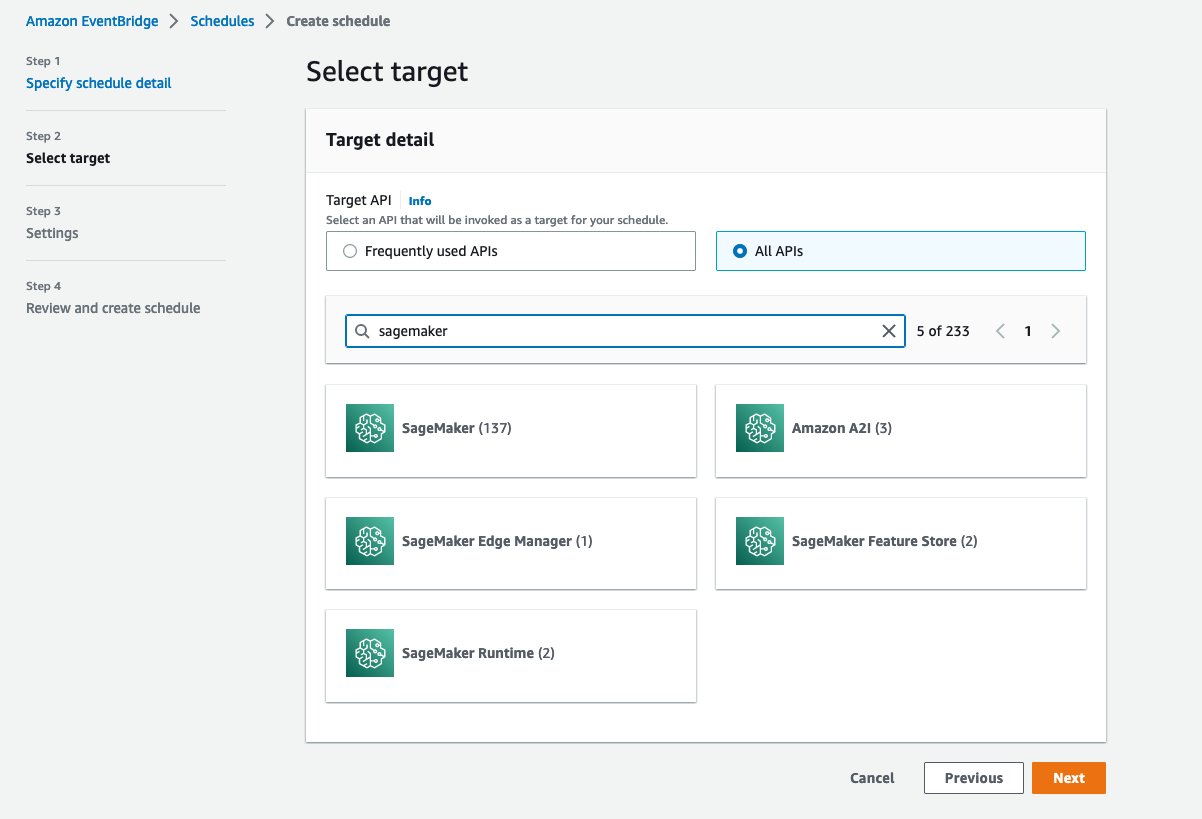
In the following set up Schedule pattern, the schedule will be on every Sunday at 08:00 in 2023 (feel free to change the Cron expression to make a new schedule)



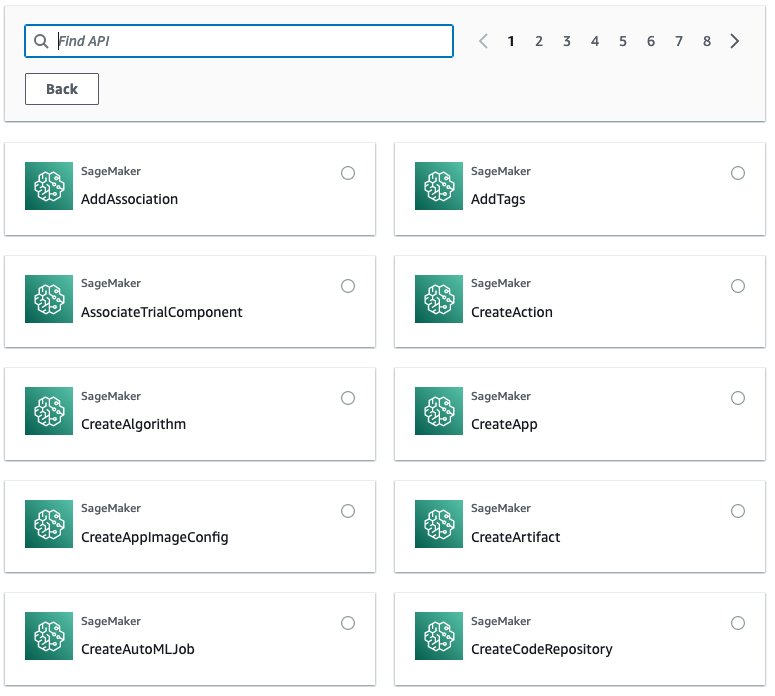
In the Time frame section, select the “Start date and time”, “End date and time” that you want your Sagemaker Processing Job starts and ends date:



