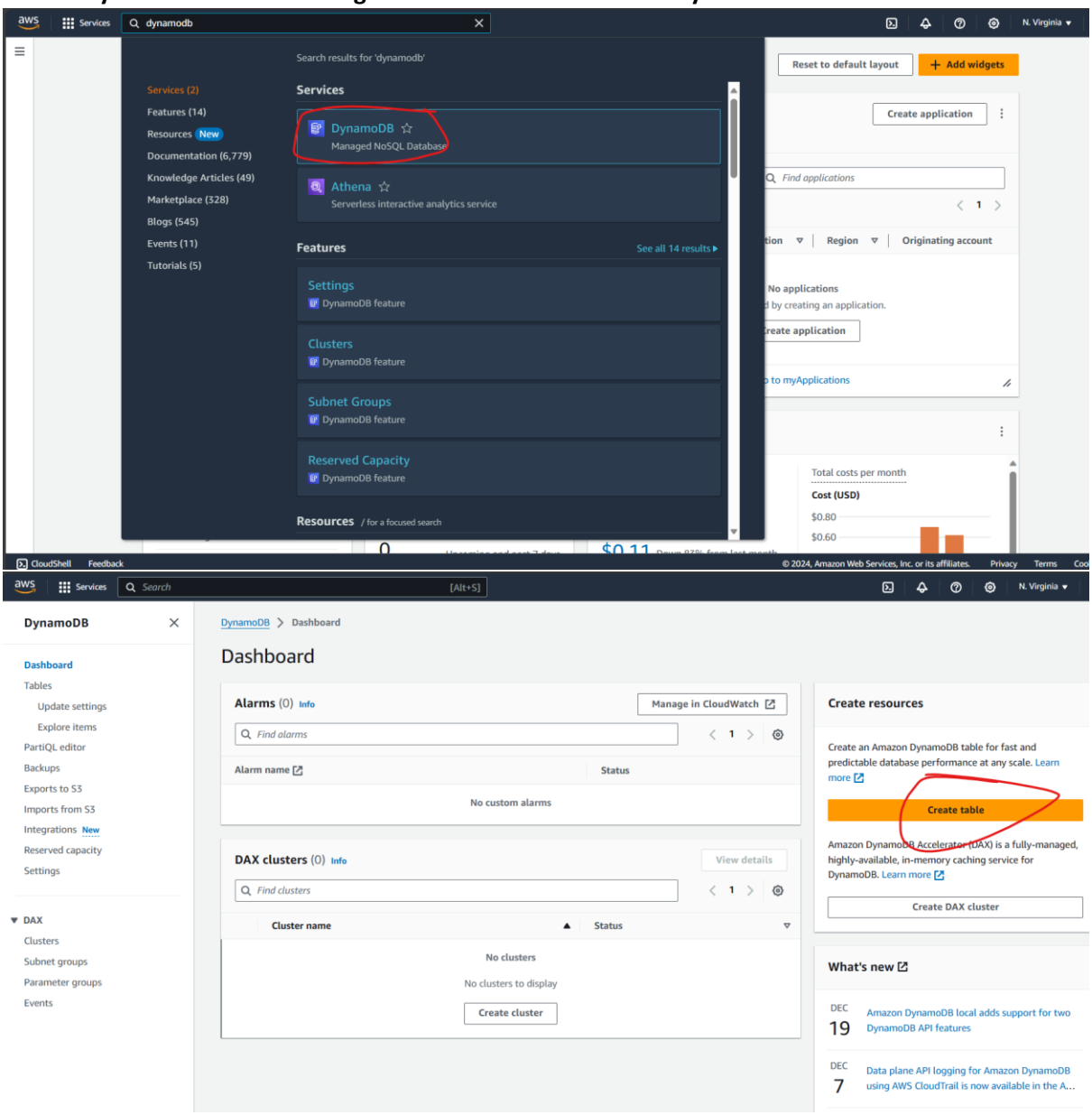
DynamoDB Setup:

➔ **Search DynamoDB in Aws Management Console and click on DynamoDB service.**



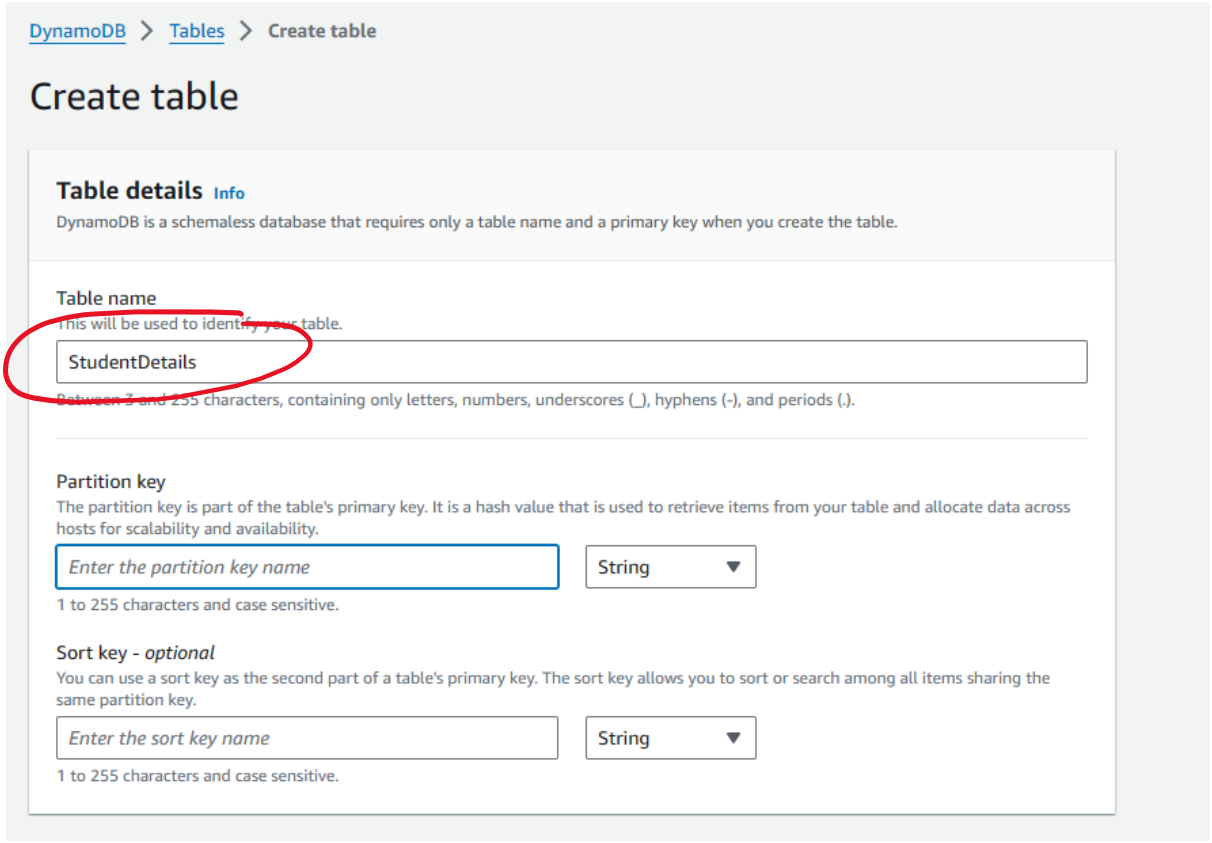
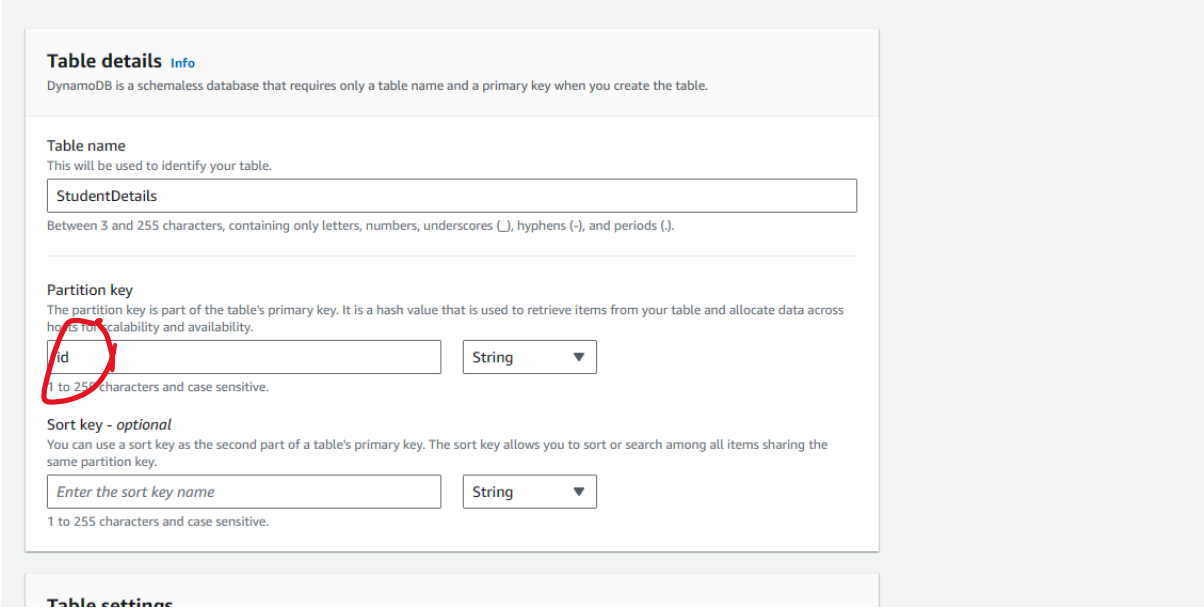
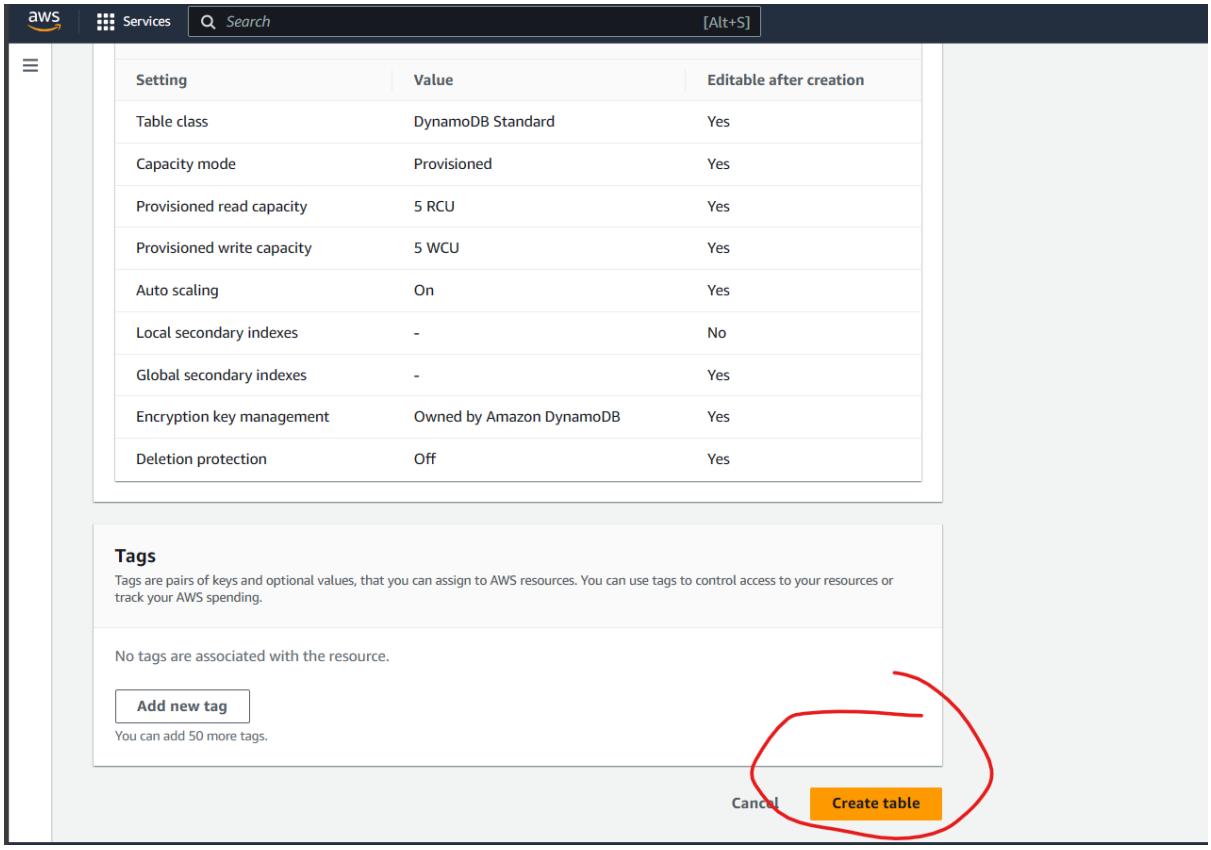


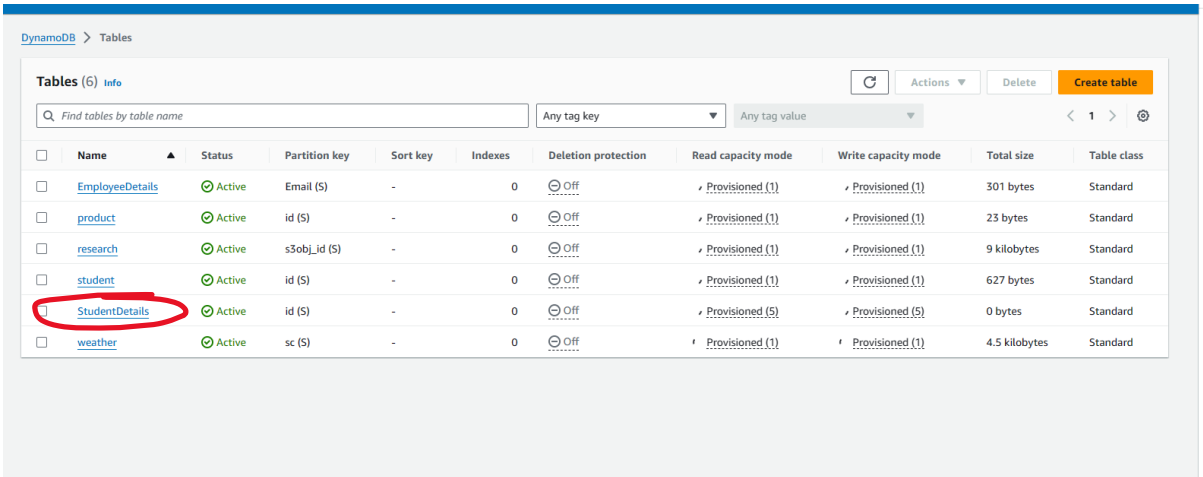
Table Name : StudentDetails



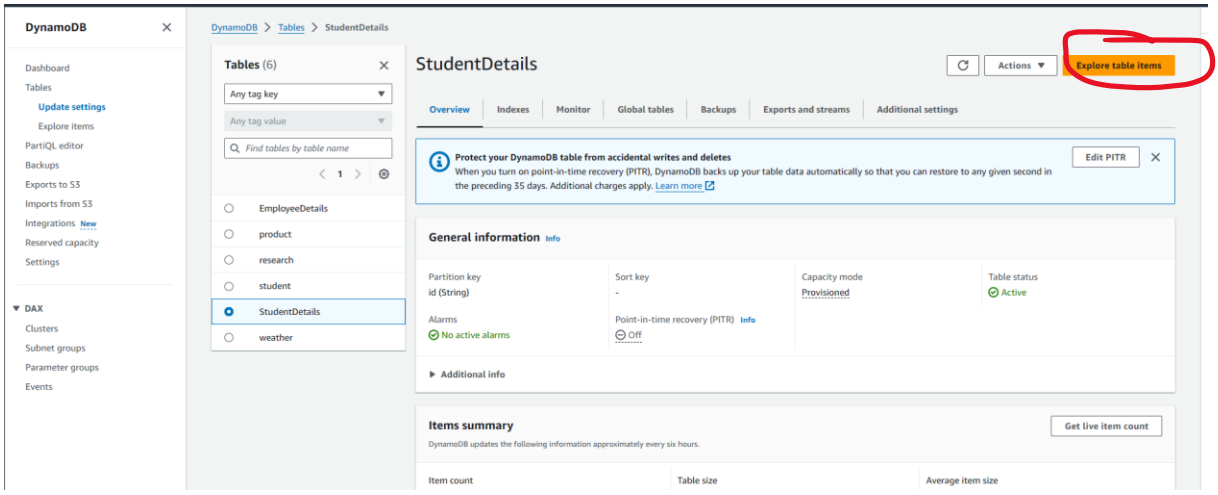
Partition key : id

Scroll Down and click create table

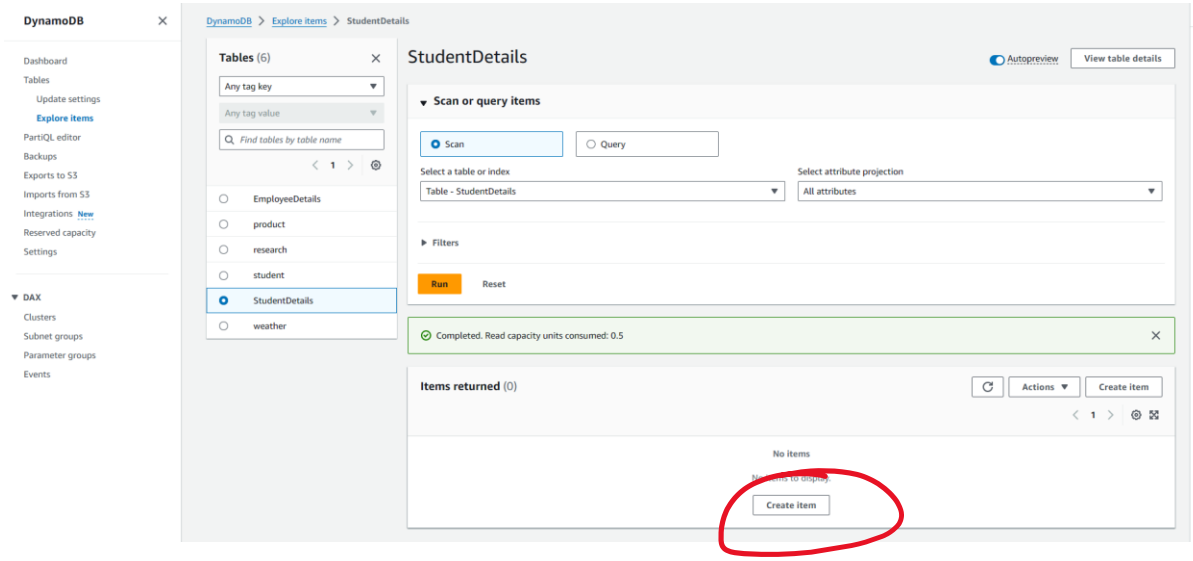




Click on StudentDetails Table



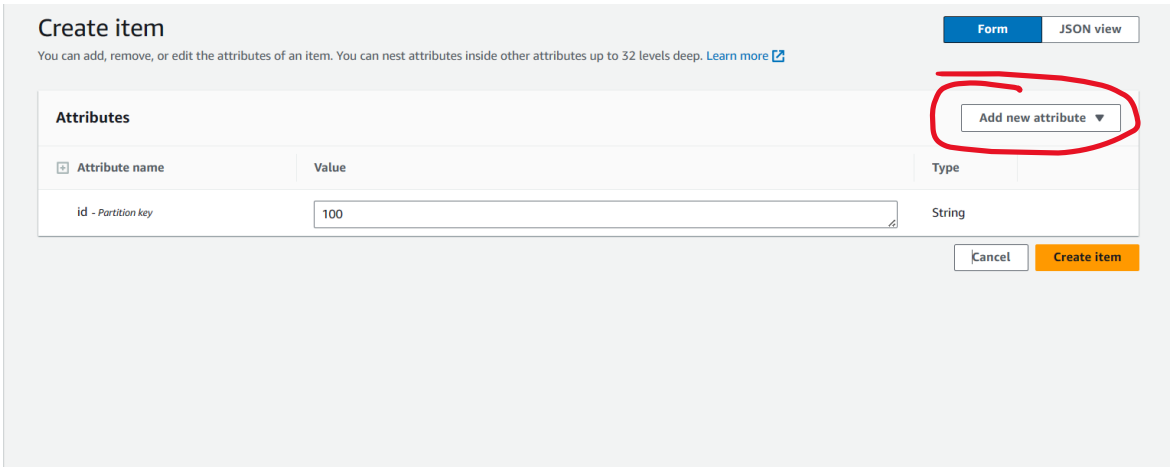
Click on Explore table items.

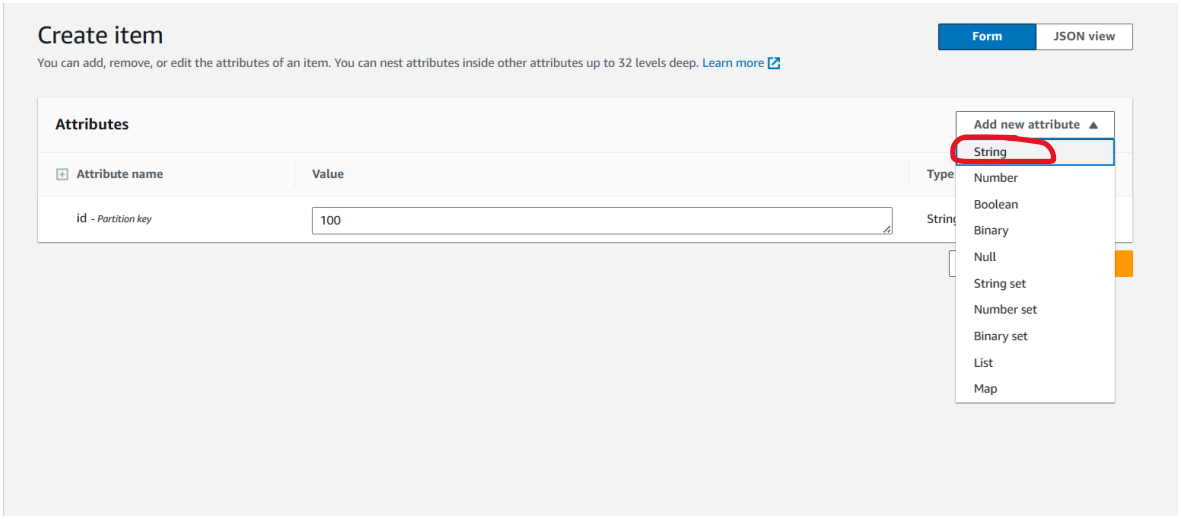


Click on Create item.

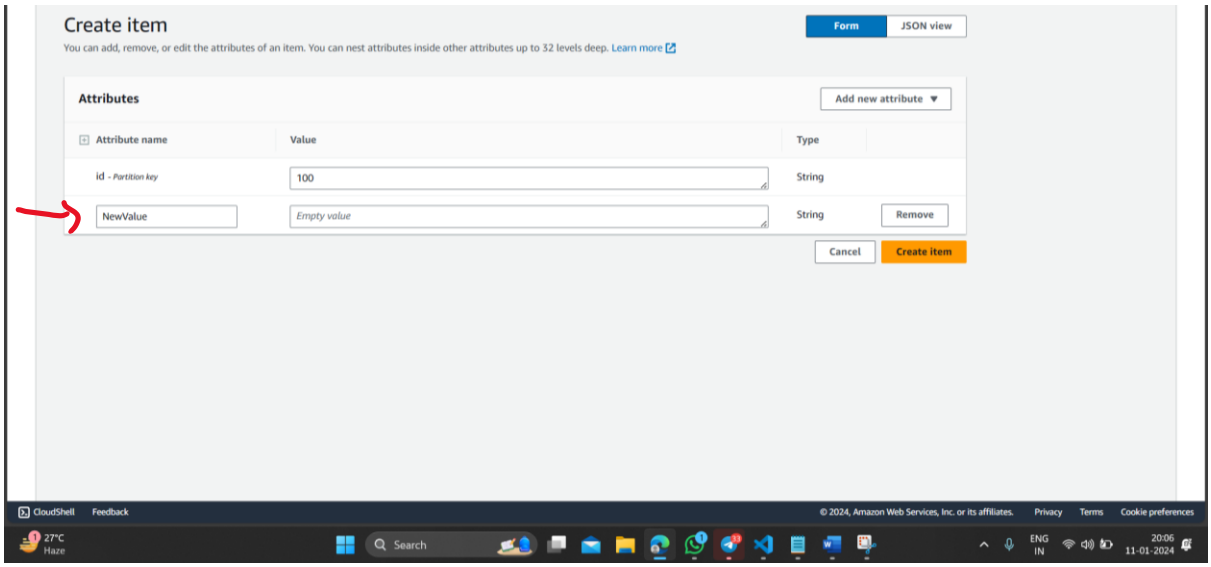
Lets add an item.