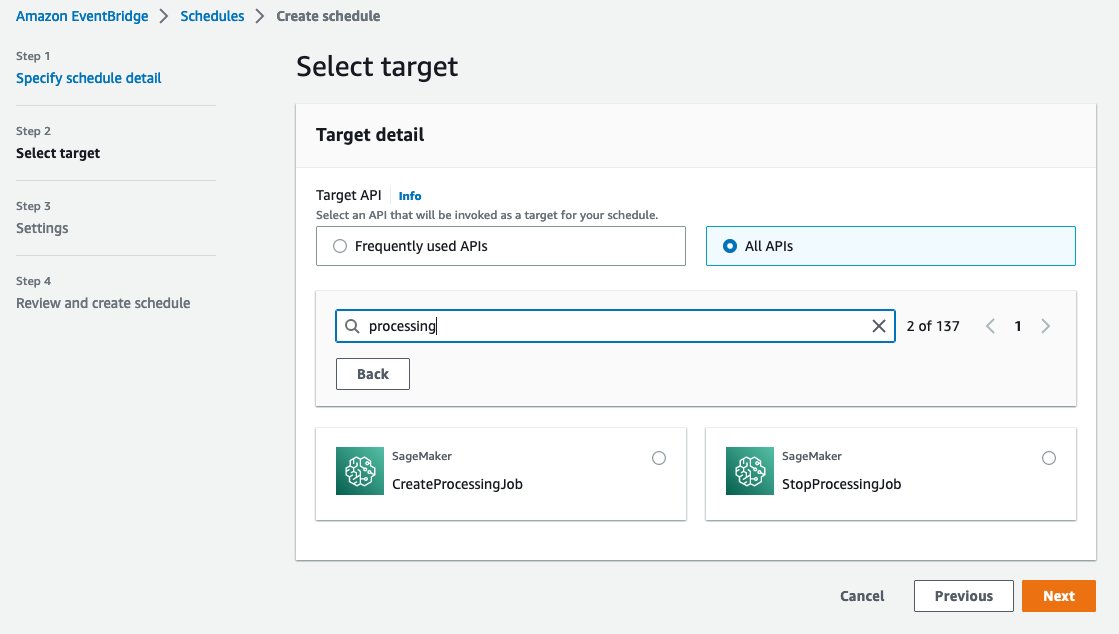
Click “Next“ and you’ll move to the Step 2: Select target as bellow:



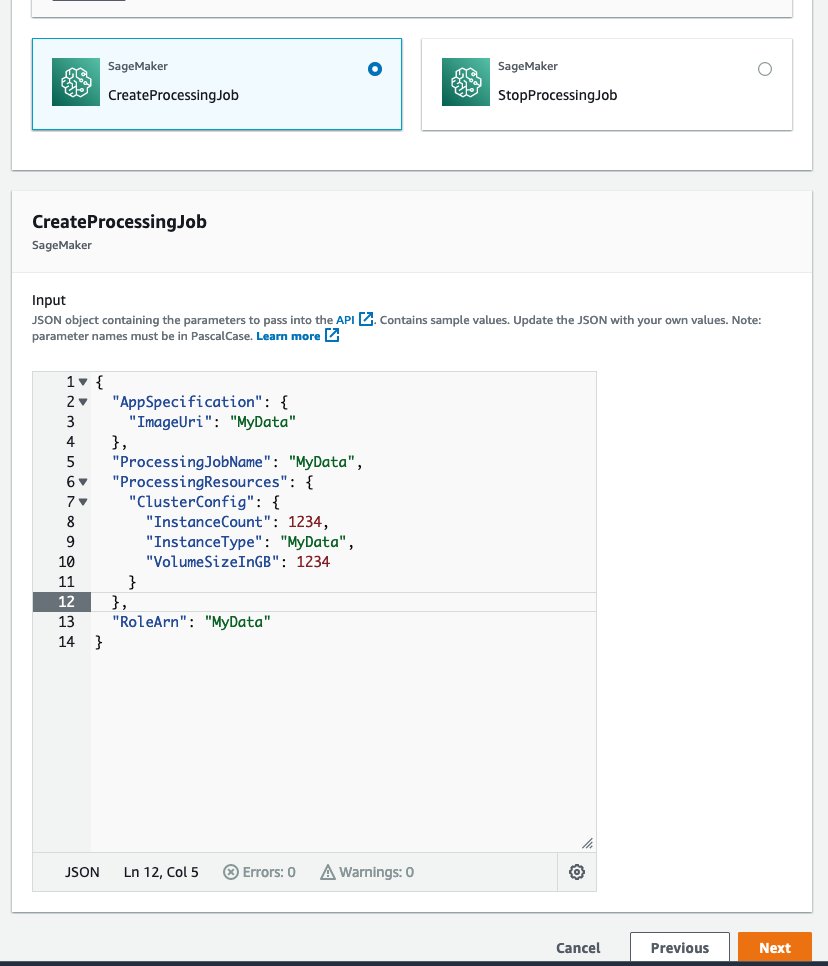
Click “All APIs”, type “sagemaker” in the search bar above, click “SageMaker (137)” box, you’ll see a list of API as bellow:



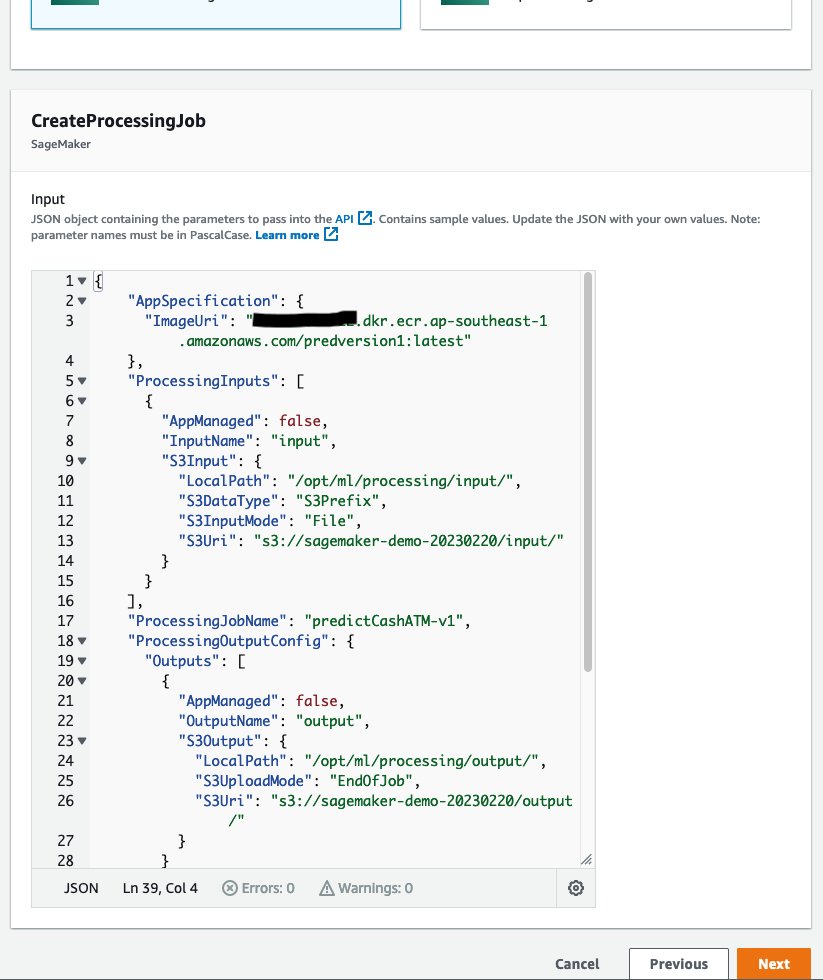
Type “processing” in Find API search bar as bellow:



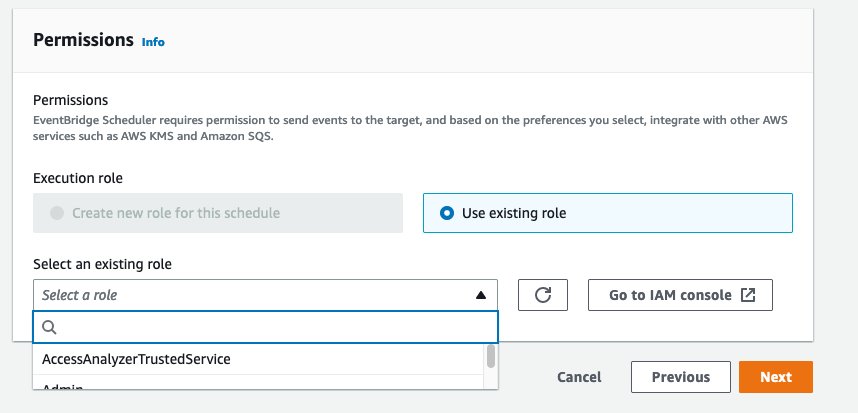
Click SageMaker CreateProcessingJob, you’ll see a JSON live editor as bellow:



Replace the JSON template above with your JSON content in the  
***sagemaker\_processing\_job\_api.json*** as similar as bellow:



Click “Next”, and scroll down to section Permissions as bellow:



Click Select a role and type : “sagemaker-execution”, a suggestion will be shown, you select that suggestion - something like “AmazonSageMaker-ExecutionRole-xxx” as bellow:

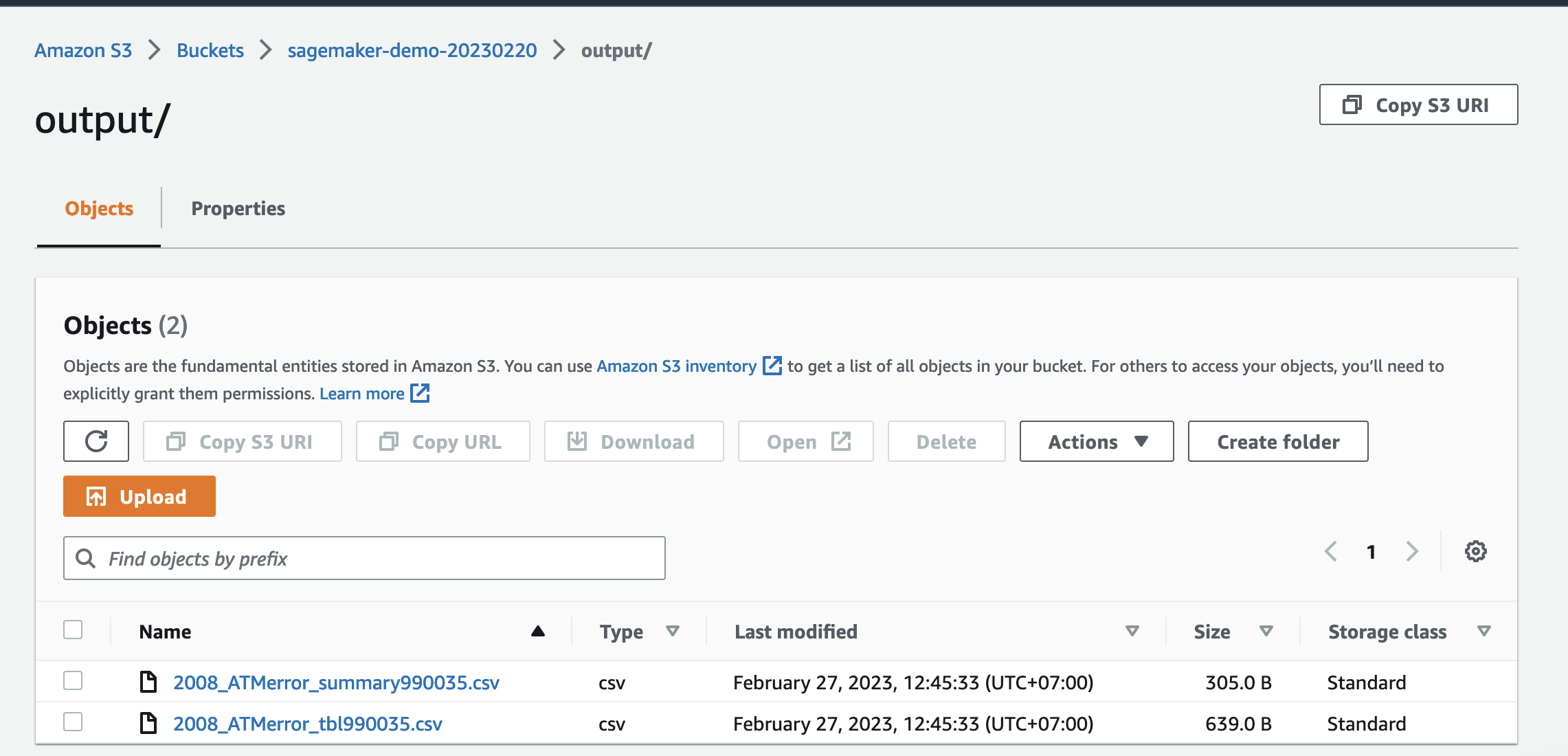


After selecting that role, click Next button.