Id : 100



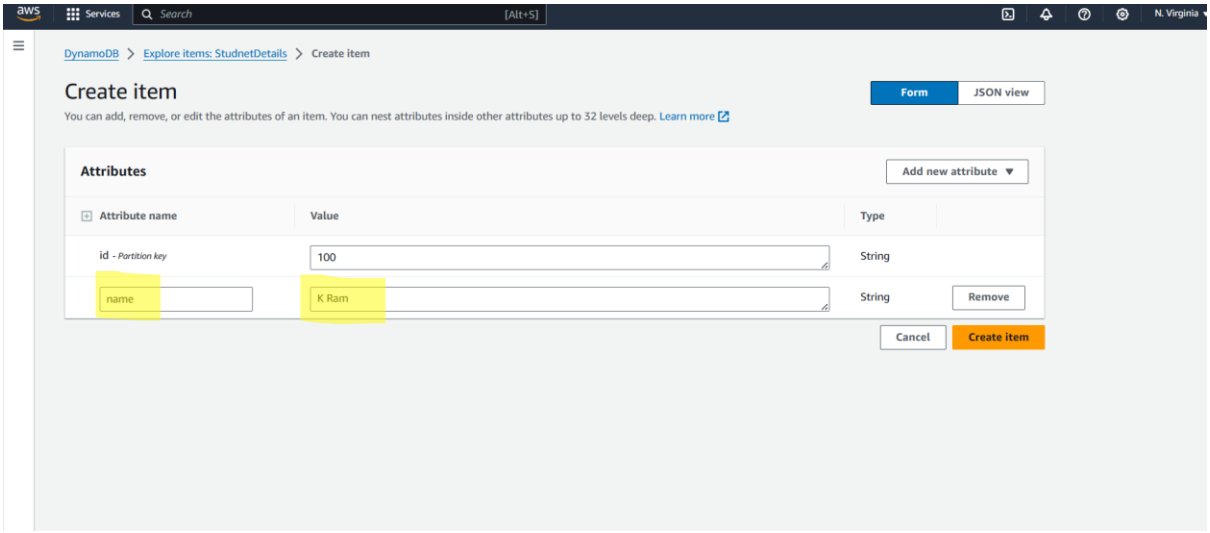


Click on String



Replace NewValue with name

name : K Ram



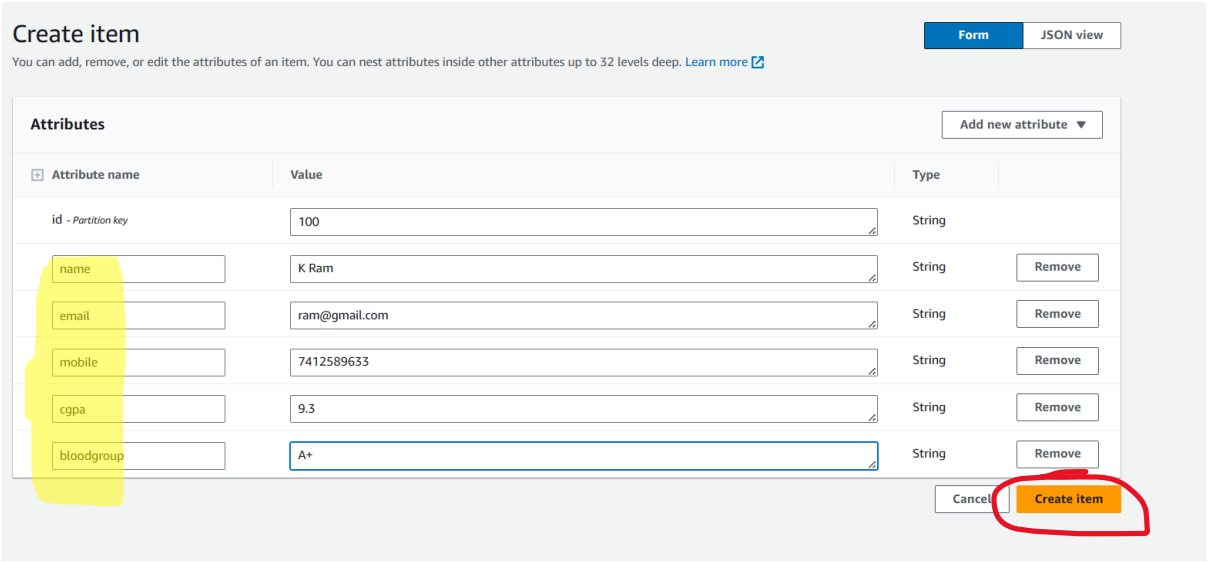
Repeat the process for the below attributes

email : ram@gmail.com

mobile : 7412589633

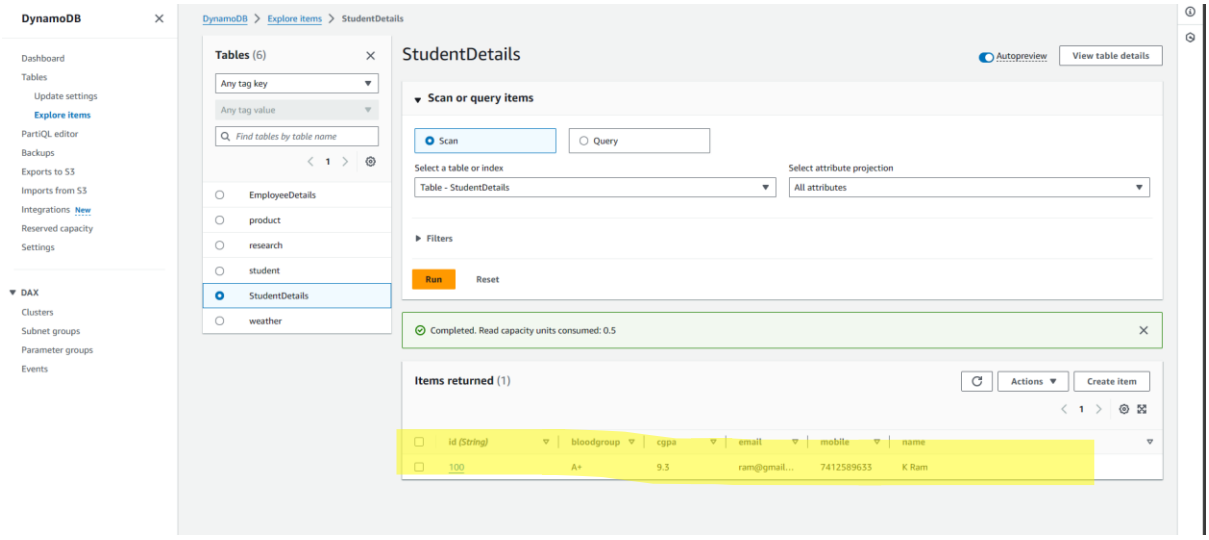
cgpa : 9.3

bloodgroup : A+



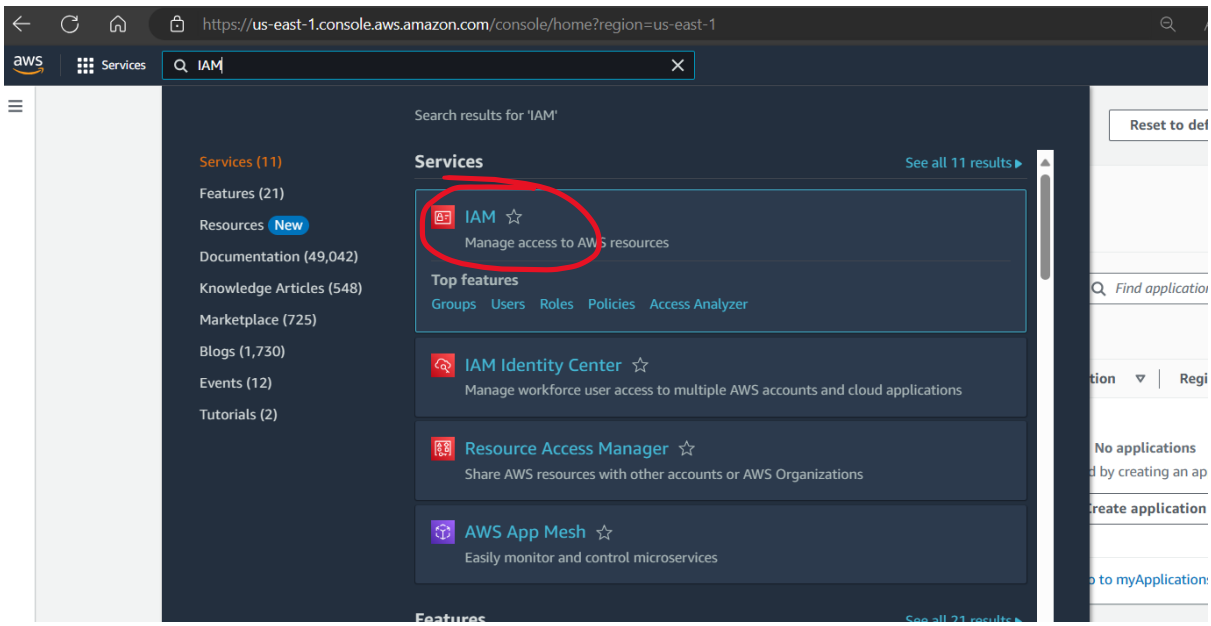
Make sure you provide same attributes (same spelling and with lowercase)

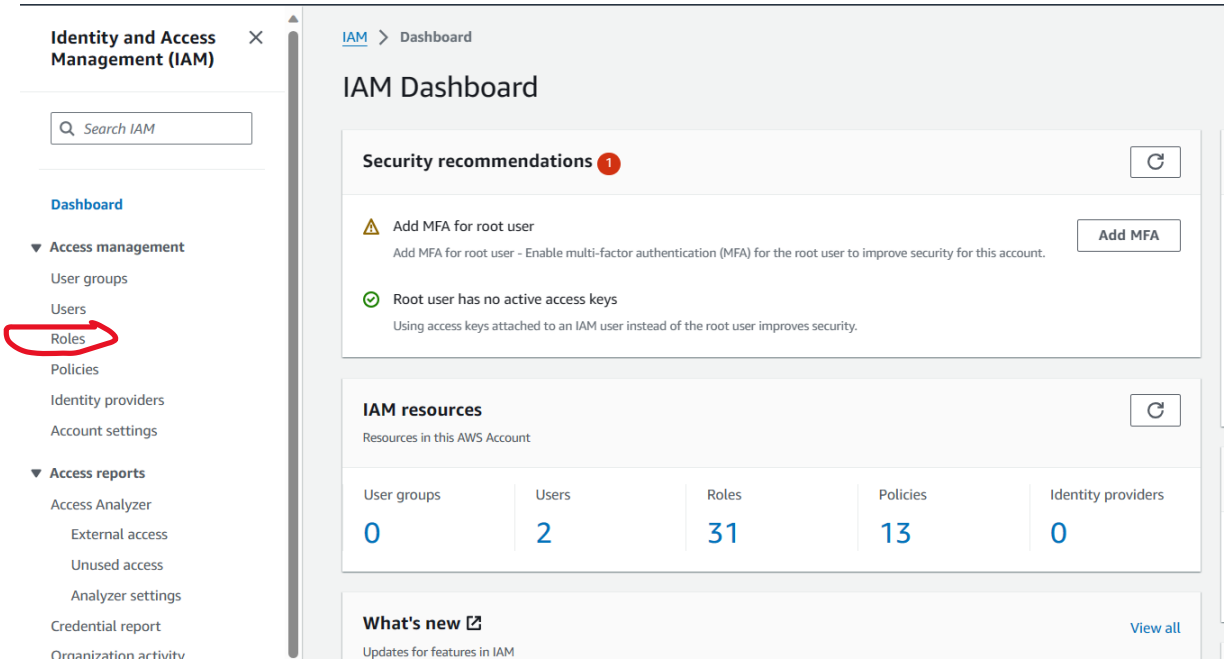
After filling the attributes and values click on create item.



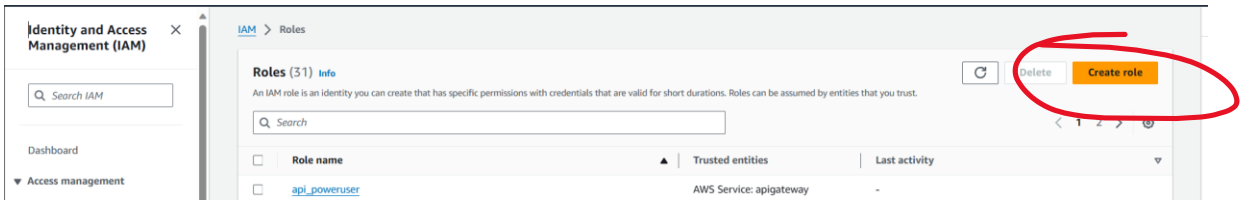
IAM Roles

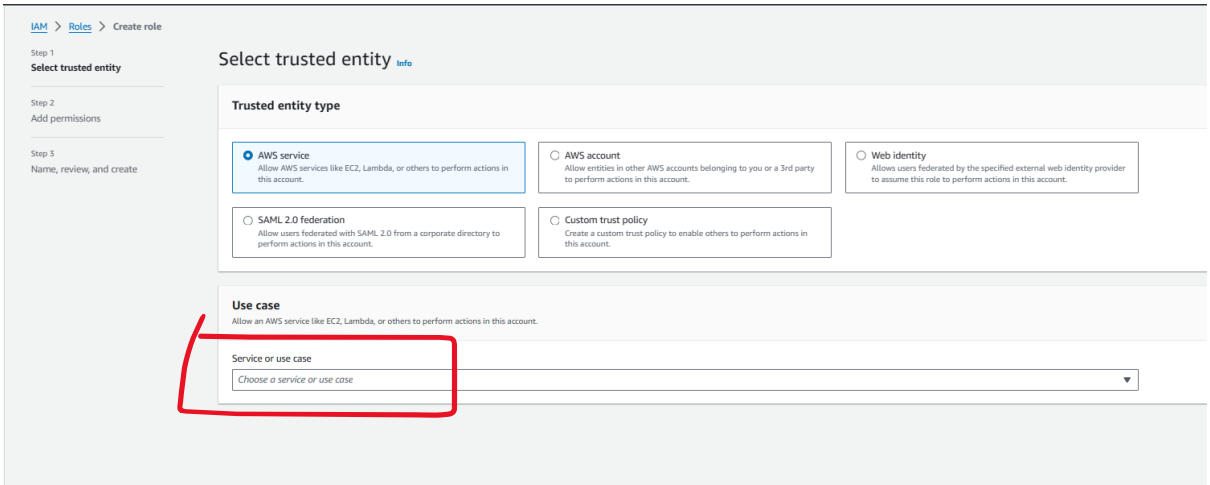
1.To get the student details from dynamodb using lambda function.

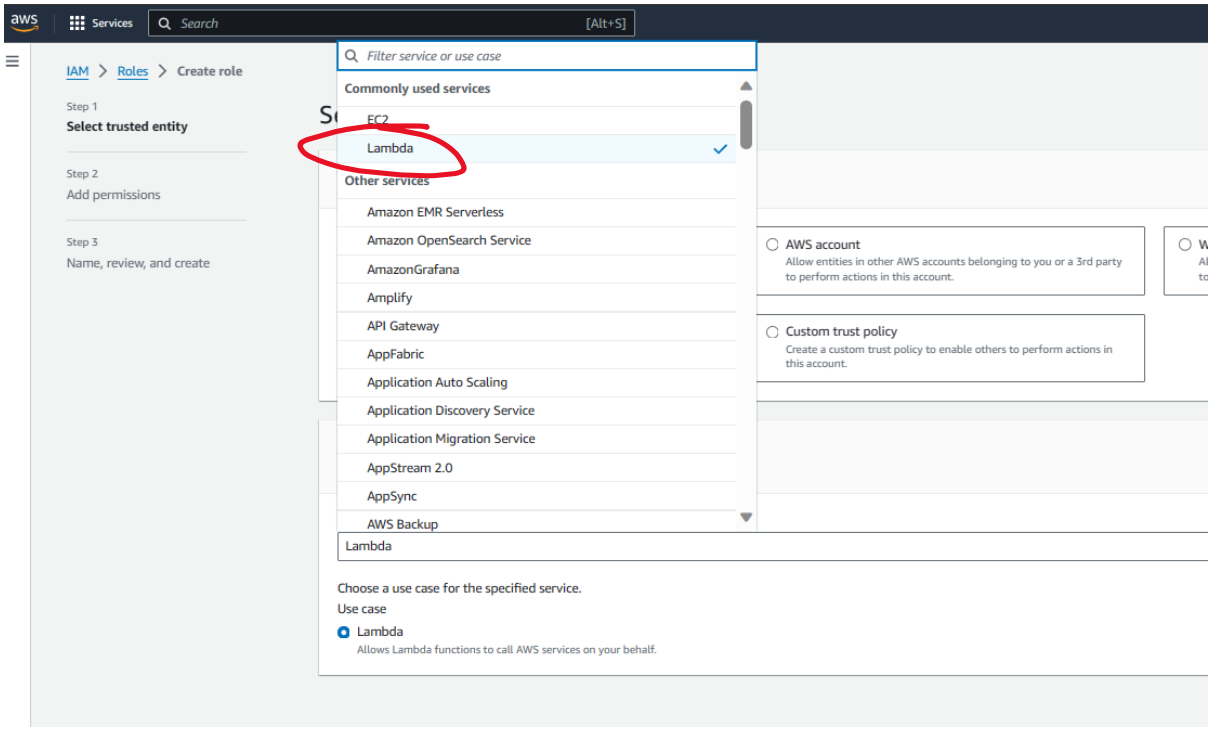


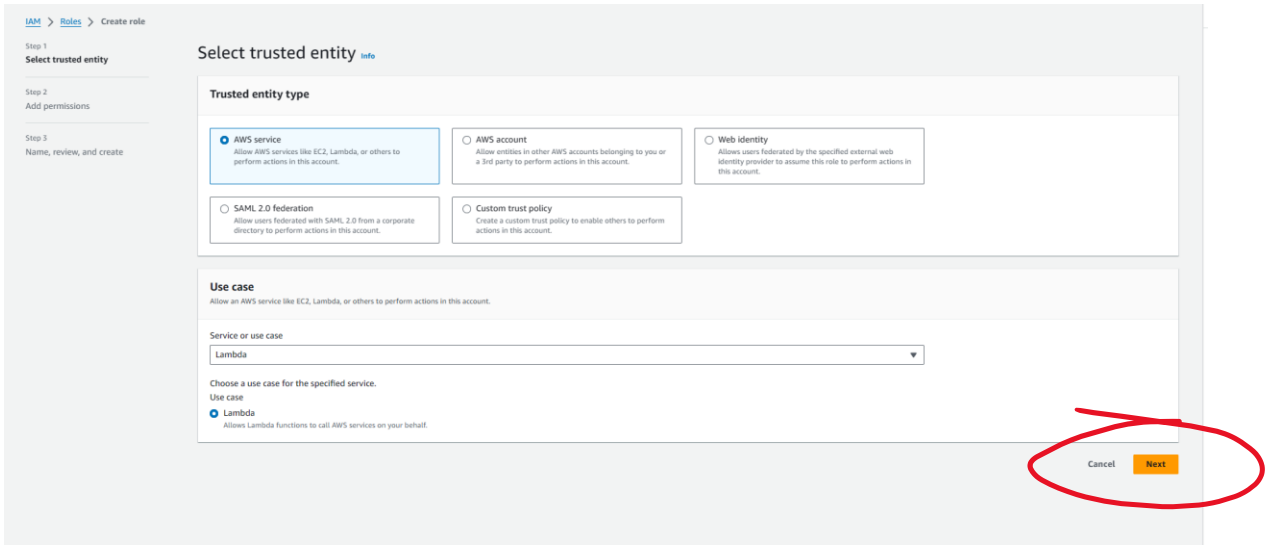


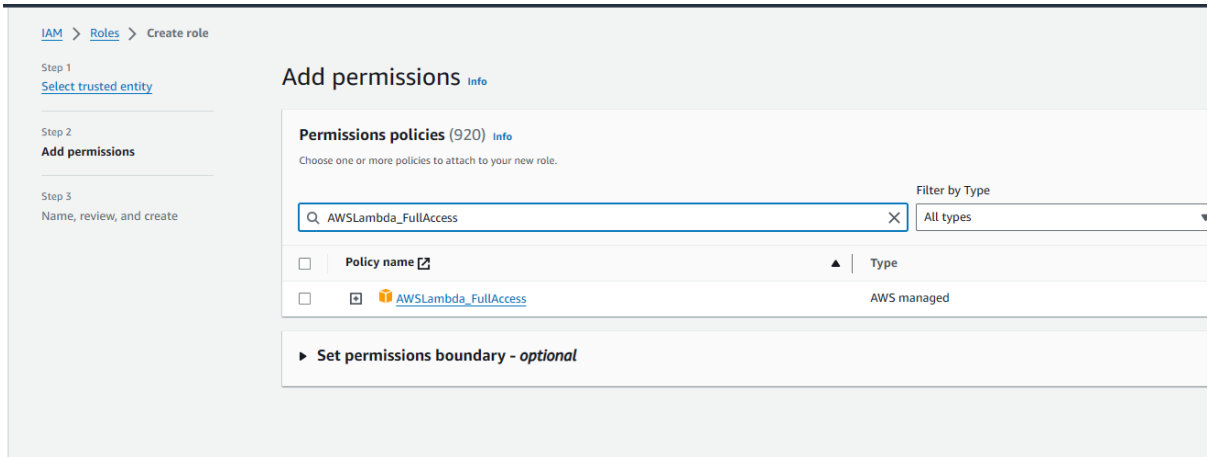
click on Roles



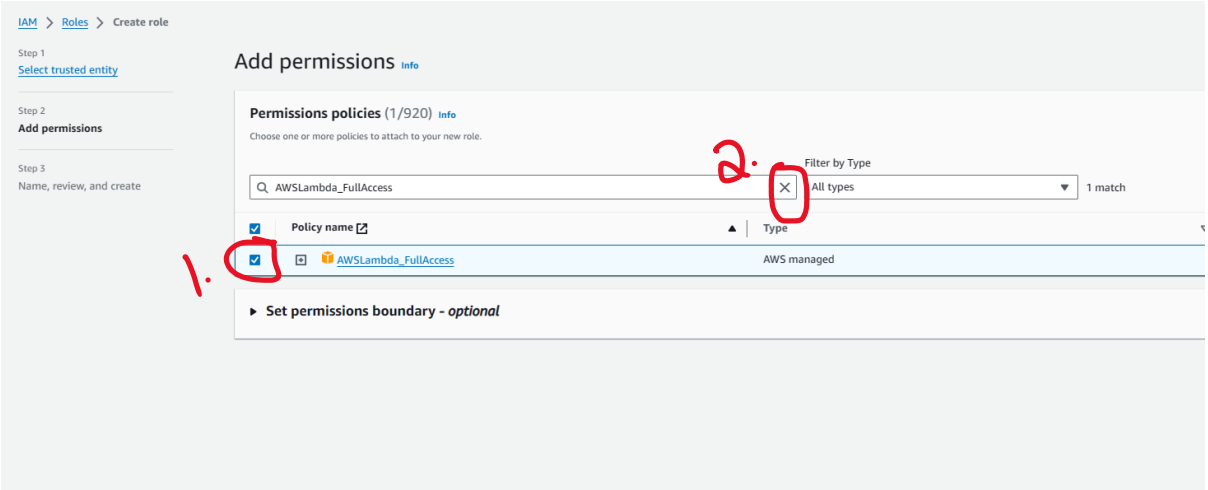


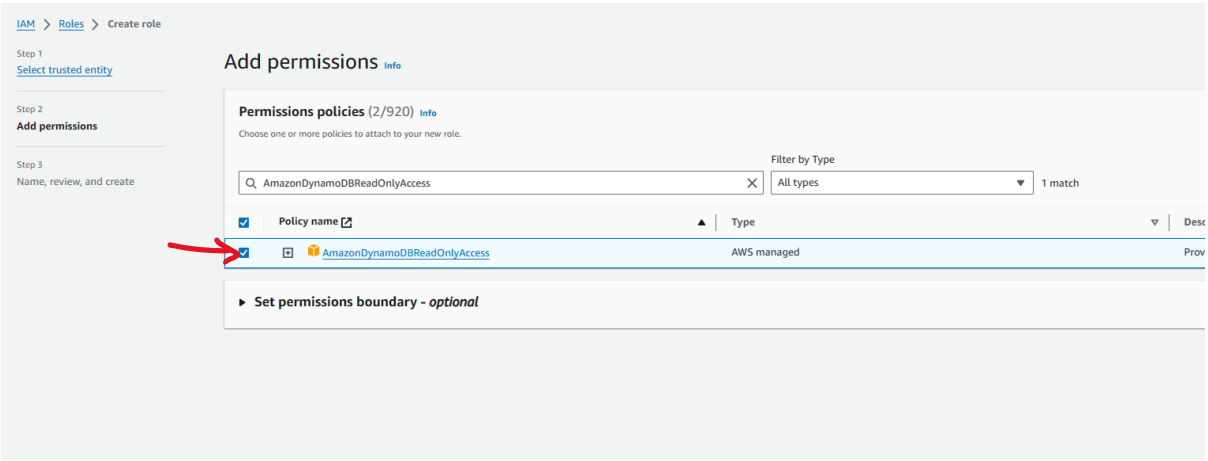

aws
Services
Search
[Alt+S]
IAM > Roles > Create role
Step 1
Select trusted entity
Step 2
Add permissions
Step 3
Name, review, and create
Filter service or use case
Commonly used services
EC2
Lambda
Other services
Amazon EMR Serverless
Amazon OpenSearch Service
AmazonGrafana
Amplify
API Gateway
AppFabric
Application Auto Scaling
Application Discovery Service
Application Migration Service
AppStream 2.0
AppSync
AWS Backup
AWS account
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
Custom trust policy
Create a custom trust policy to enable others to perform actions in this account.
Lambda
Choose a use case for the specified service.
Use case
Lambda
Allows Lambda functions to call AWS services on your behalf.


IAM > Roles > Create role
Step 1
Select trusted entity
Step 2
Add permissions
Step 3
Name, review, and create
Select trusted entity Info
Trusted entity type
AWS service
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
AWS account
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
Web identity
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
SAML 2.0 federation
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
Custom trust policy
Create a custom trust policy to enable others to perform actions in this account.
Use case
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.
Service or use case
Lambda
Choose a use case for the specified service.
Use case
Lambda
Allows Lambda functions to call AWS services on your behalf.
Cancel
Next

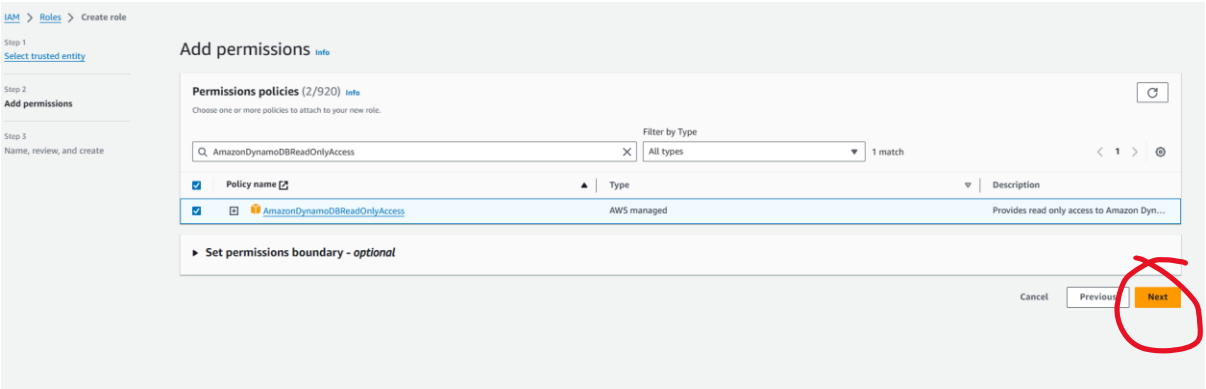

IAM > Roles > Create role
Step 1
Select trusted entity
Step 2
Add permissions
Step 3
Name, review, and create
Add permissions Info
Permissions policies (920) Info
Choose one or more policies to attach to your new role.
Filter by Type
AWSLambda_FullAccess
All types
Policy name
Type
AWSLambda_FullAccess
AWS managed
Set permissions boundary - optional

Search for AWSLambda_FullAccess. Click the check box

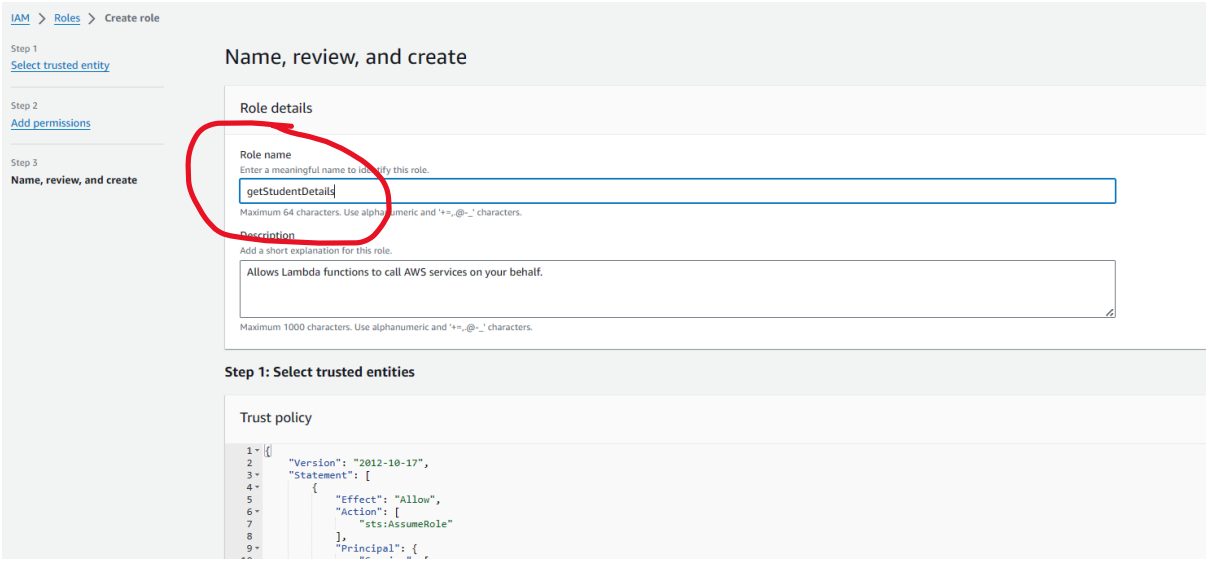




Search for AmazonDynamoDBReadOnlyAccess. Click the check box



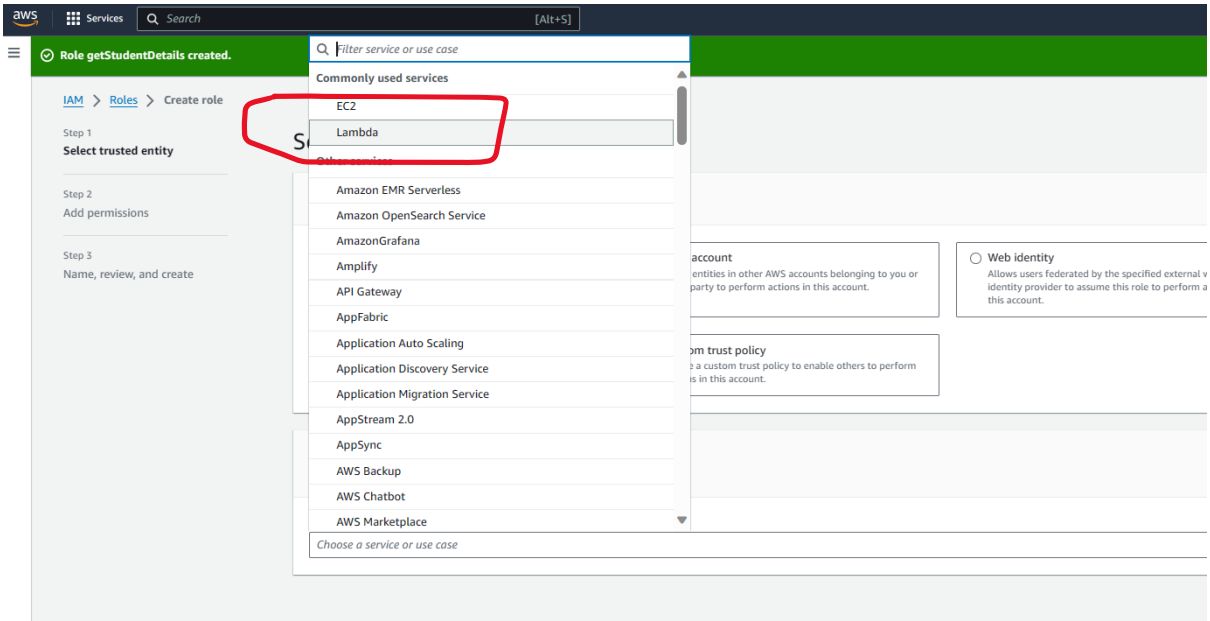
Click on next.



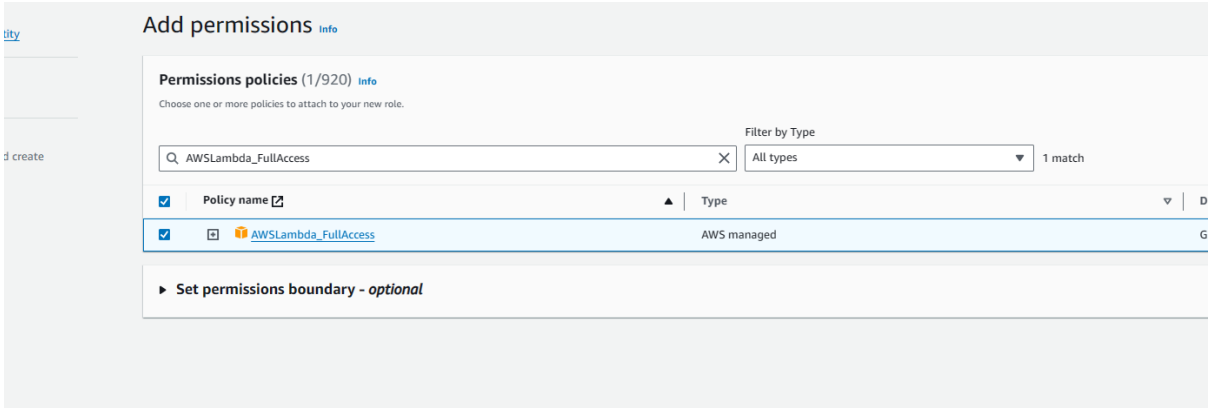
Role Name : getStudentDetails

Scoll down and click create role.

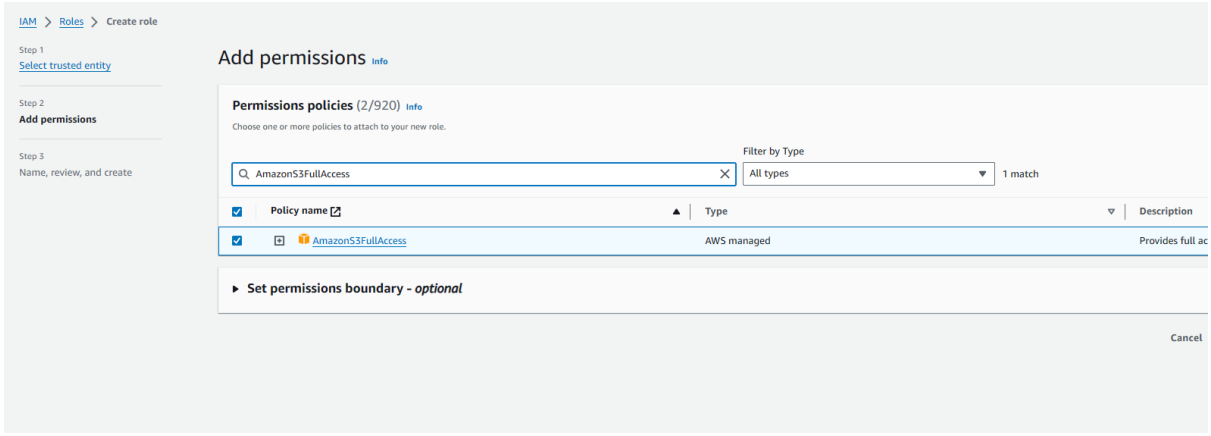
2. To save the student details to dynamodb using lambda function.



Click next

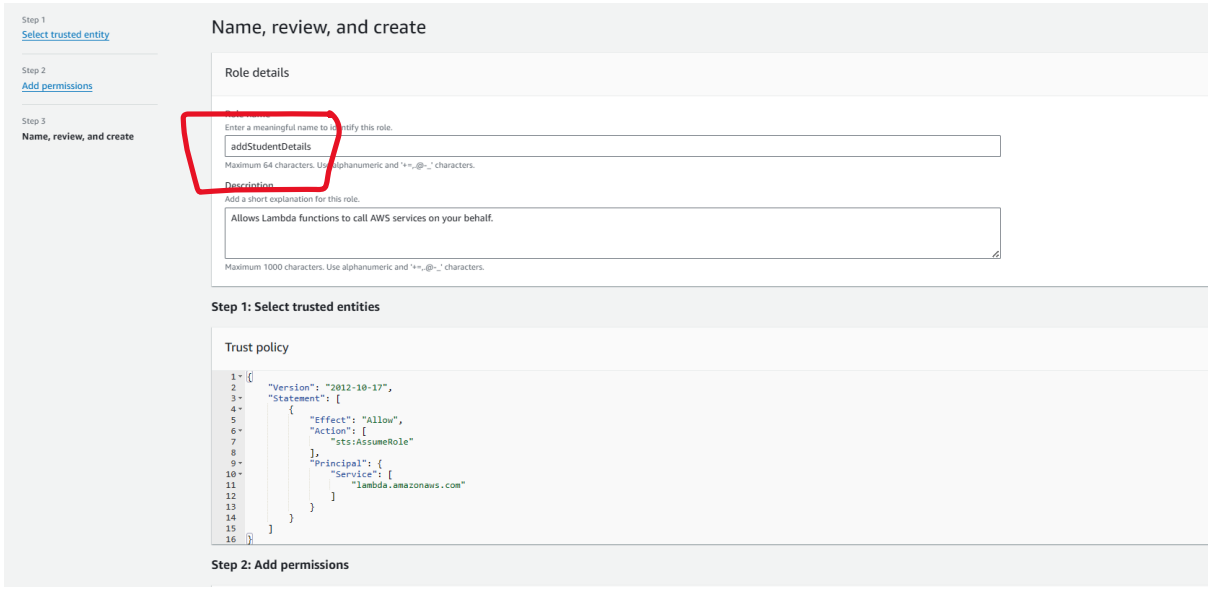


Search for AWSLambda_FullAccess



Search for AmazonS3FullAccess

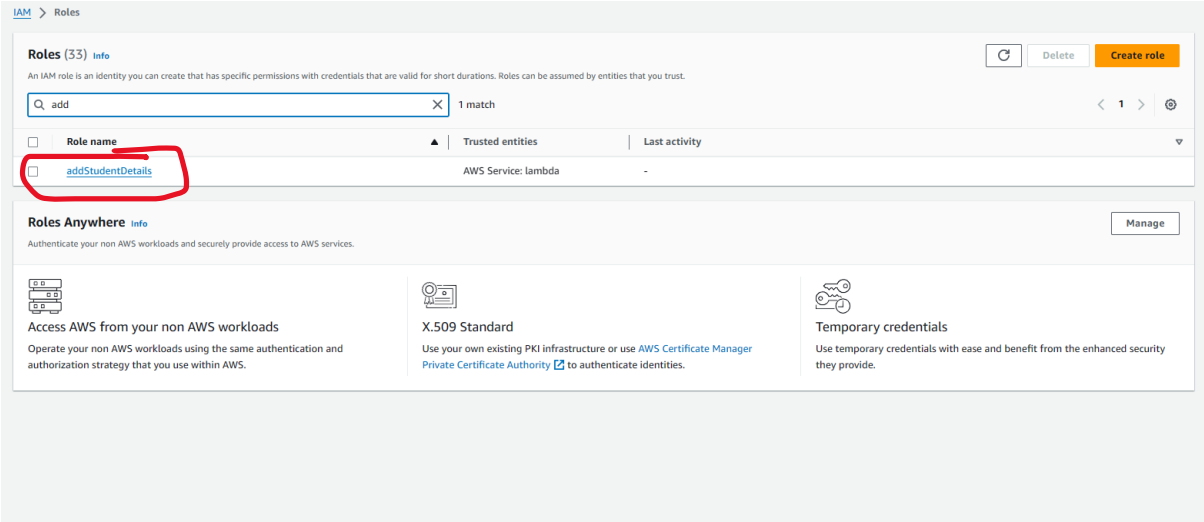
Click on next.



Role name : addStudentDetails

Scroll down and Click on create role.

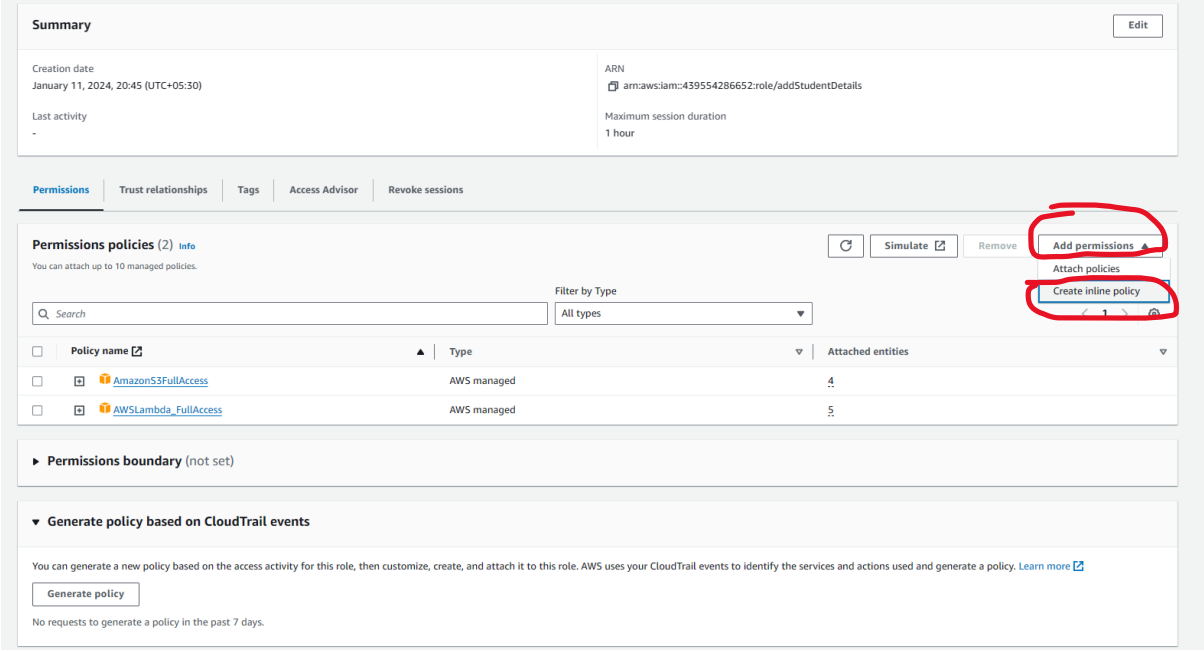
Go to Roles page and search for addStudentDetails role.

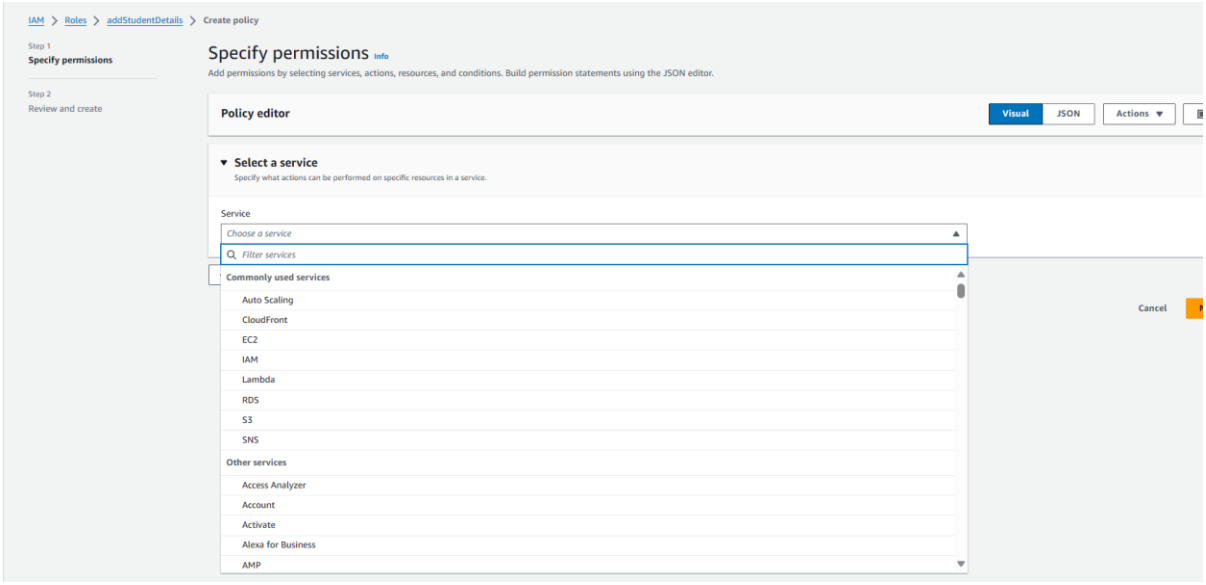


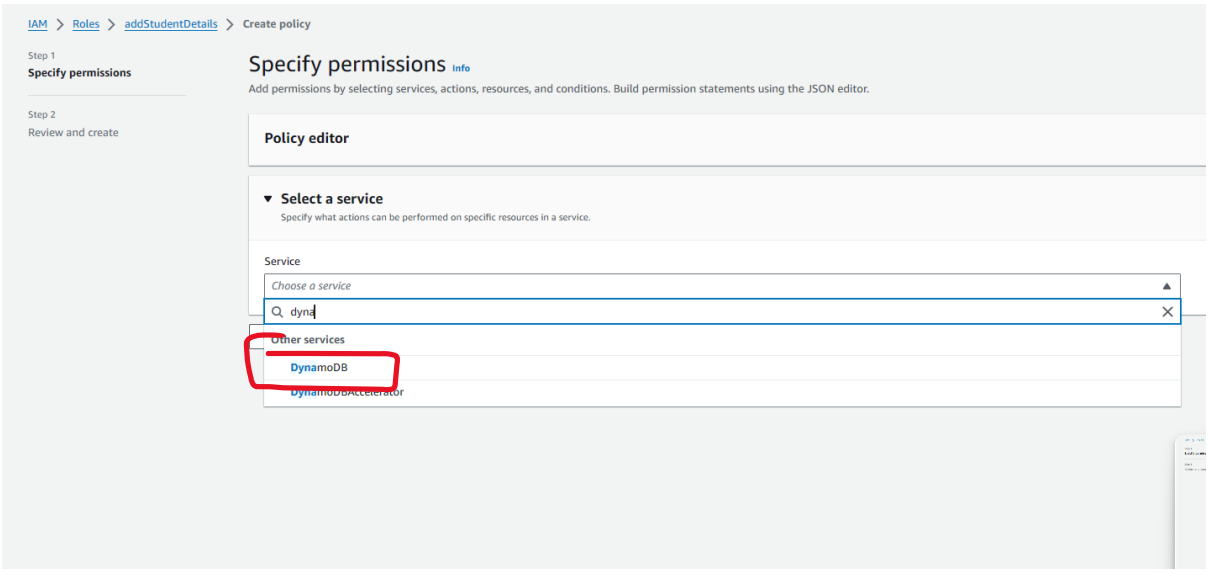
Click on the role addStudentDetails.

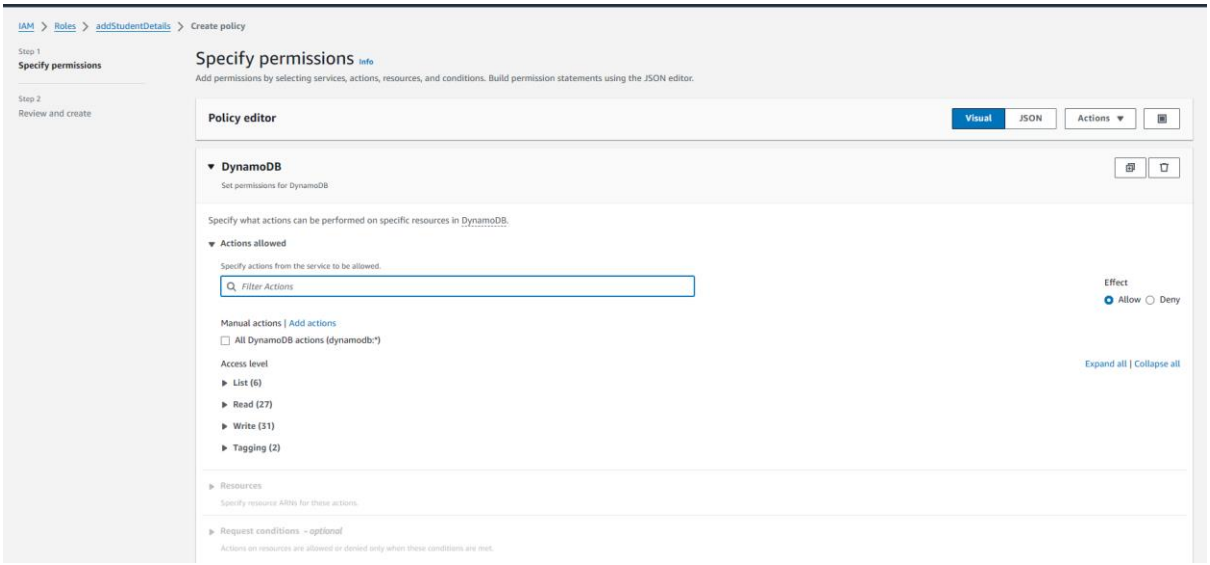
Scroll down and click on Add permissions.

click on Create Inline policy.