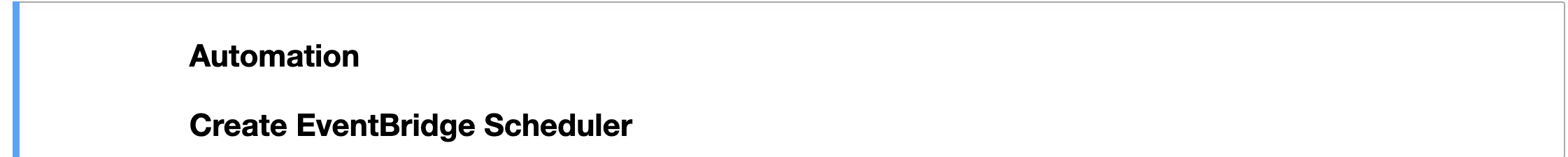
Now you’re viewing Step 4: Review and create schedule, scroll down and click button : “Create schedule”

→ A green banner which writes “**Your schedule run\_atm\_cash\_pred is being created**” shows after → that means you do create successfully scheduler.

Now you are completed for this manual section.

Check “output” in S3 after running Sagemaker processing job successfully:

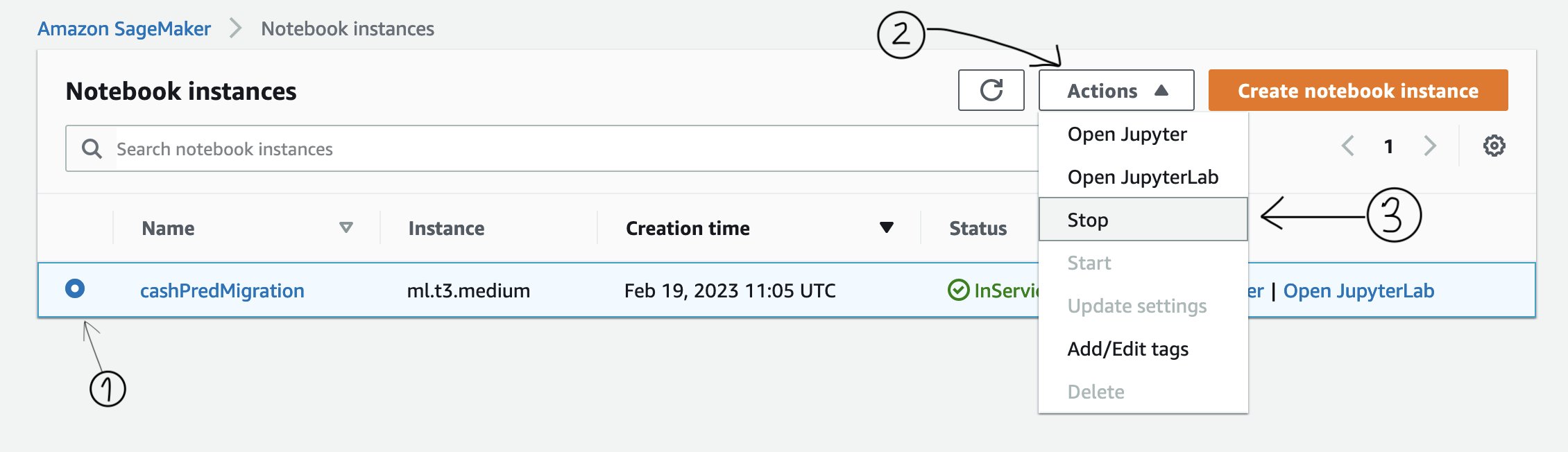
#### 4.5.2. Implement automatically:

Open your jupyternotebook (open Deployment in section 4.3. Implement component 4 (Amazon ECR)) and run all the cells below “Create EventBridge Scheduler”: 

### 4.6. Clean up un-used resouce:

After this Notebook run all cells successfully,

Stop it by selecting 3 buttons as following image :



## 5. Further improvement:

If BIDV team wants to improve running time of the current algorithm (as mentioned in option 2 in section 2.2, the algorithm run time will be ~ 15mins) then Data scientist needs to refactor the code to run code in Parallel in the [ml.c5.18xlarge](https://ap-southeast-1.console.aws.amazon.com/servicequotas/home/services/sagemaker/quotas/L-2D8CD70A) instance (ref link: <https://www.r-bloggers.com/2017/10/running-r-code-in-parallel/>)

End of documentation.