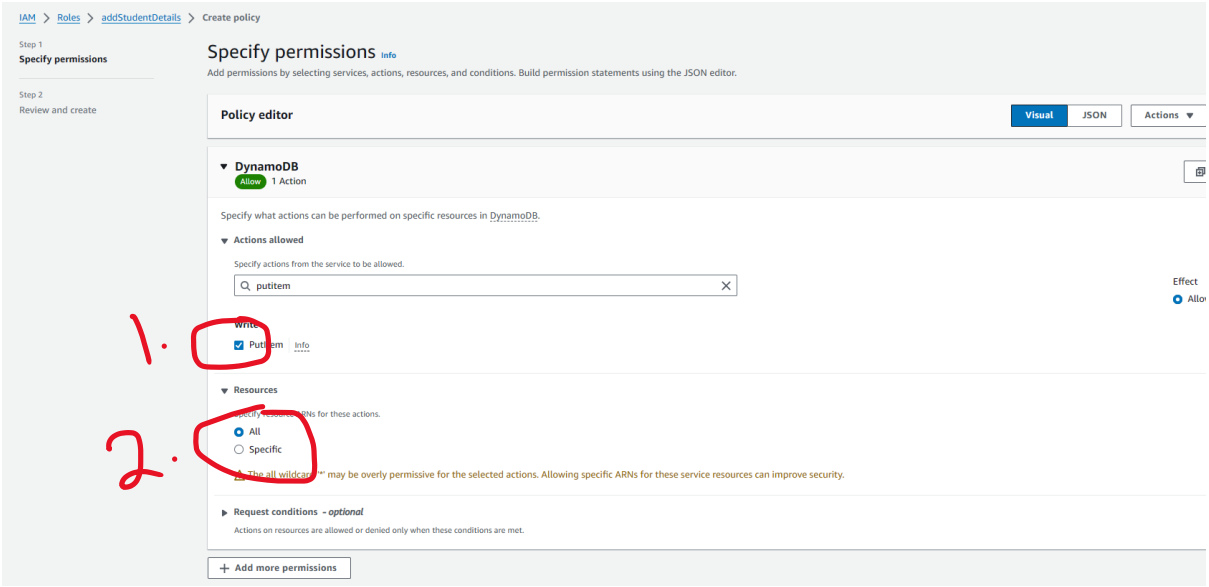




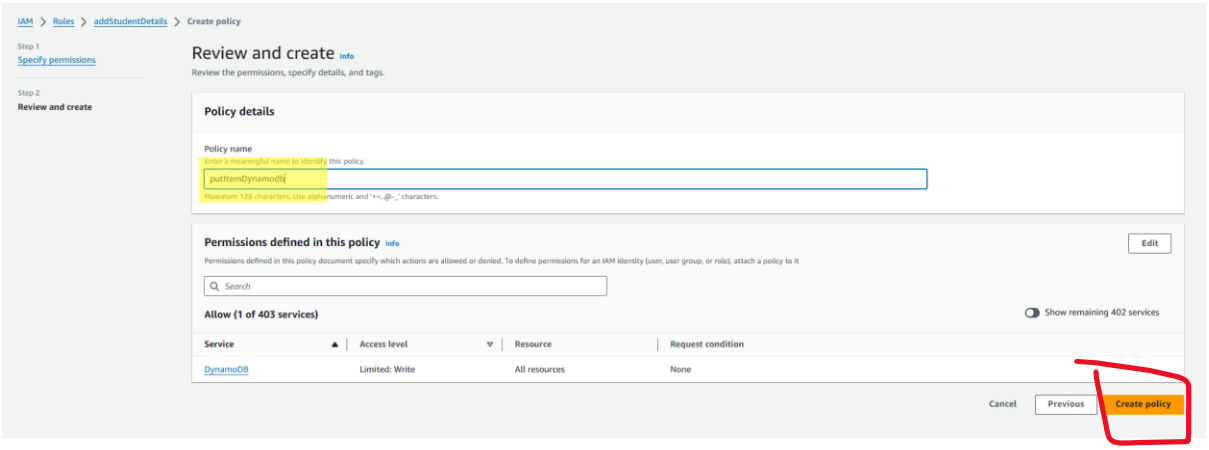




Search for putItem

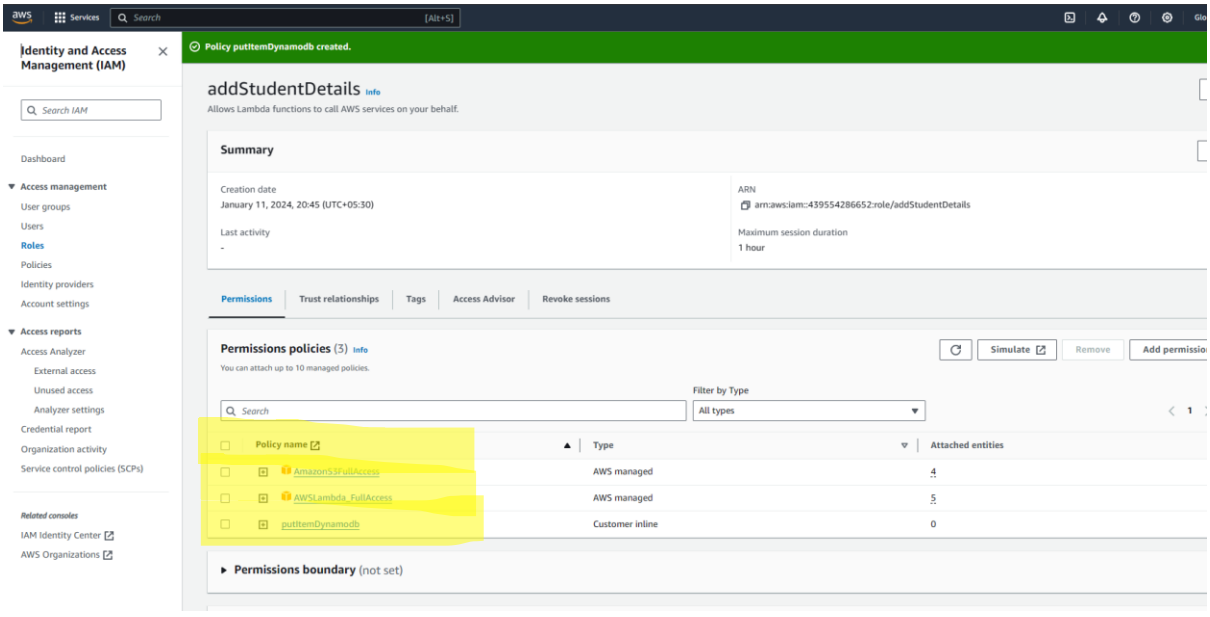


Scroll down and click on next.

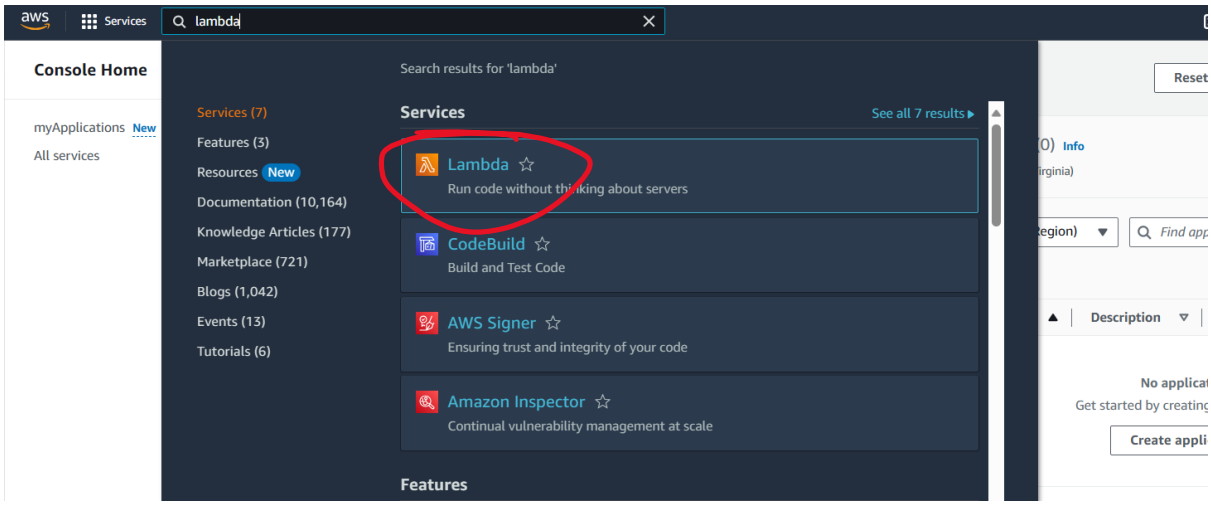


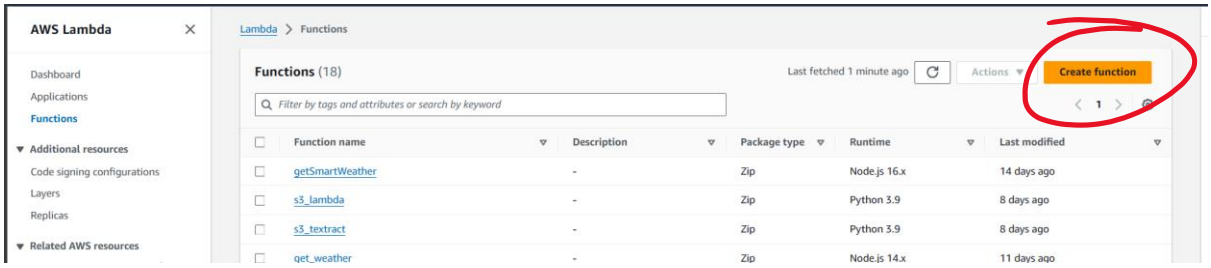
Policy name : putItemDynamodb

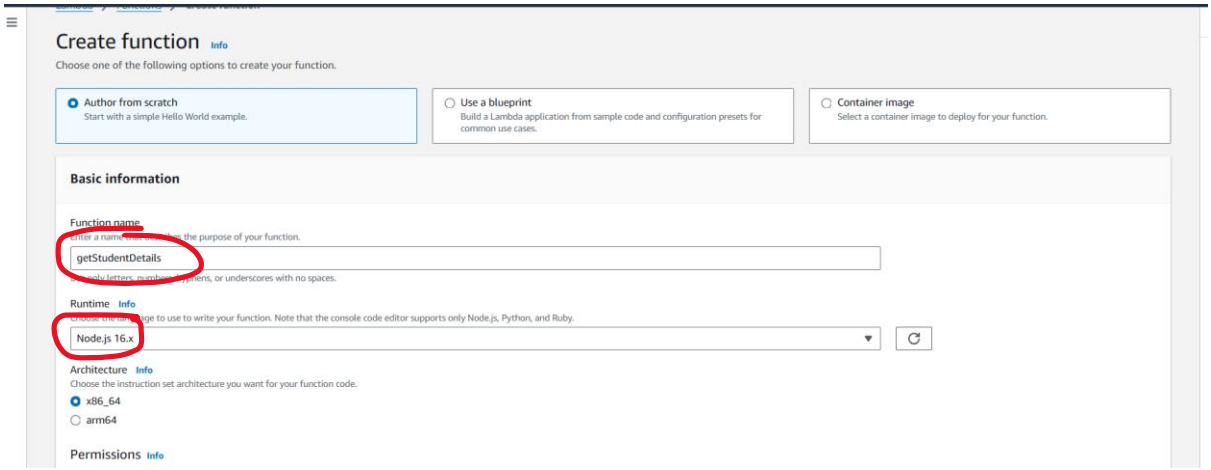
Click on Create policy.



Lambda



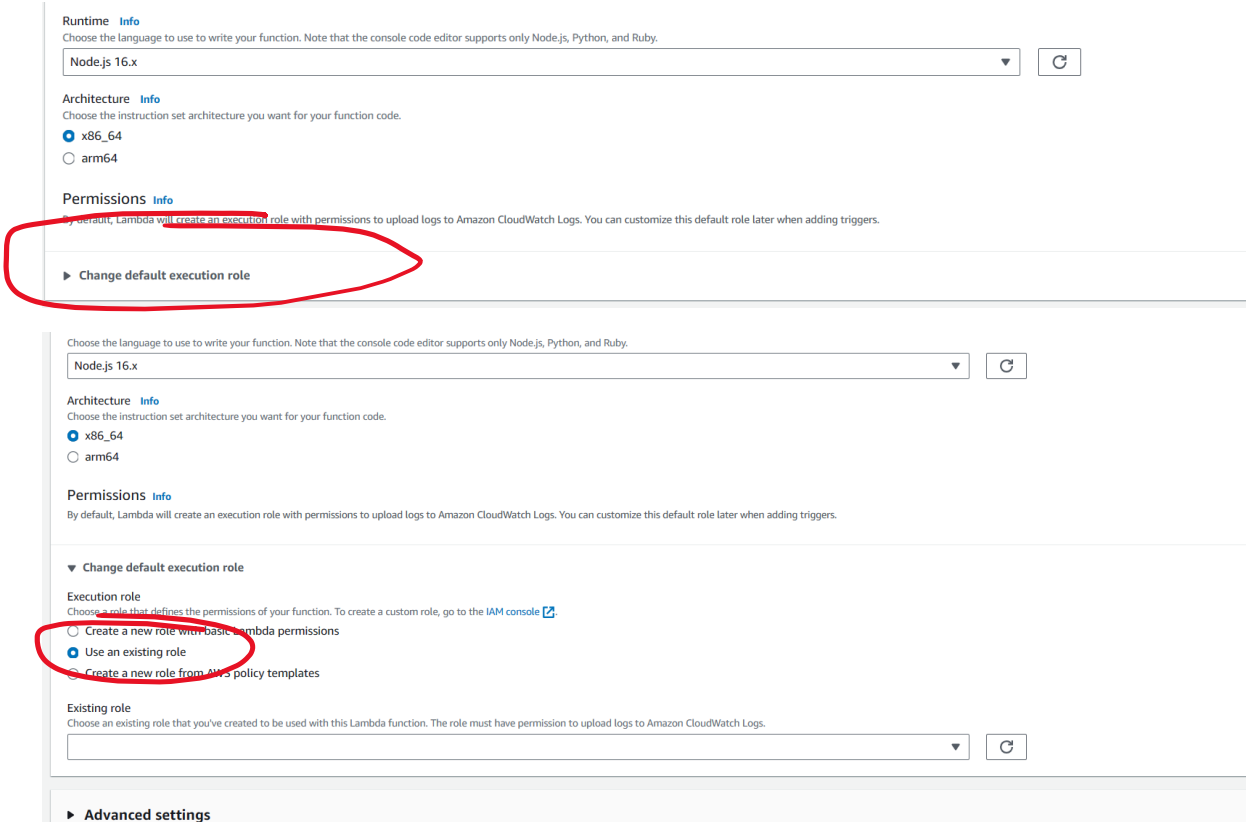




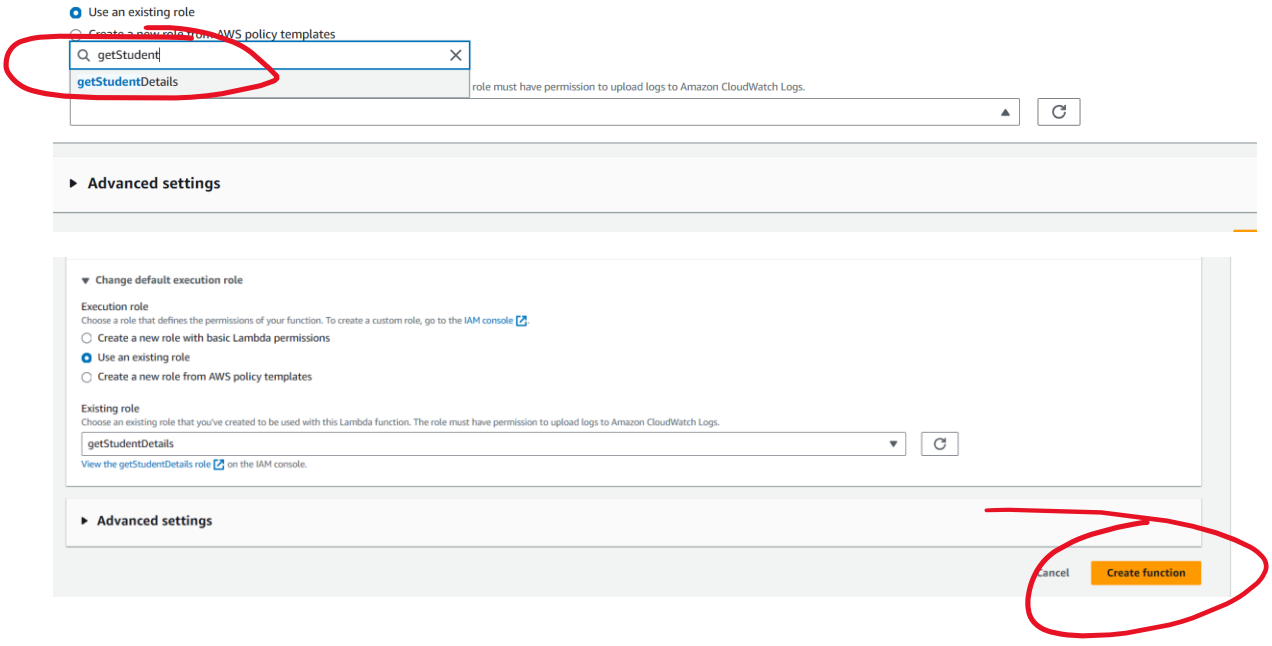
Function name : getStudentDetails

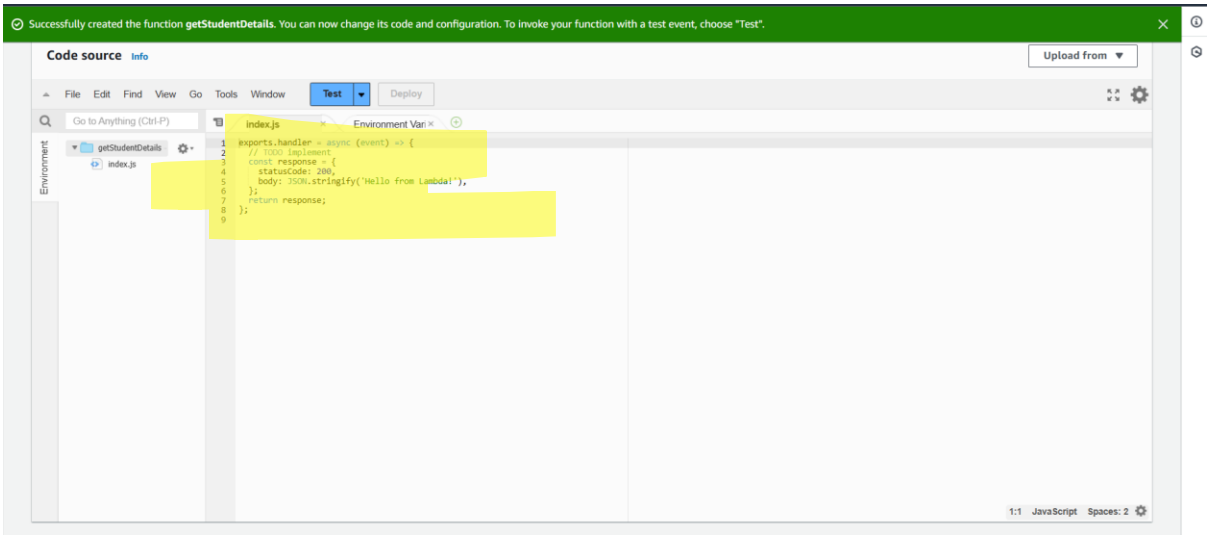
Runtime : Node.js 16.x

Scroll down



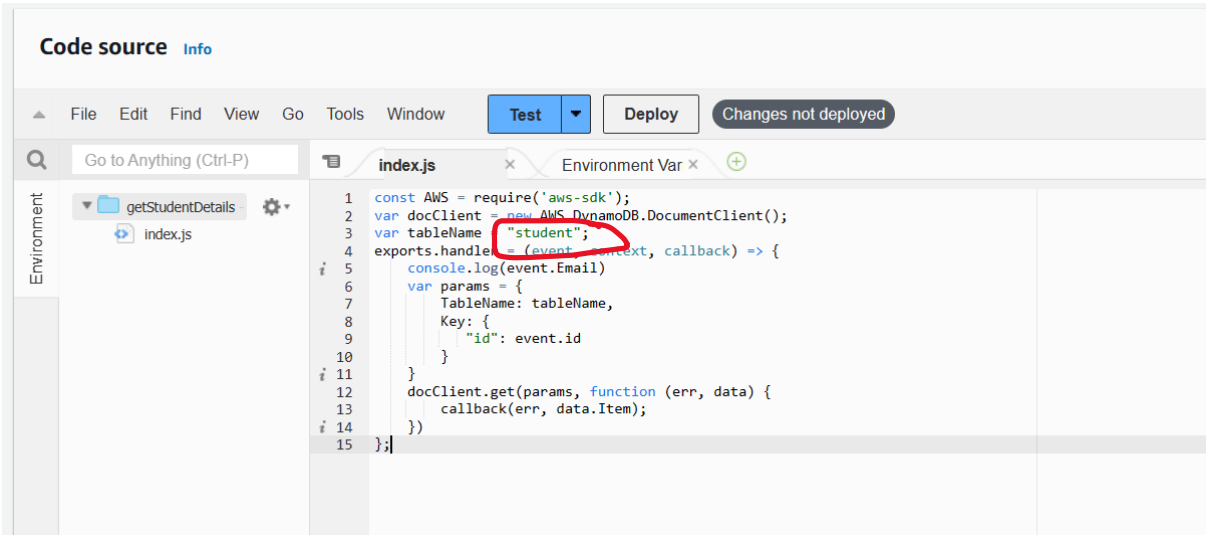
Type getStudentDetails to search existing roles.



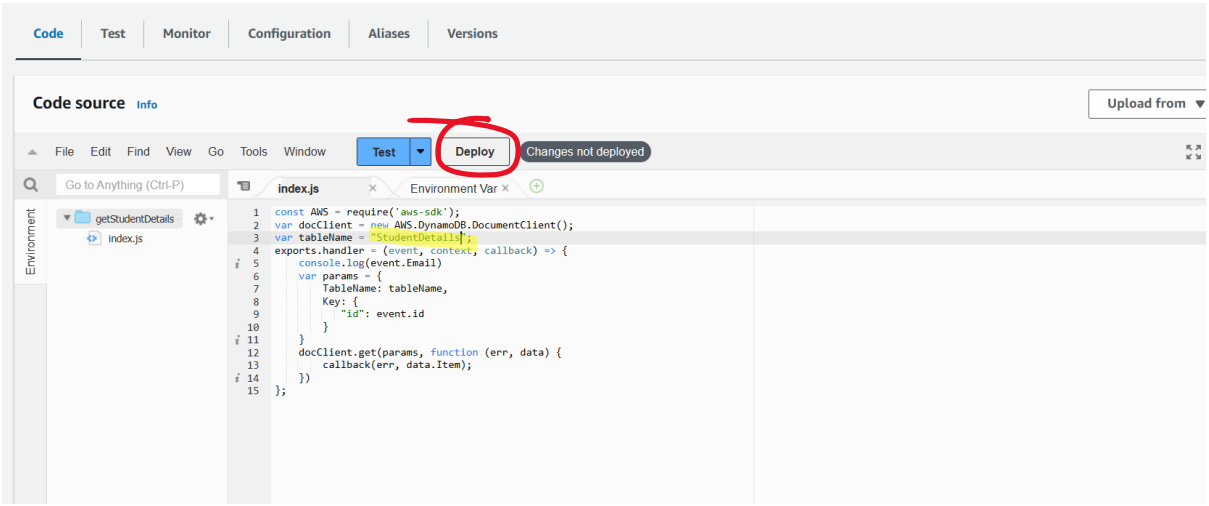


Replace the code with the below one :

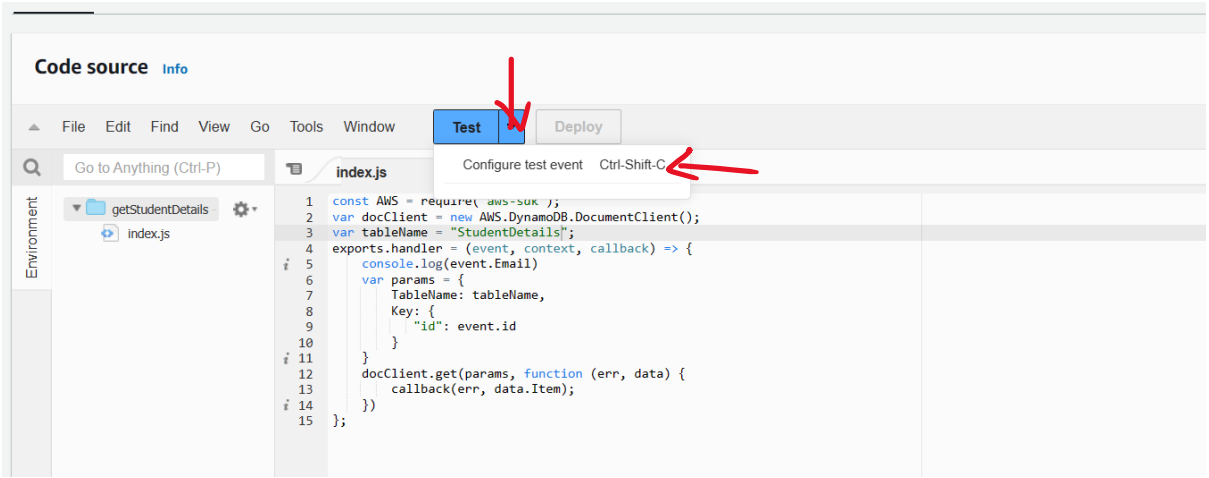
```
const AWS = require('aws-sdk');
var docClient = new AWS.DynamoDB.DocumentClient();
var tableName = "student";
exports.handler = (event, context, callback) => {
   console.log(event.Email)
   var params = {
      TableName: tableName,
      Key: {
        "id": event.id
      }
   }
   docClient.get(params, function (err, data) {
      callback(err, data.Item);
   })
};
```

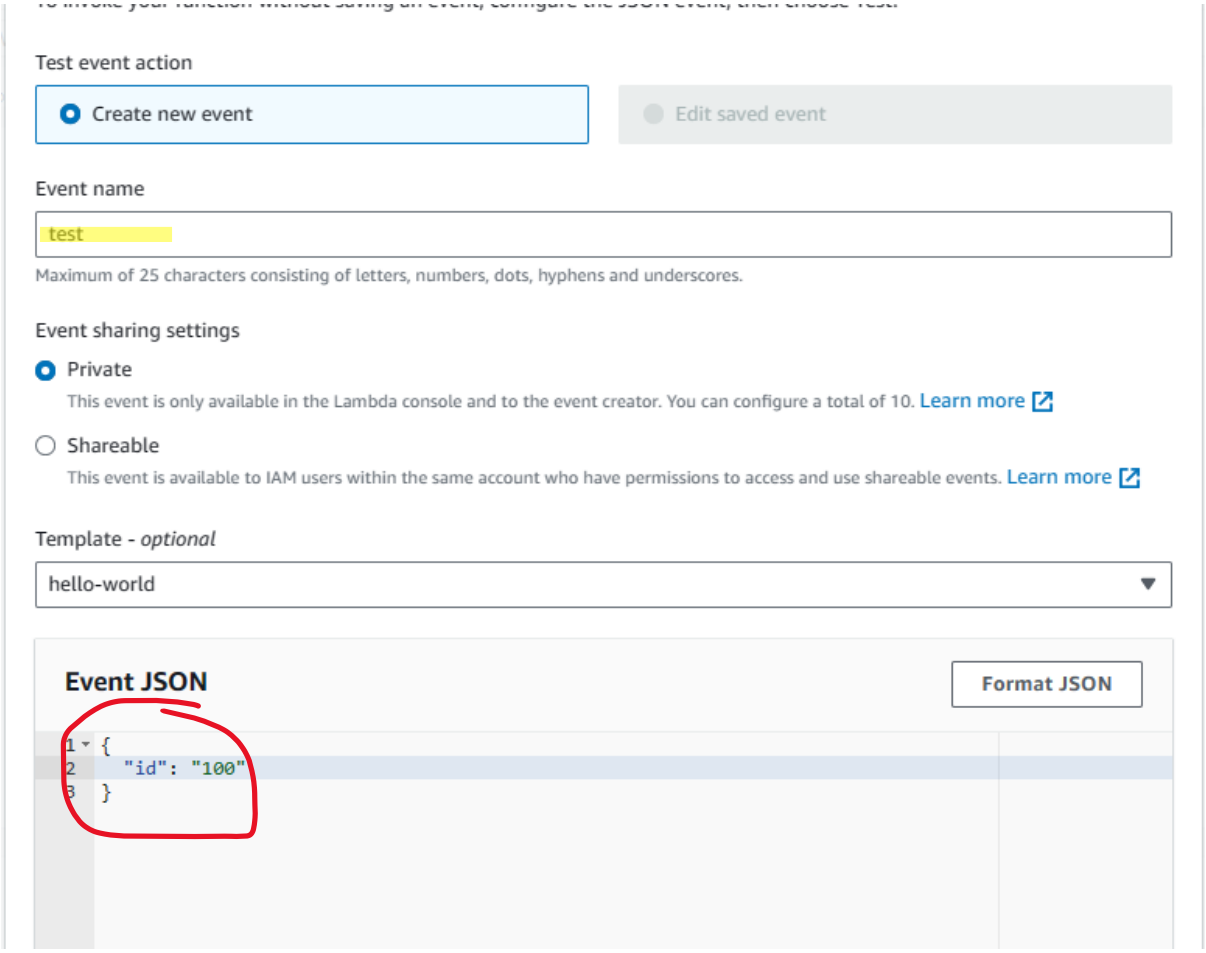


Replace the tableName with StudentDetails



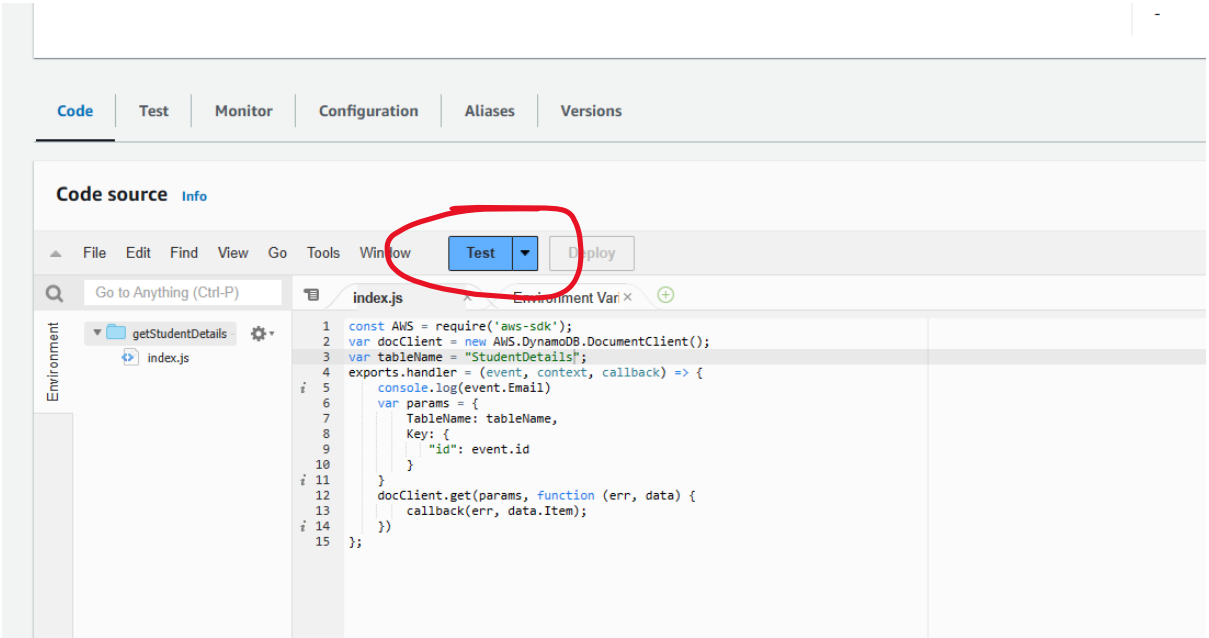
click on deploy



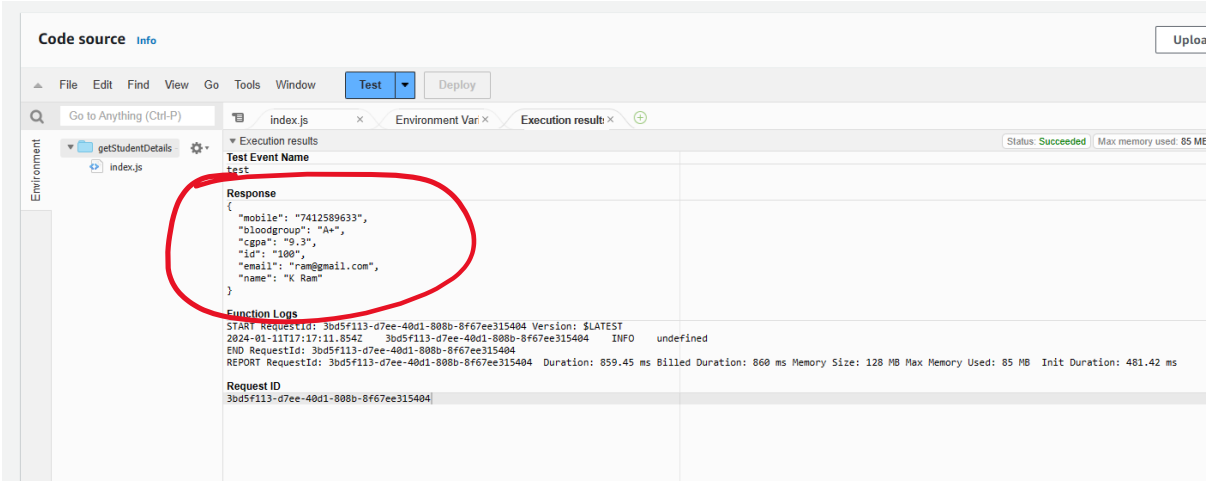


{
"id": "100"
}

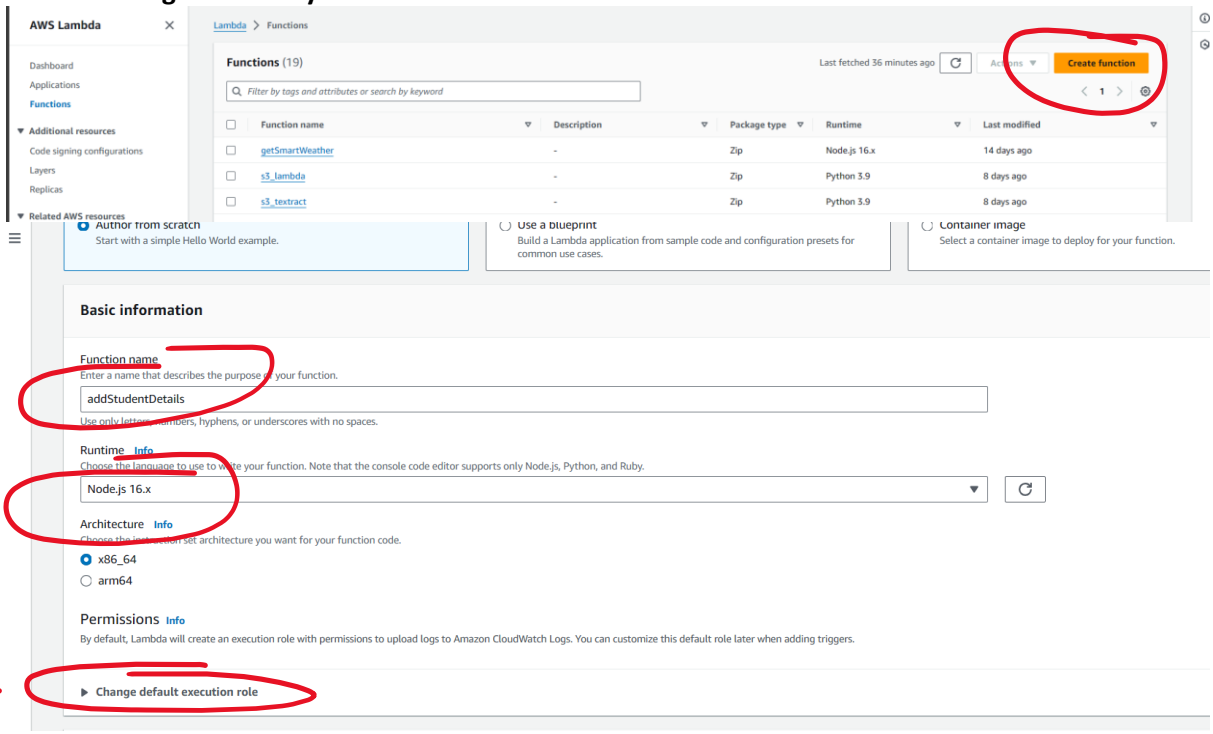
Scroll down and Click on save.



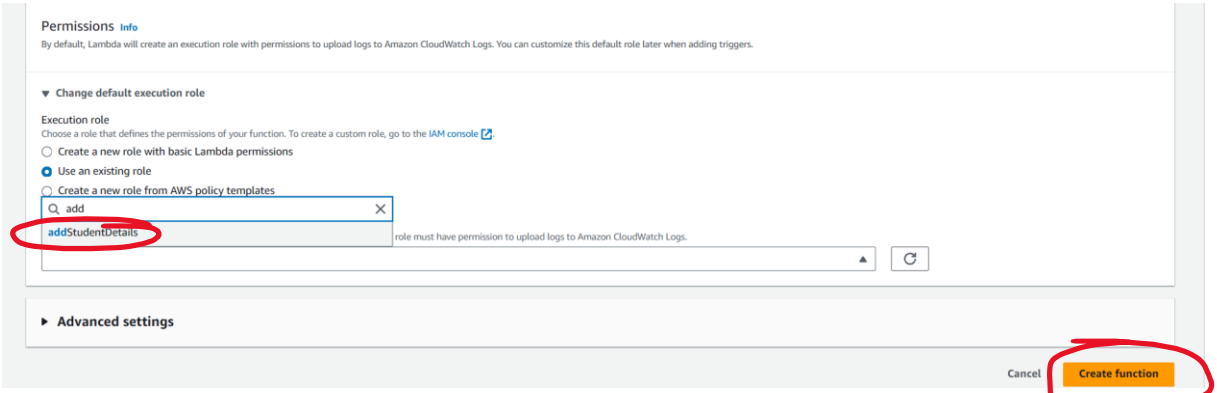
Click on test.



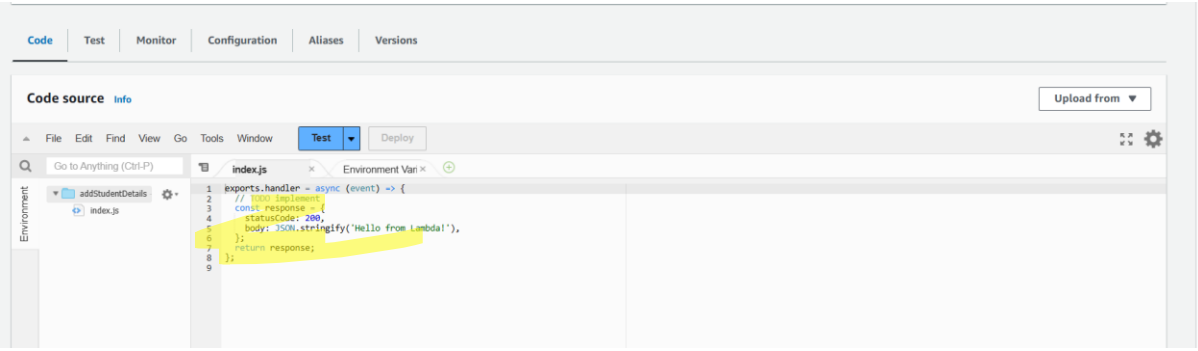
If you didn't get this response, go back check everything once again.
Now we are able to retrieve the items from dynamodb table.
Next if adding items to dynamodb.



Search for addStudentDetails role.

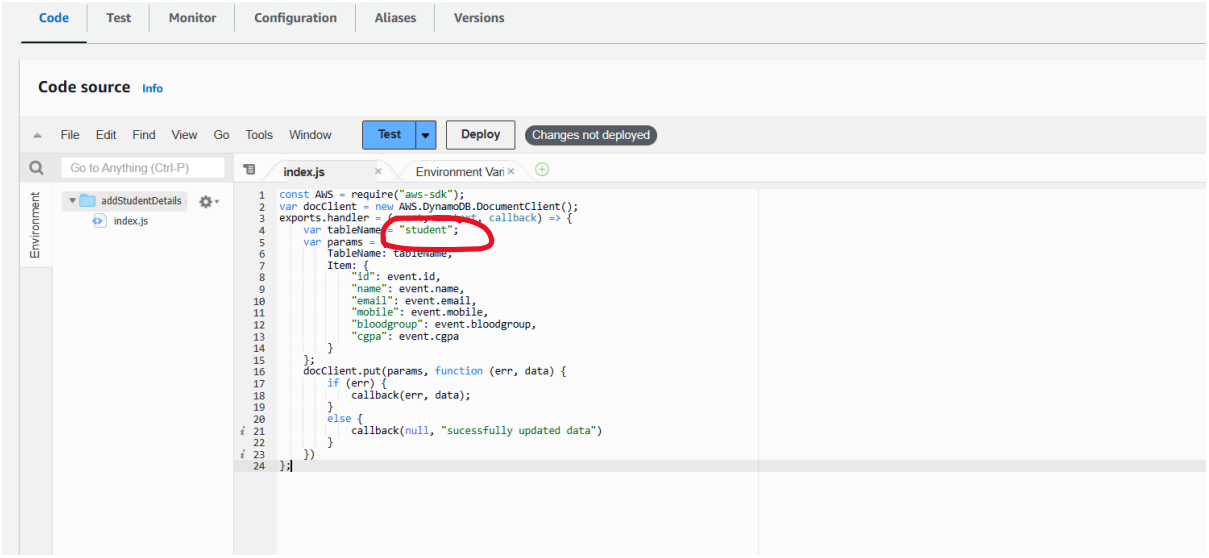


Select it and click on create function.

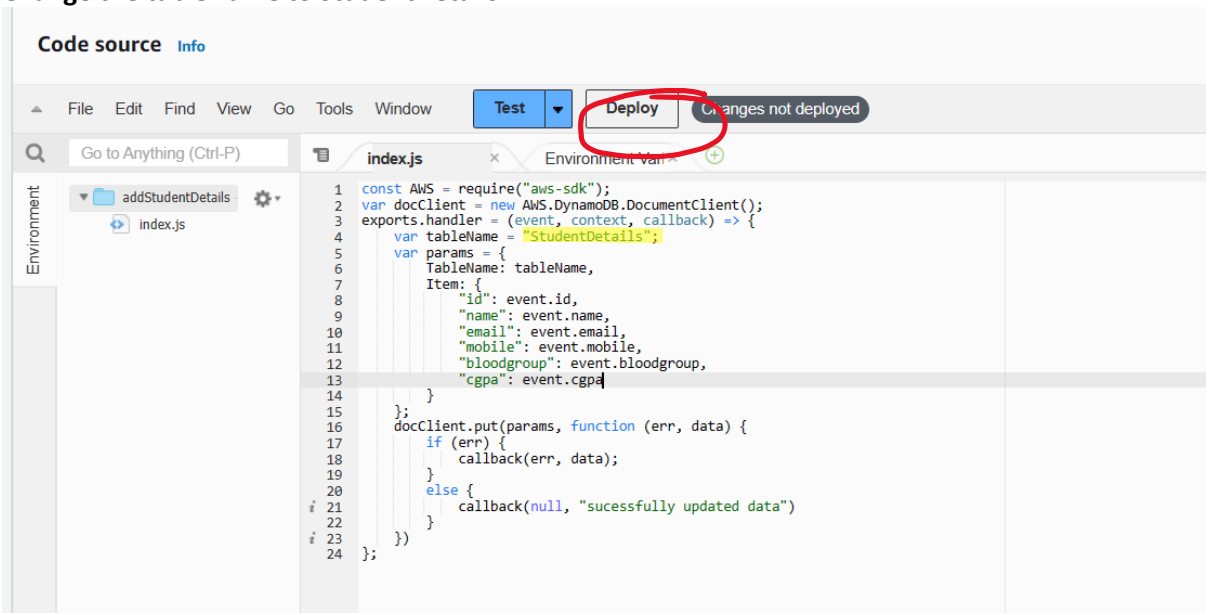


Replace the above code with the below one :

```
const AWS = require("aws-sdk");
var docClient = new AWS.DynamoDB.DocumentClient();
exports.handler = (event, context, callback) => {
   var tableName = "student";
   var params = {
      TableName: tableName,
      Item: {
         "id": event.id,
         "name": event.name,
         "email": event.email,
         "mobile": event.mobile,
         "bloodgroup": event.bloodgroup,
         "cgpa": event.cgpa
      }
   };
   docClient.put(params, function (err, data) {
      if (err) {
         callback(err, data);
      }
      else {
         callback(null, "sucessfully updated data")
      }
   })
};
```

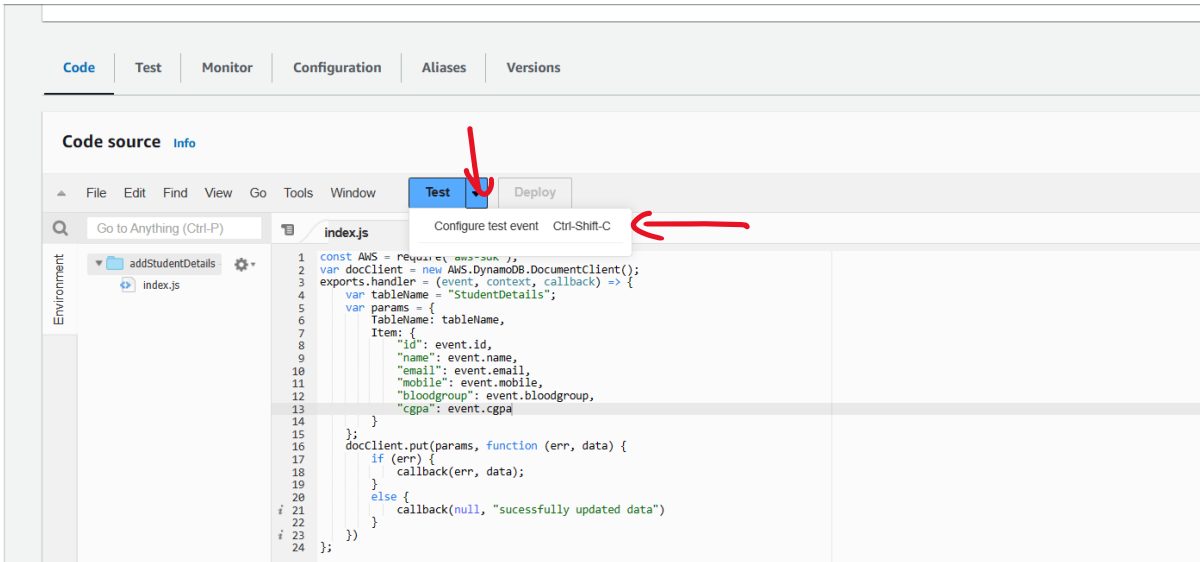


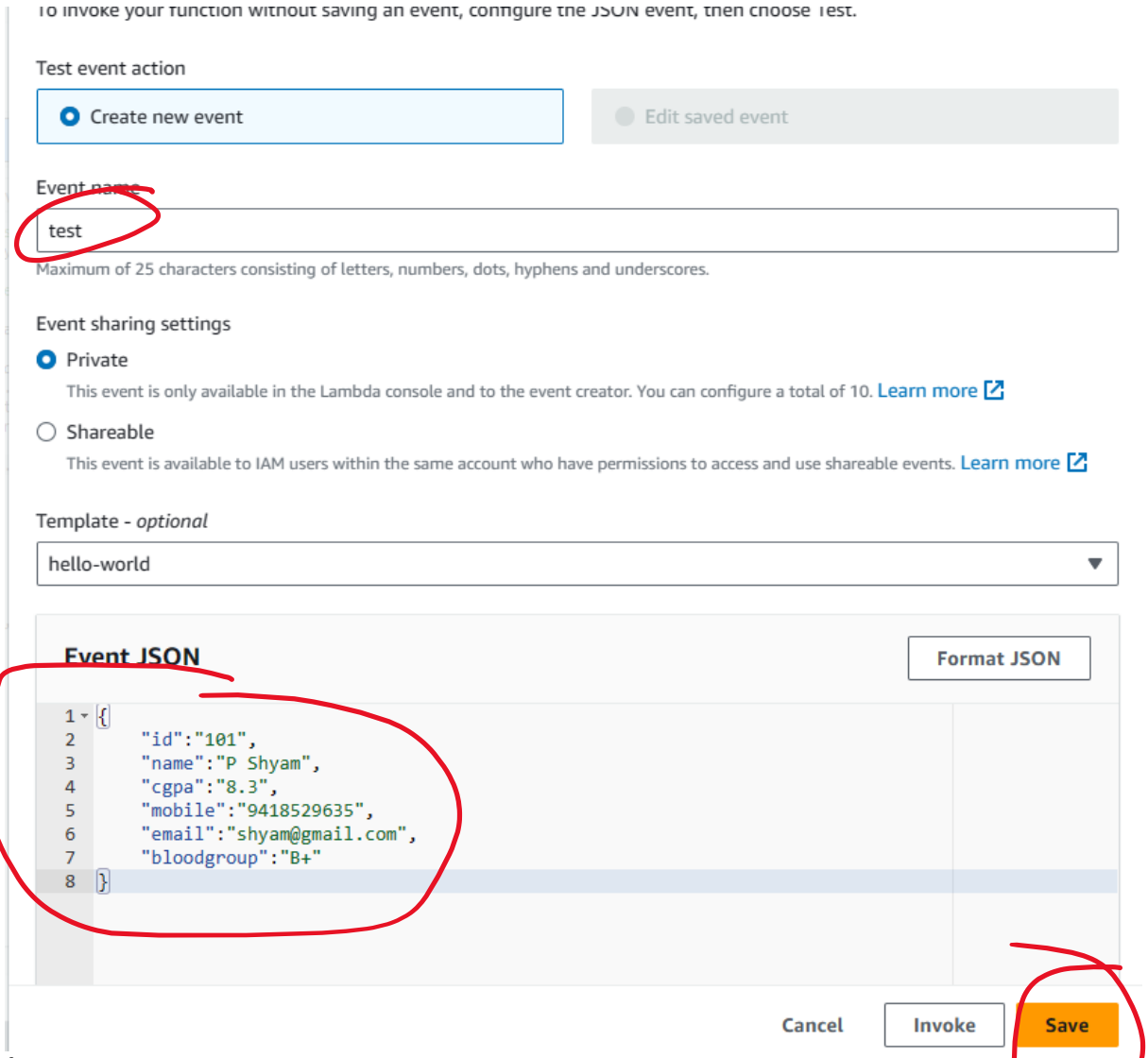
Change the tableName to StudentDetails.



Click on deploy.

Click on test



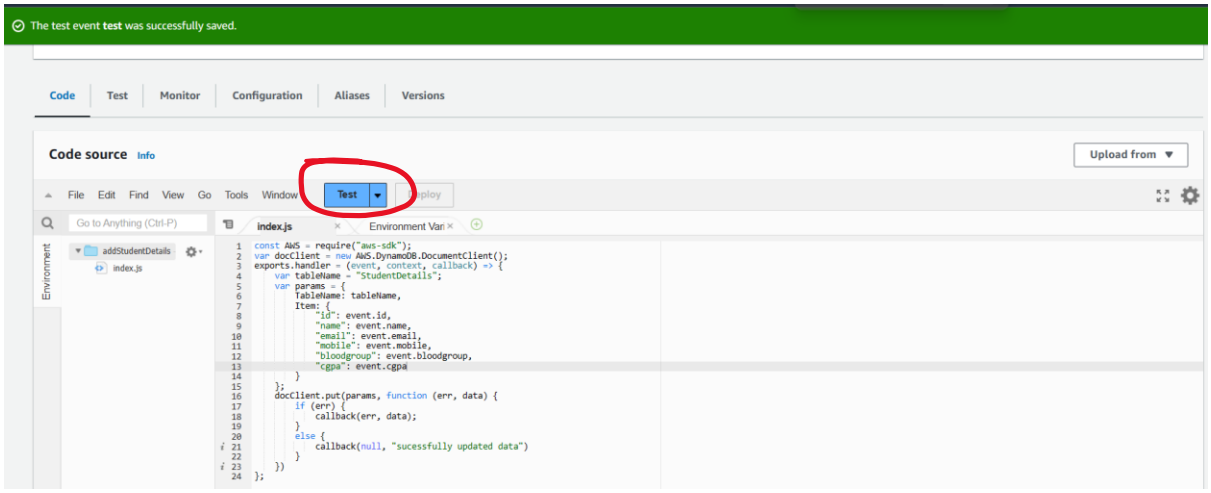


{

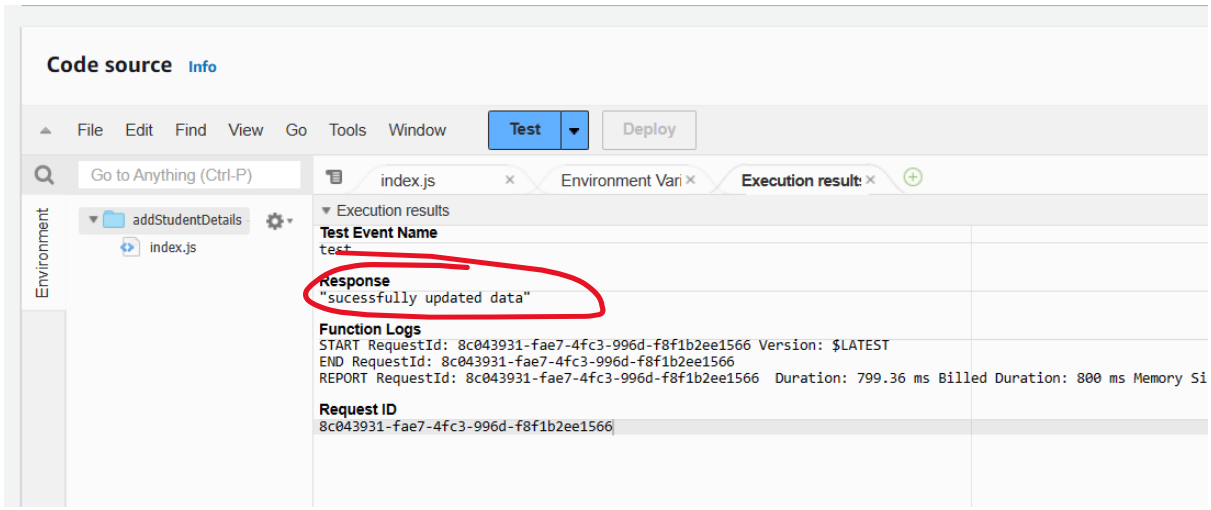
"id":"101",
"name":"P Shyam",
"cgpa":"8.3",
"mobile":"9418529635",
"email":"shyam@gmail.com",
"bloodgroup":"B+"

}

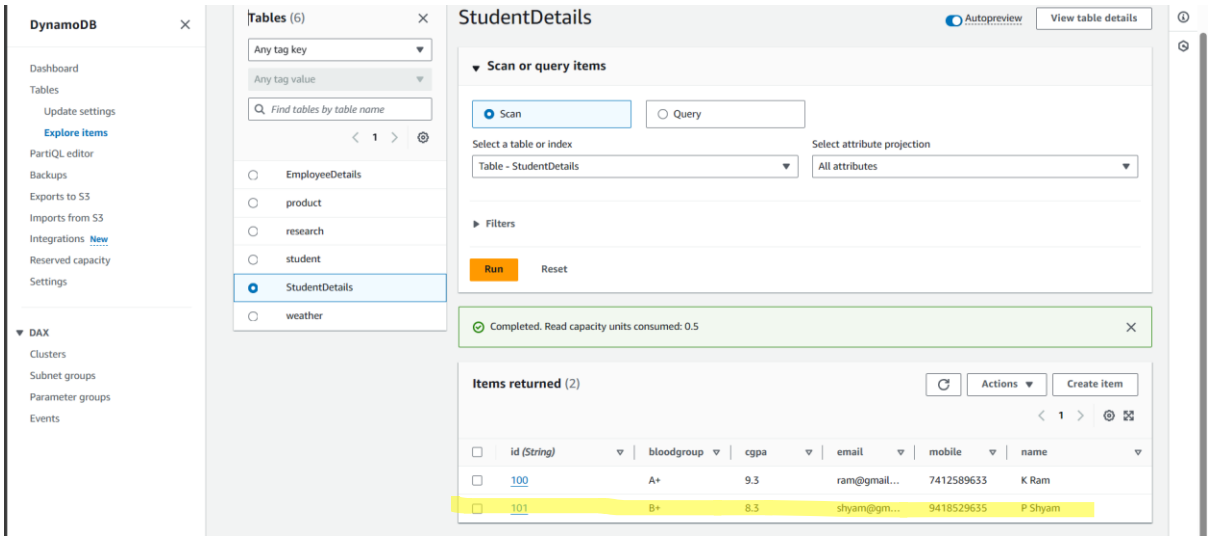
Click on save.



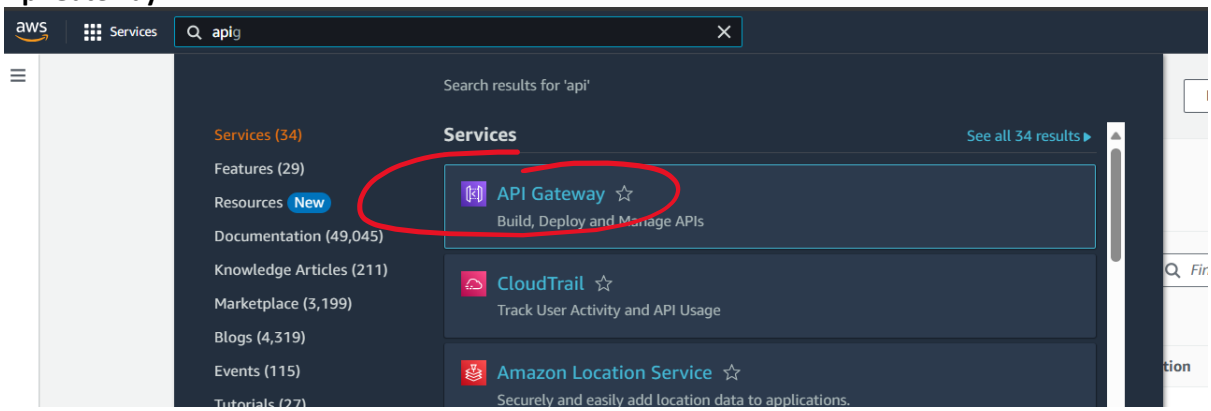
Click on test.



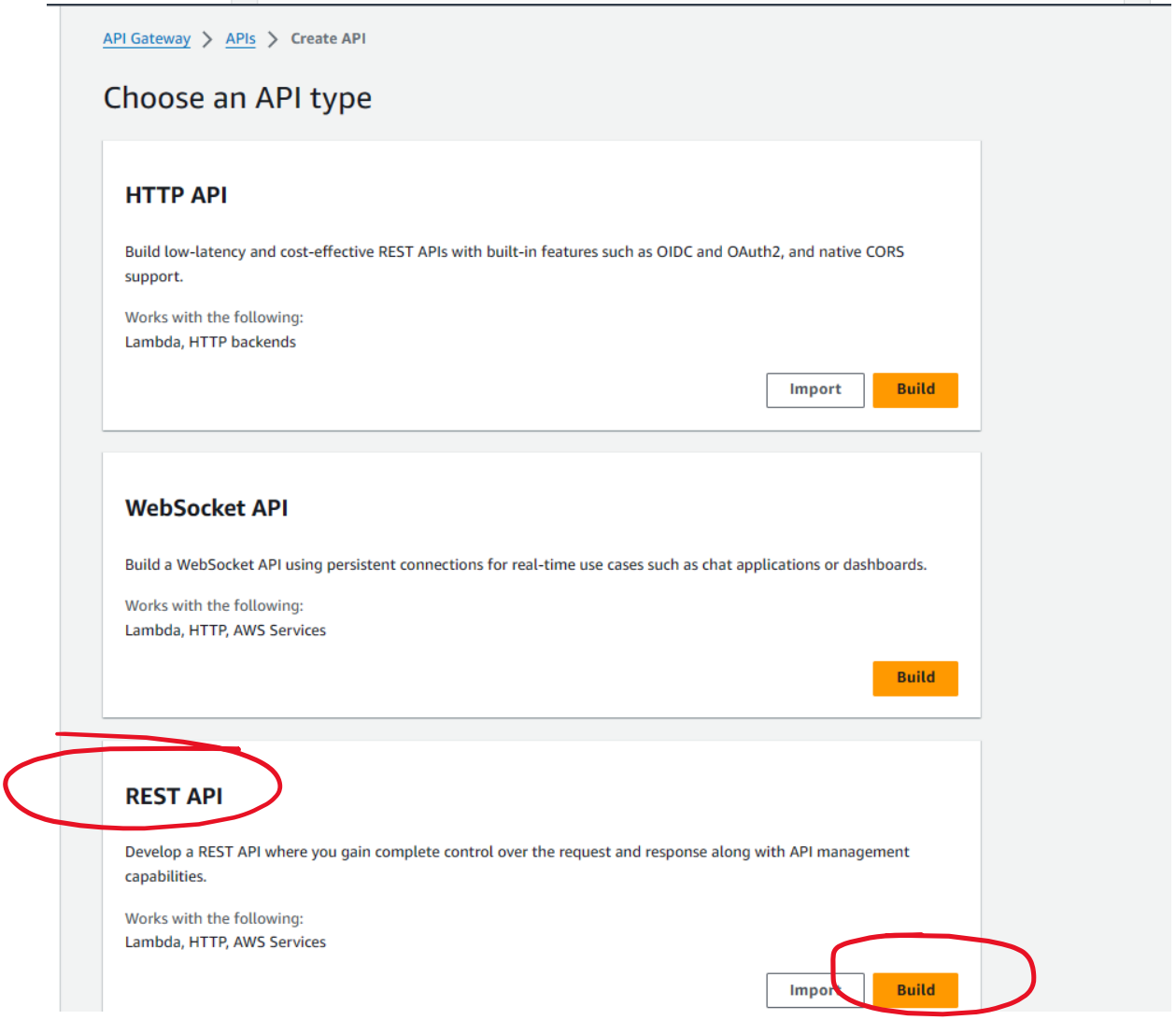
Lets check in dynamodb table StudentDetails whether the item is added or not.

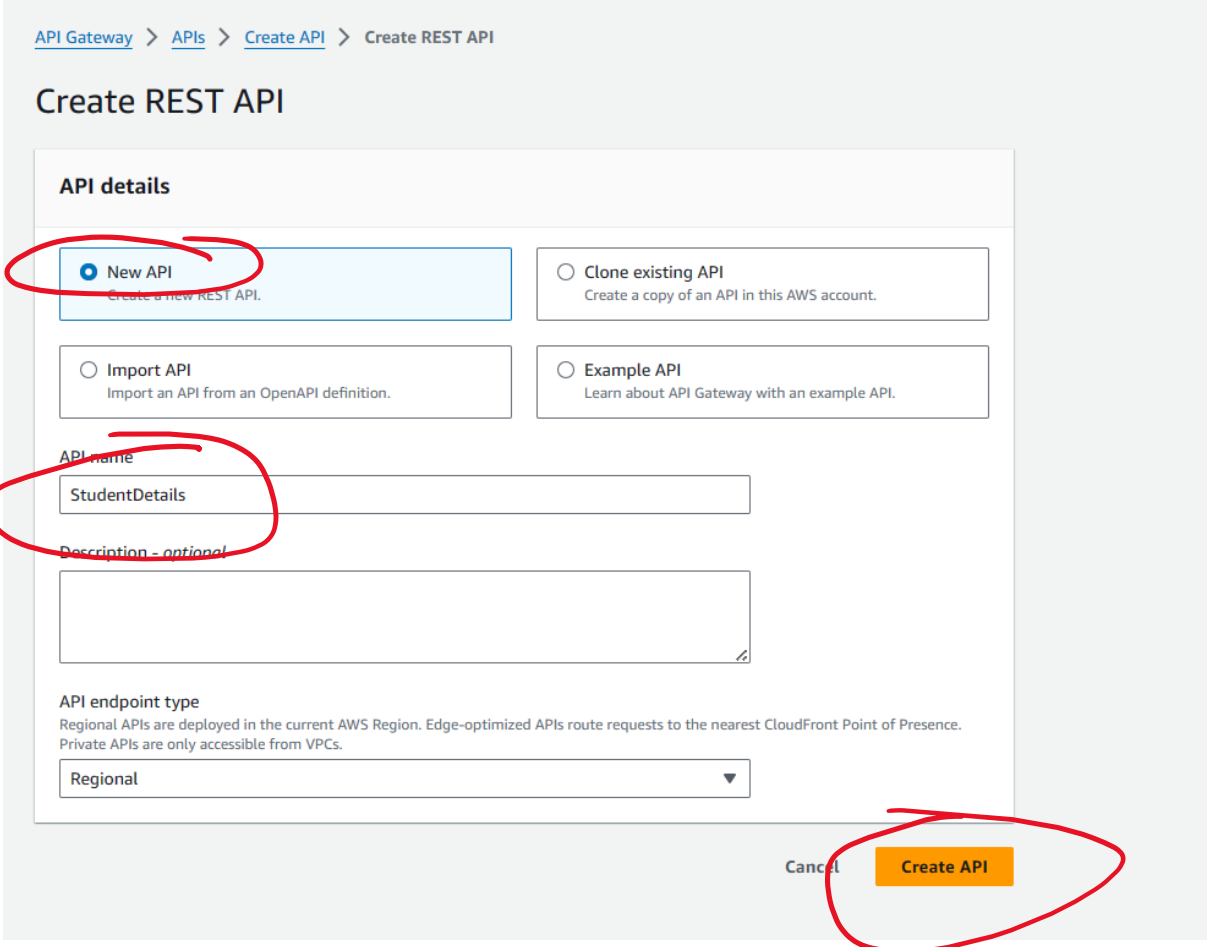


Api Gateway

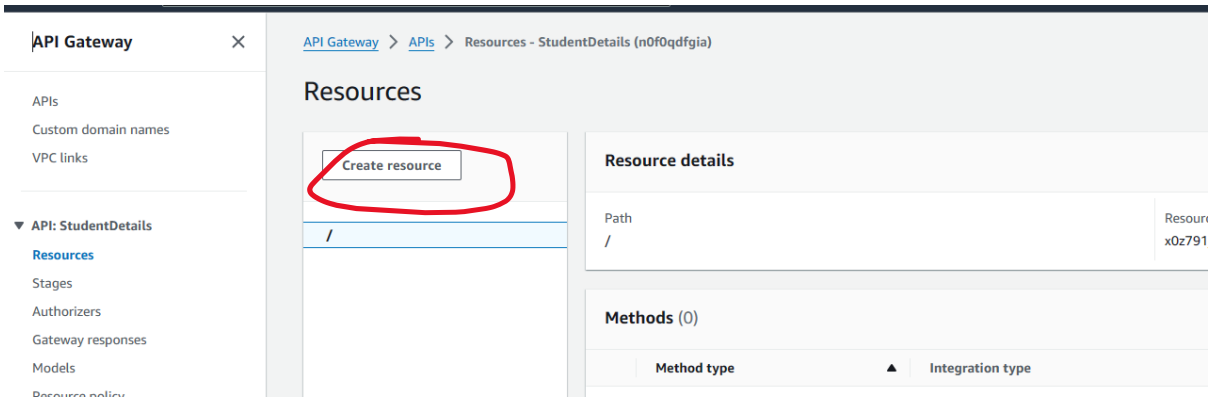

aws
Services
apig
Search results for 'api'
Services (34)
Features (29)
Resources New
Documentation (49,045)
Knowledge Articles (211)
Marketplace (3,199)
Blogs (4,319)
Events (115)
Services
See all 34 results
API Gateway
Build, Deploy and Manage APIs
CloudTrail
Track User Activity and API Usage
Amazon Location Service
Securely and easily add location data to applications.


API Gateway
APIs
Custom domain names
VPC links
API Gateway > APIs
APIs (6/6)
Find APIs
Delete
Create API

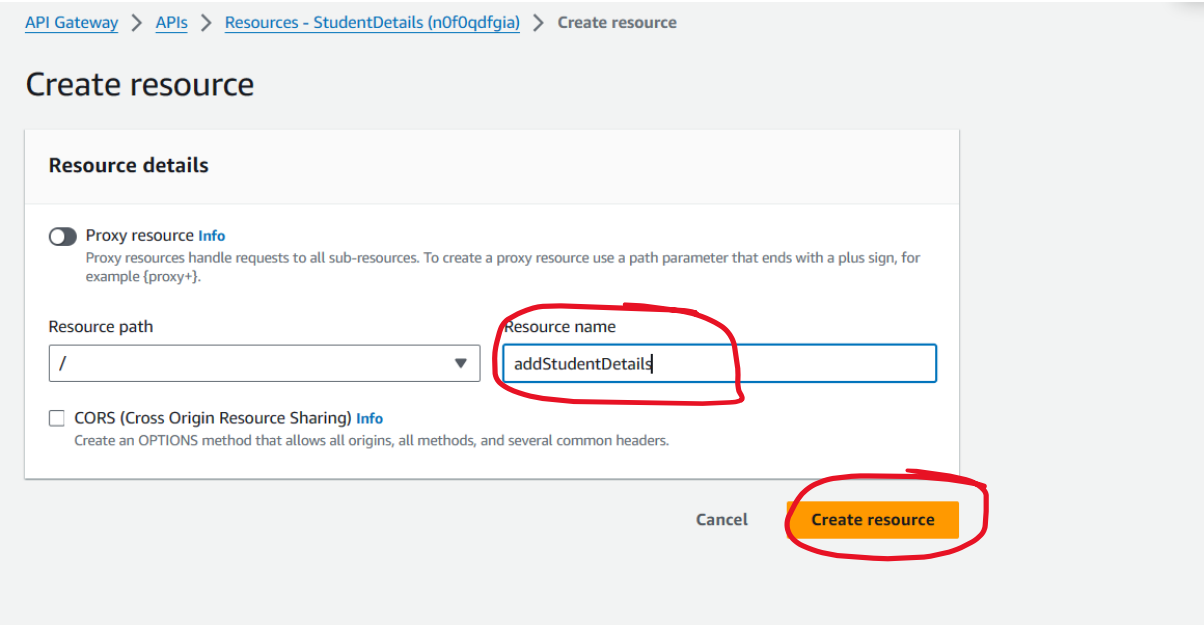

API Gateway > APIs > Create API
Choose an API type
HTTP API
Build low-latency and cost-effective REST APIs with built-in features such as OIDC and OAuth2, and native CORS support.
Works with the following:
Lambda, HTTP backends
Import
Build
WebSocket API
Build a WebSocket API using persistent connections for real-time use cases such as chat applications or dashboards.
Works with the following:
Lambda, HTTP, AWS Services
Build
REST API
Develop a REST API where you gain complete control over the request and response along with API management capabilities.
Works with the following:
Lambda, HTTP, AWS Services
Build



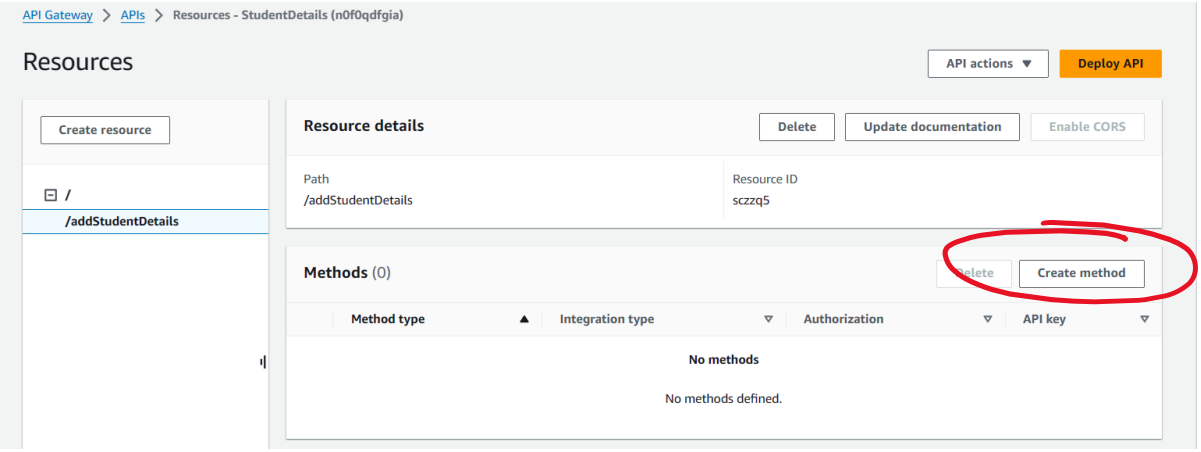
API name : **StudentDetails**
Click on create



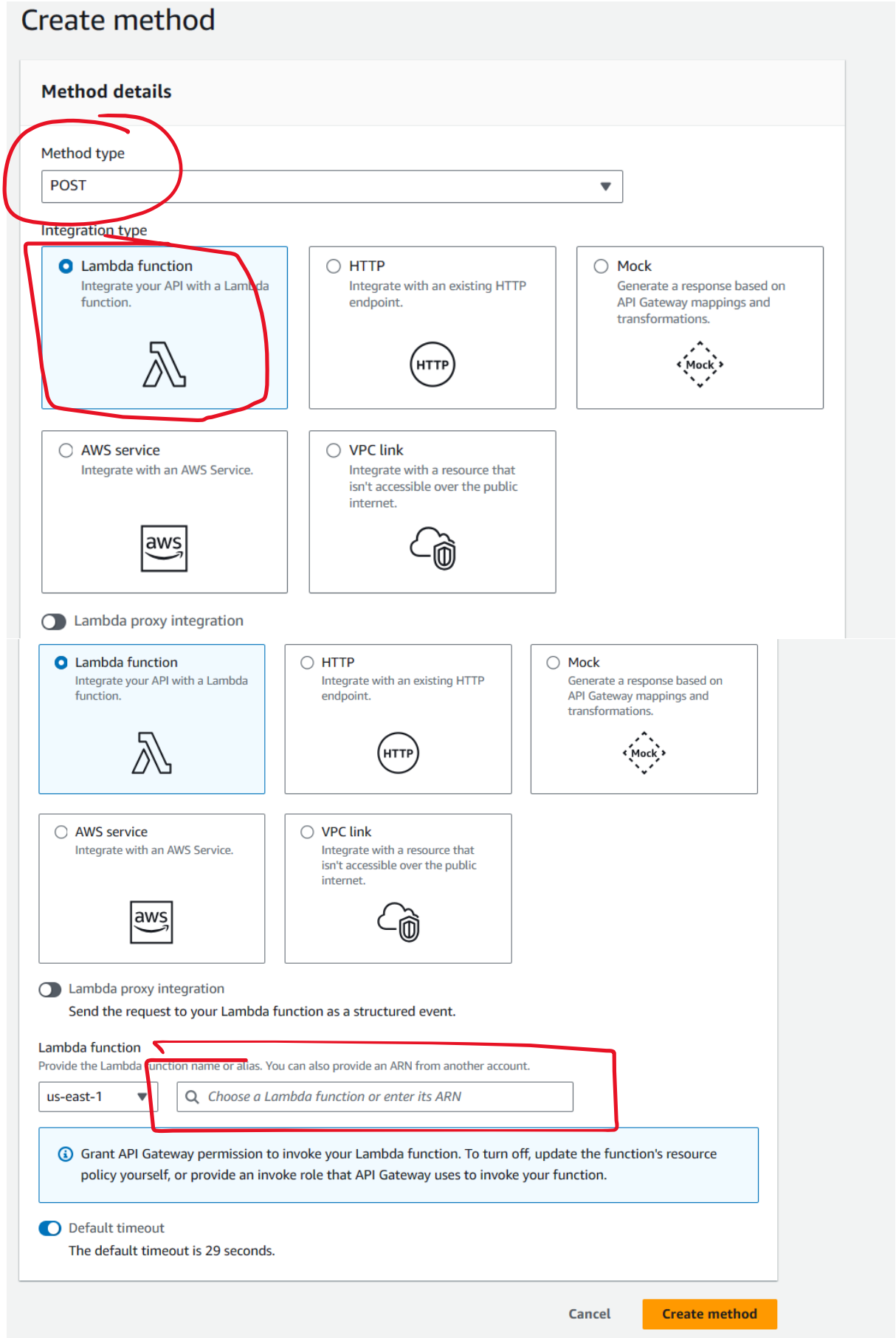
Click on create resource



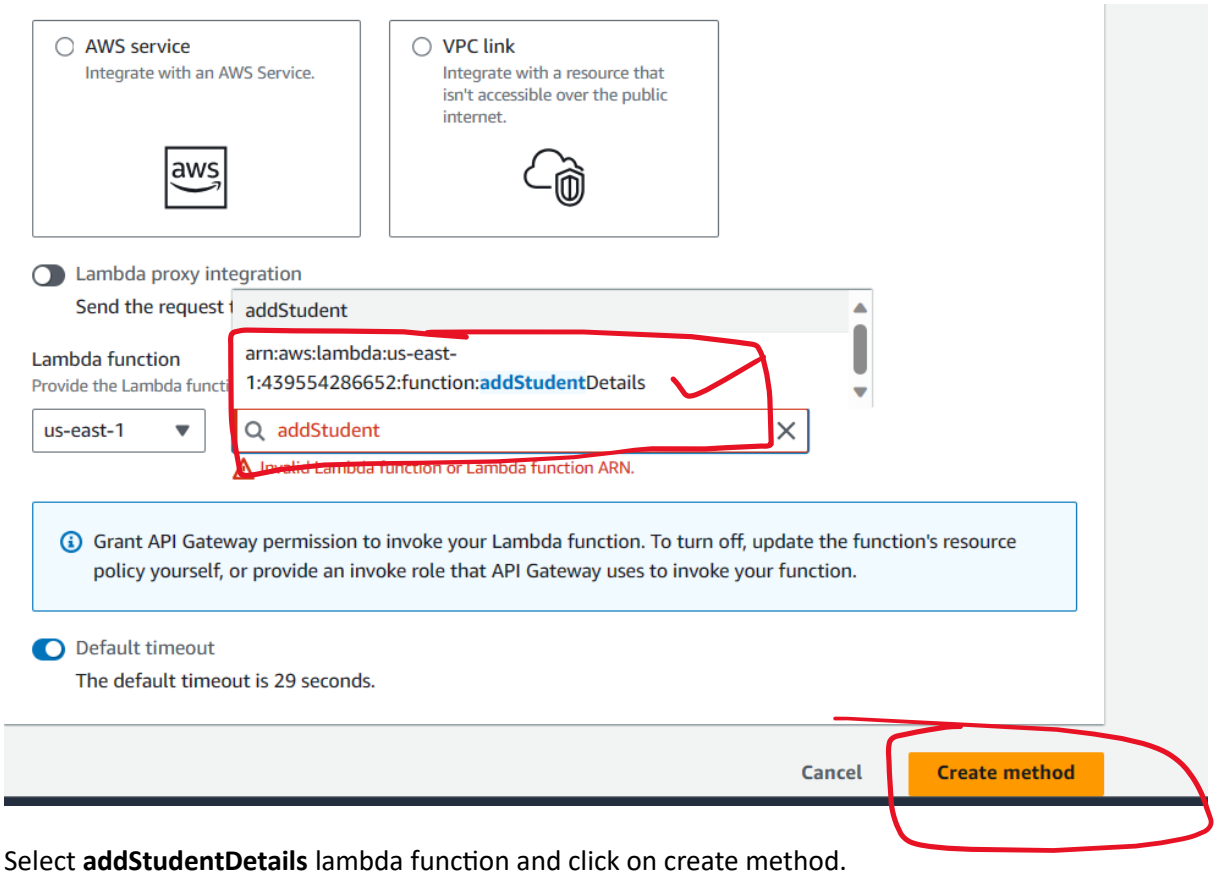
Resource name : **addStudentDetails**



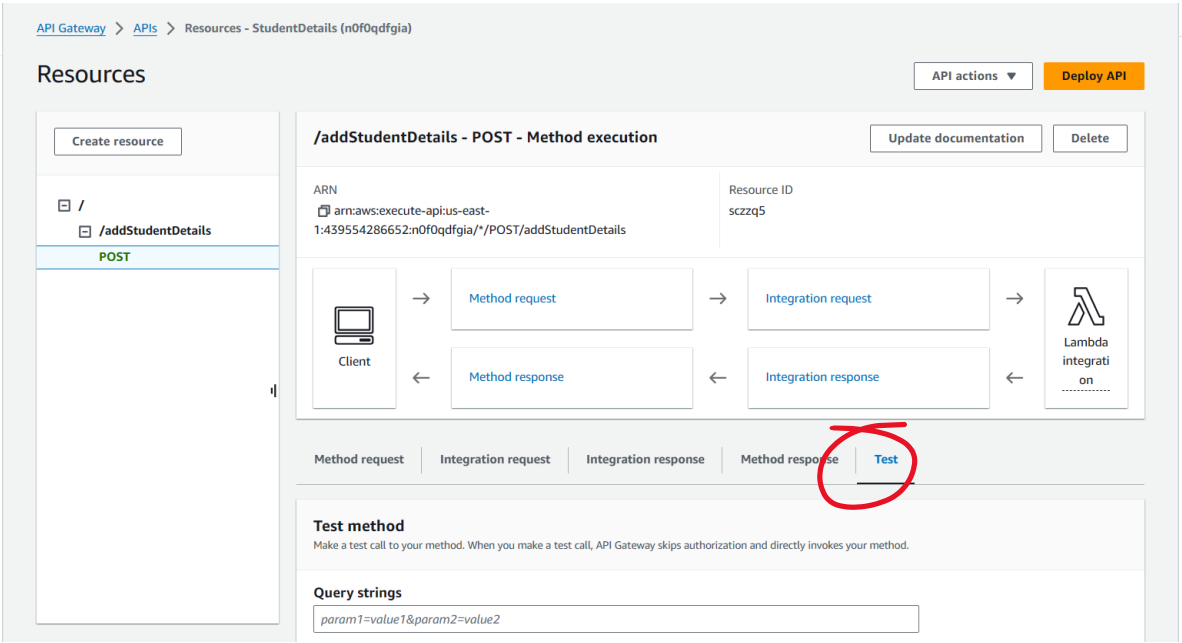
Click on create method .
Method type : POST
Integration type : Lambda



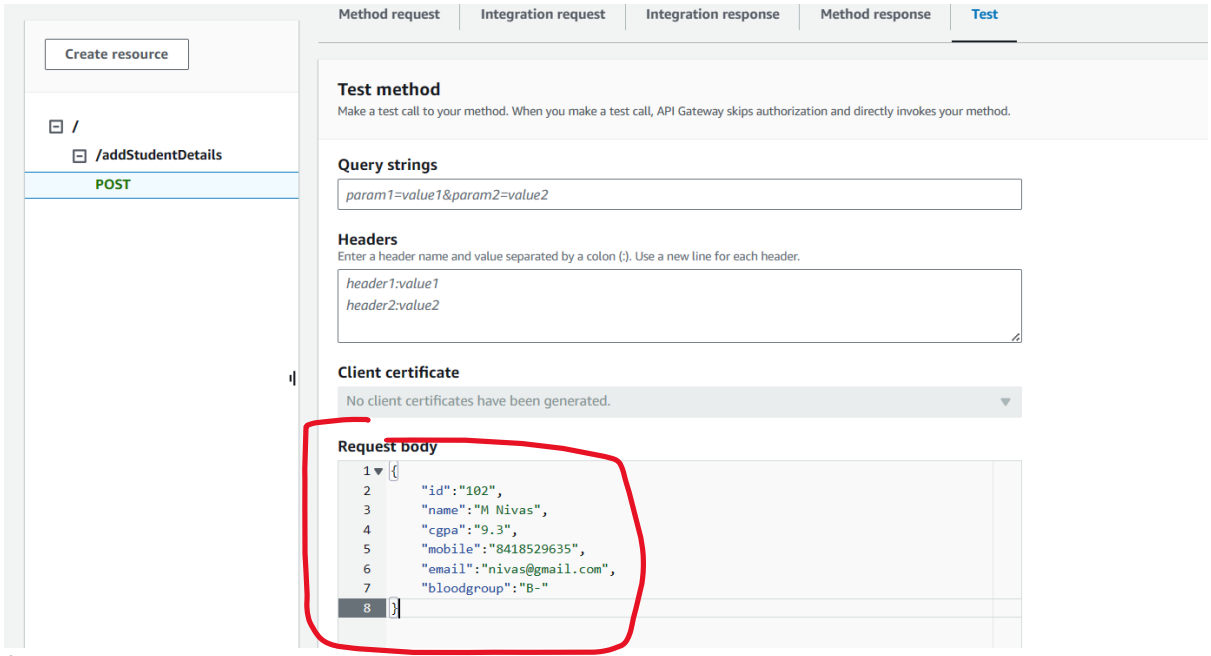
Search for **addStudentDetails** lambda function.



Select **addStudentDetails** lambda function and click on create method.

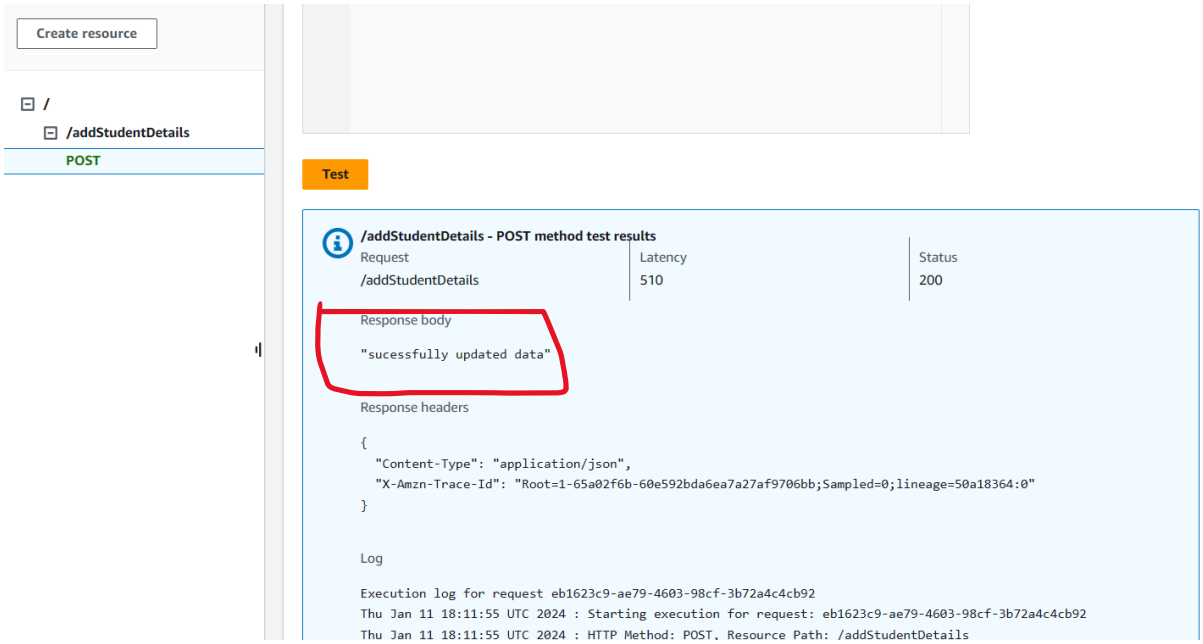


Click on **Test**.

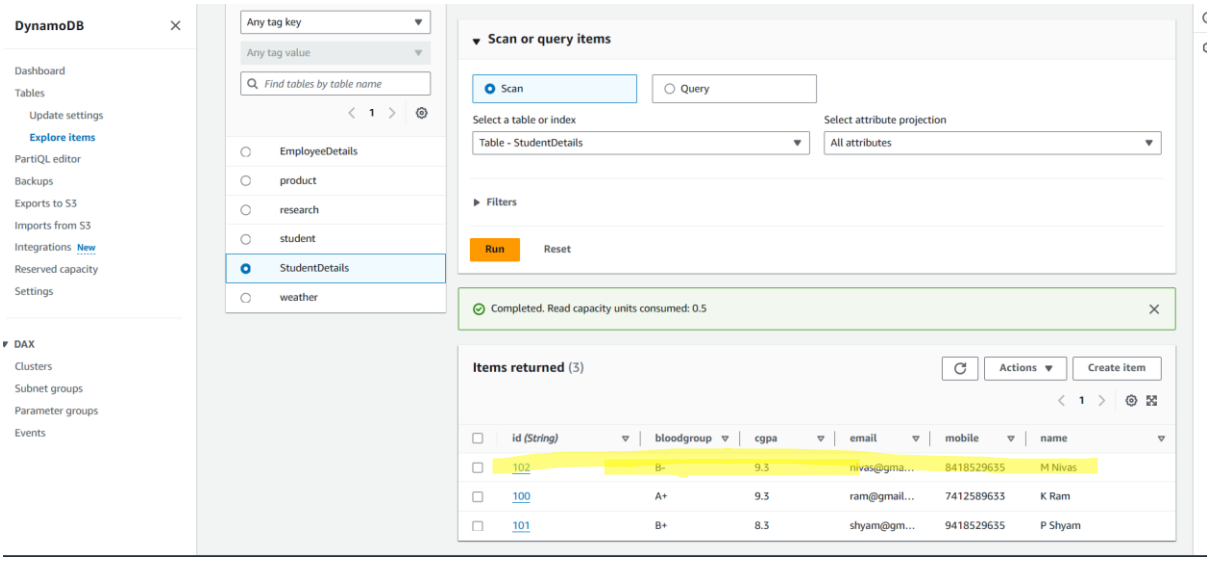



```
{
        "id":"102",
        "name":"V Nivas",
        "cgpa":"9.3",
        "mobile":"8418529635",
        "email":"nivas@gmail.com",
        "bloodgroup":"B-"
}
```

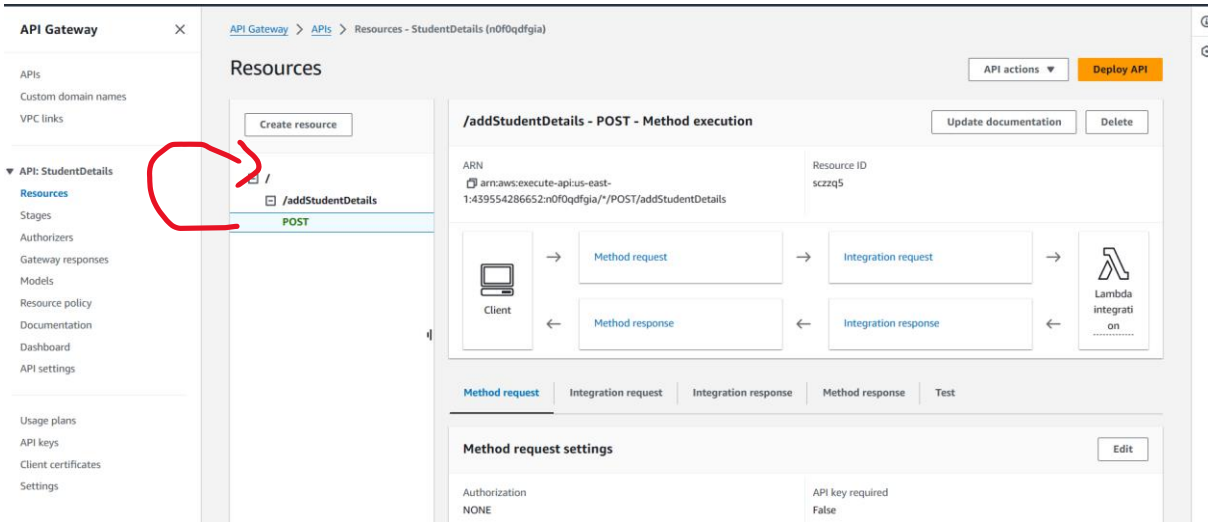

Scroll down and click on **Test** .



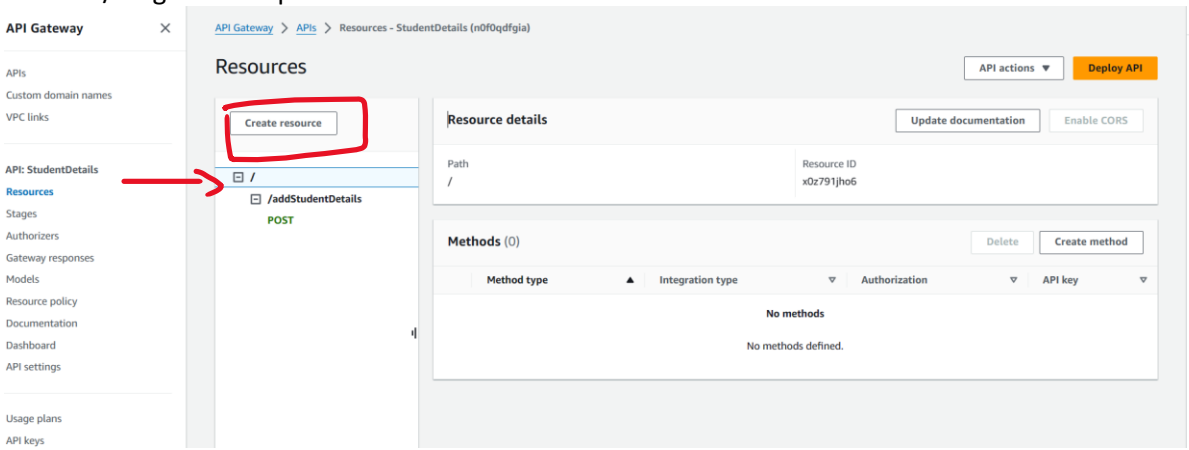
Lets check in dynamodb table **StudentDetails** .



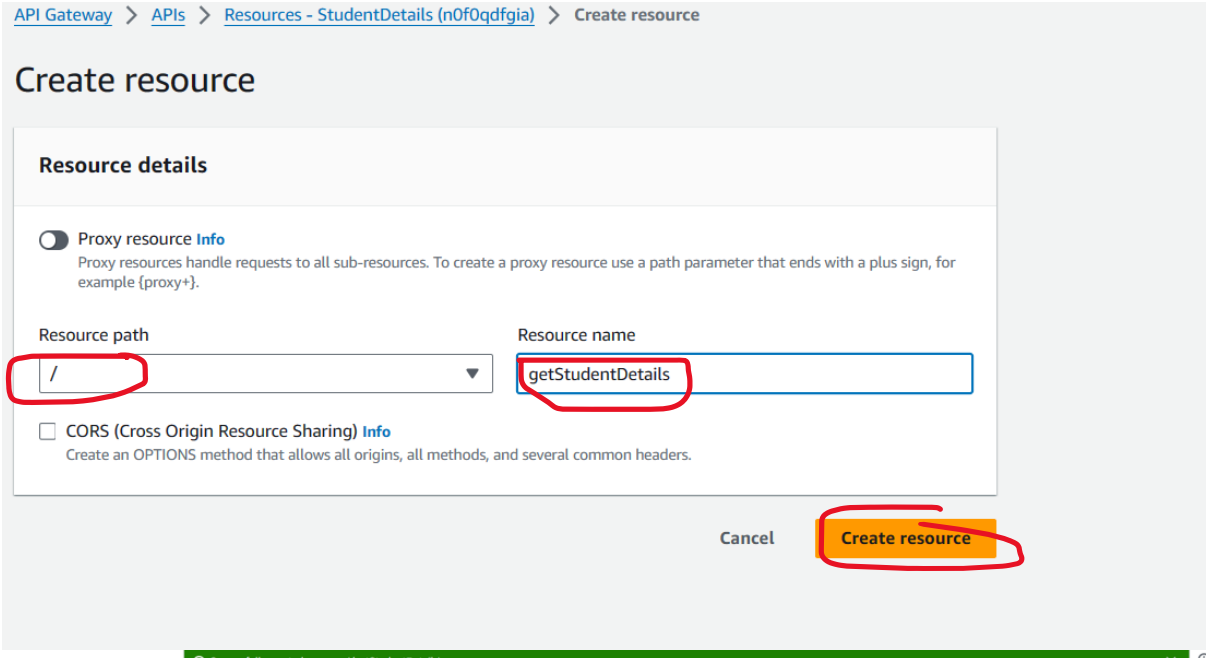
Lets create another resource to get the student details.

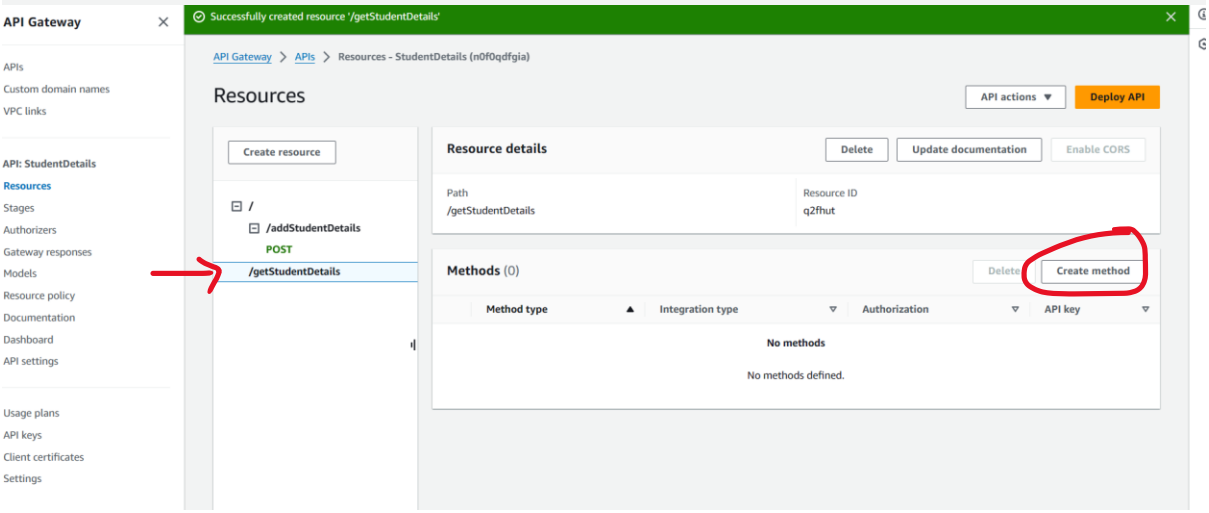


Click on / to go to root path .

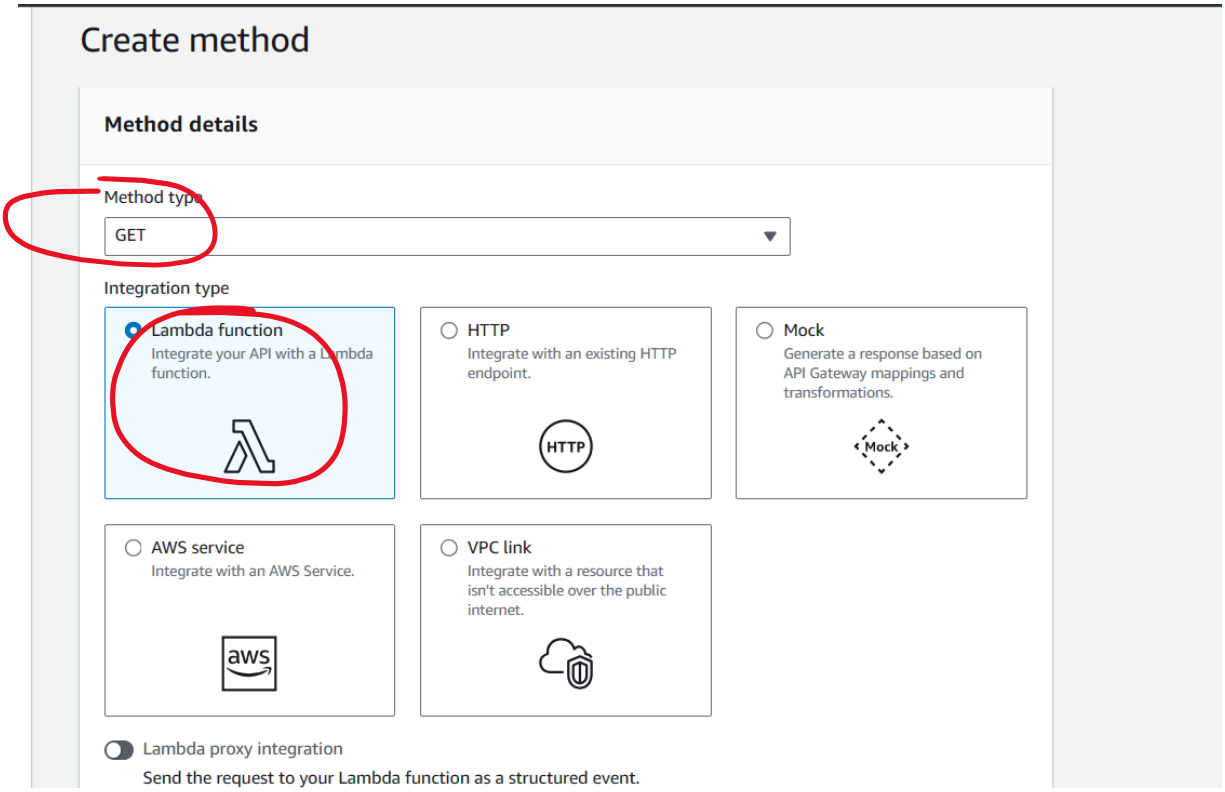


Now click on create resource

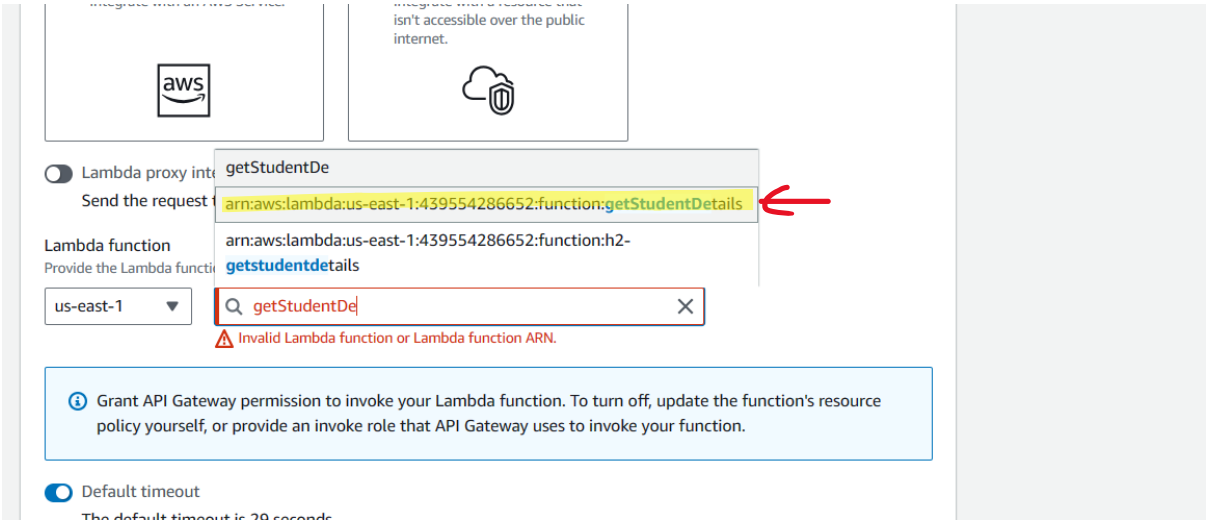




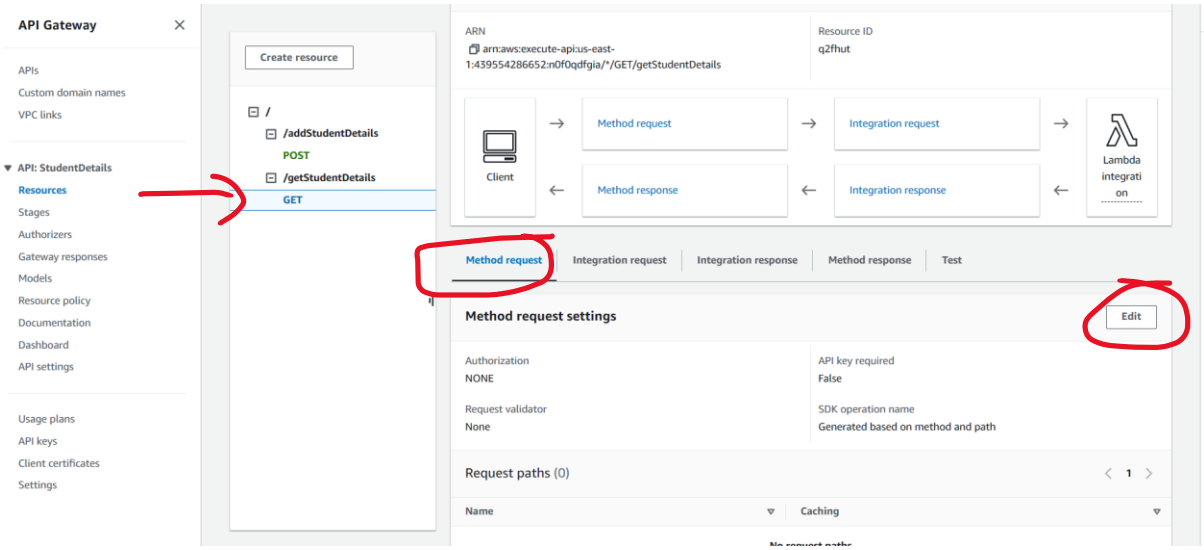
Click on create method in **getStudentDetails**.
Method Type : GET
Integration type : Lambda



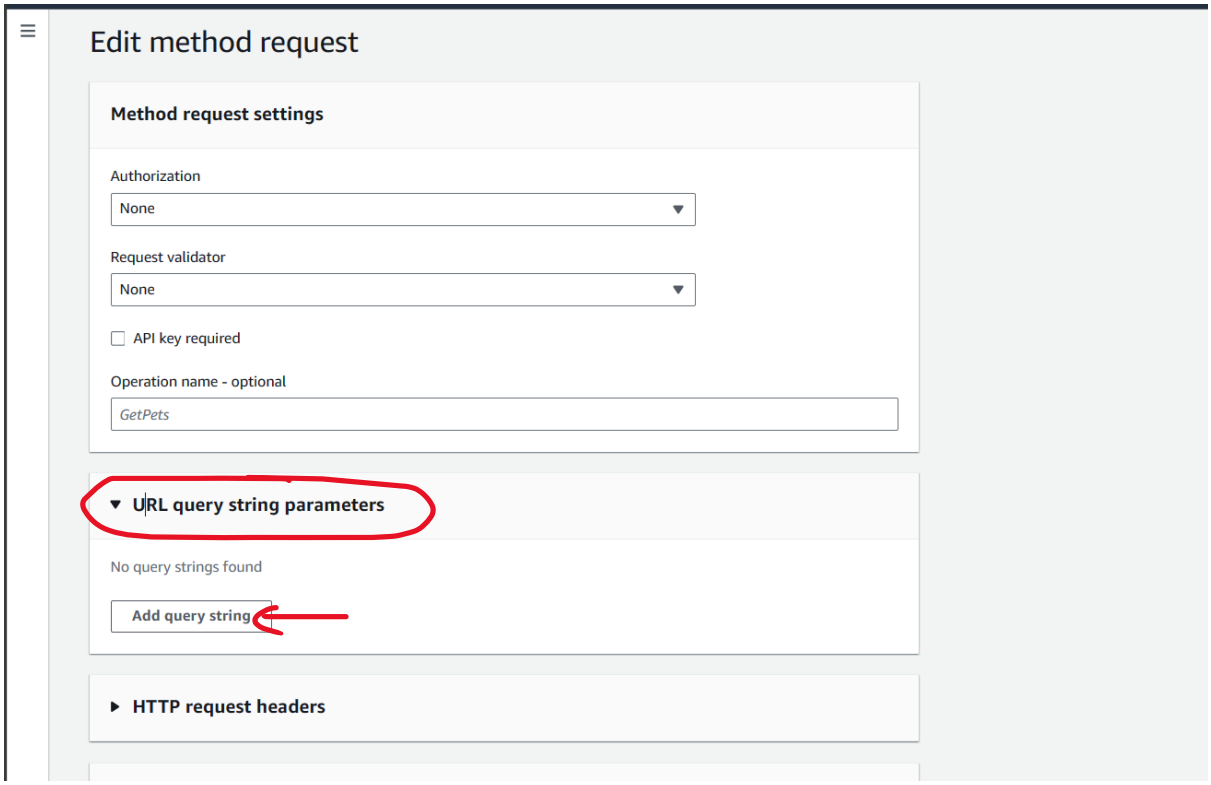
Then in Lambda function option search for getStudentDetails Lambda function.



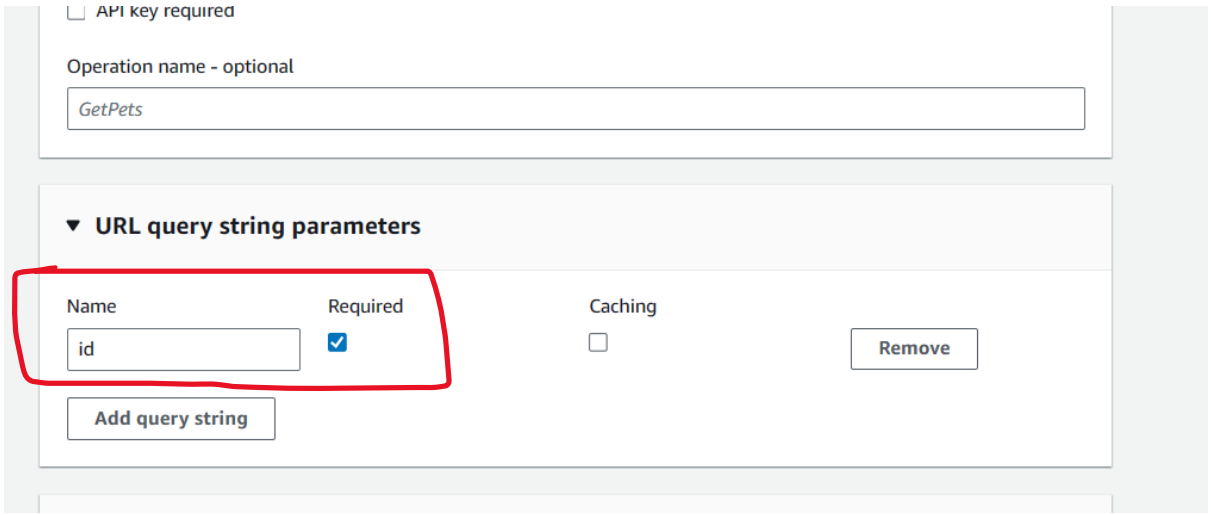
Next , Click on **create method**.



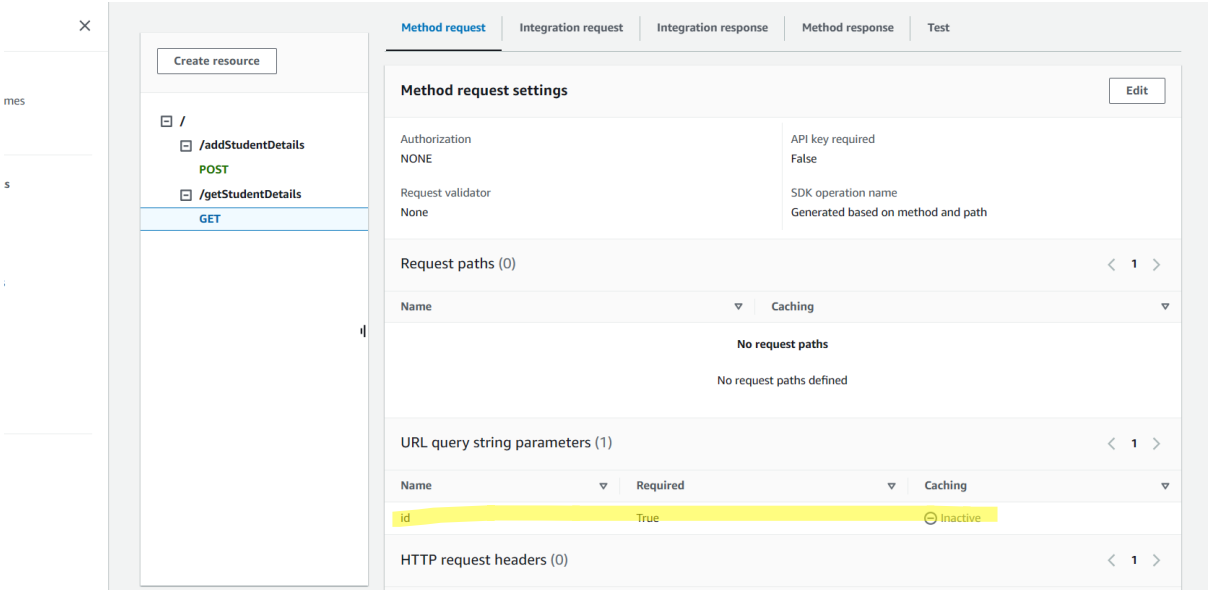
Go to Method request.
Click on Edit.



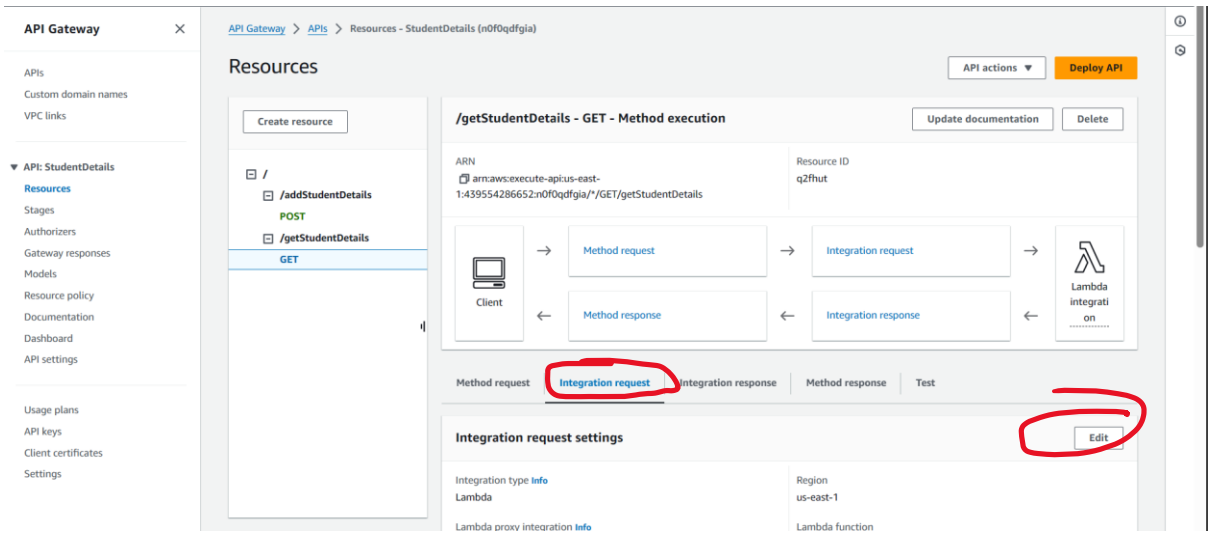
Click on Add query string.



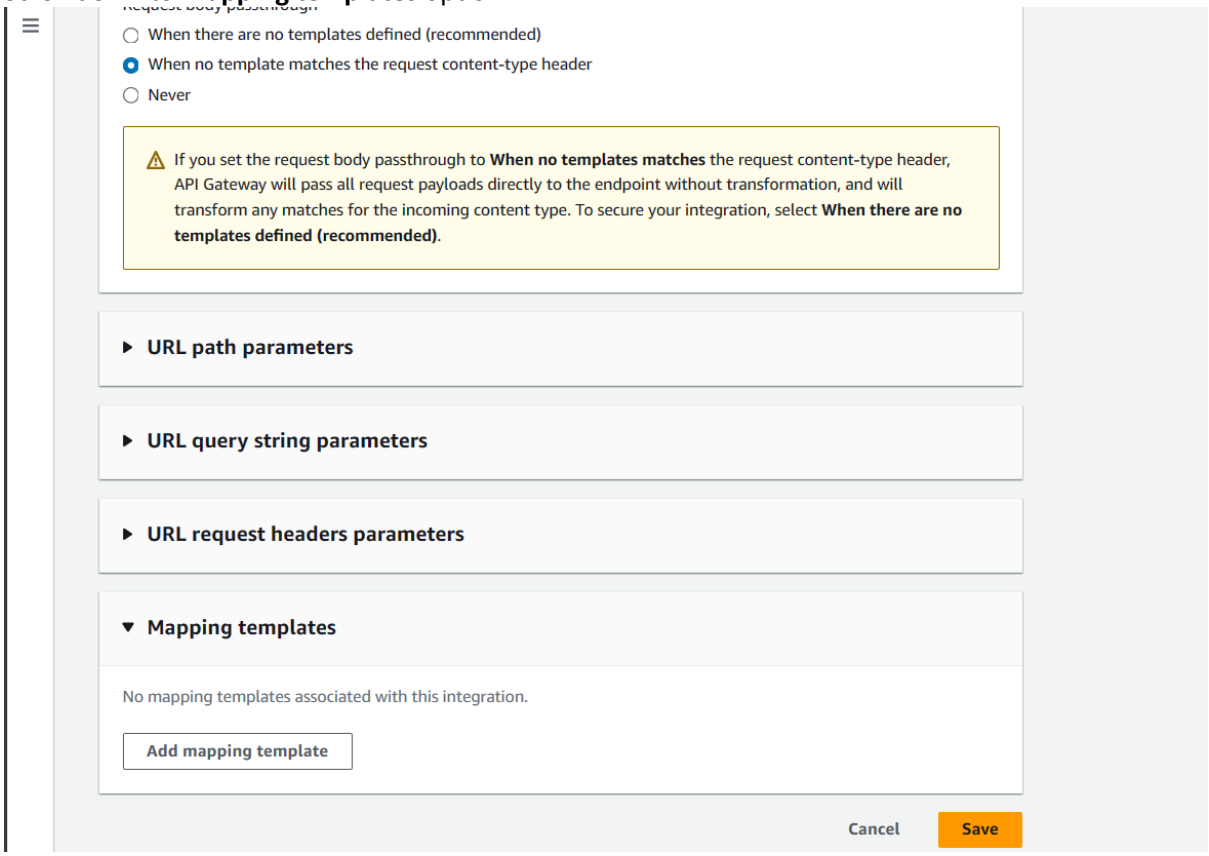
Scroll down and click on **save** .



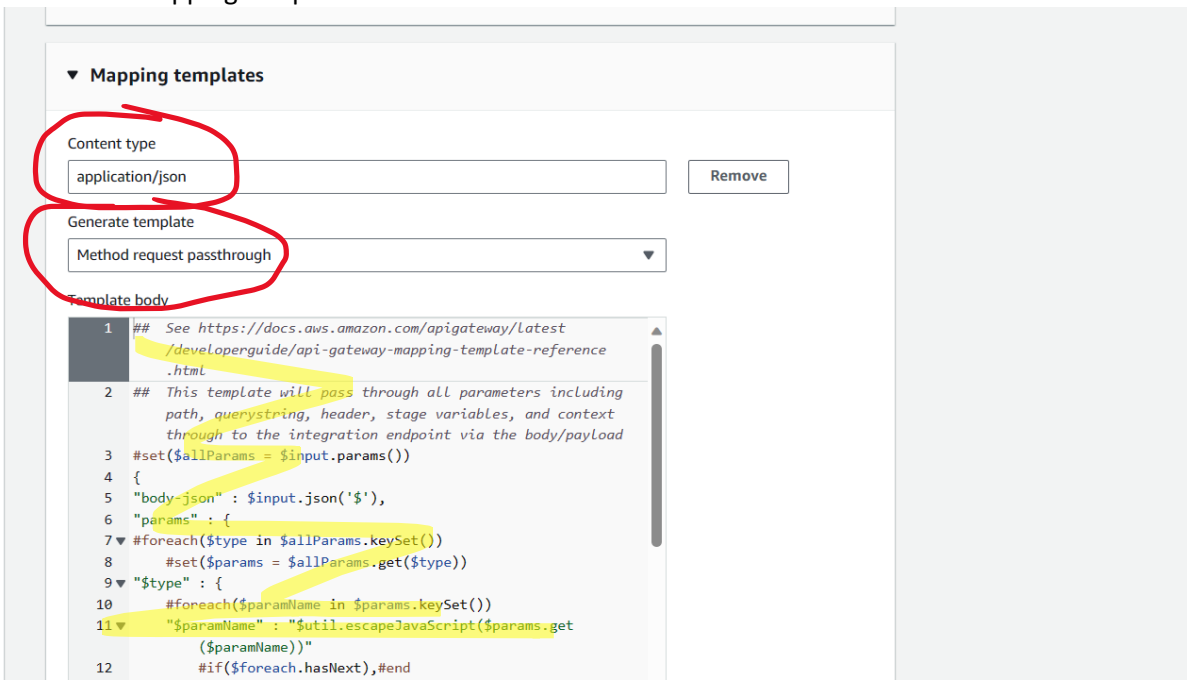
Click on Integration Request.
Click on edit.



Scroll down to **Mapping templates** option.

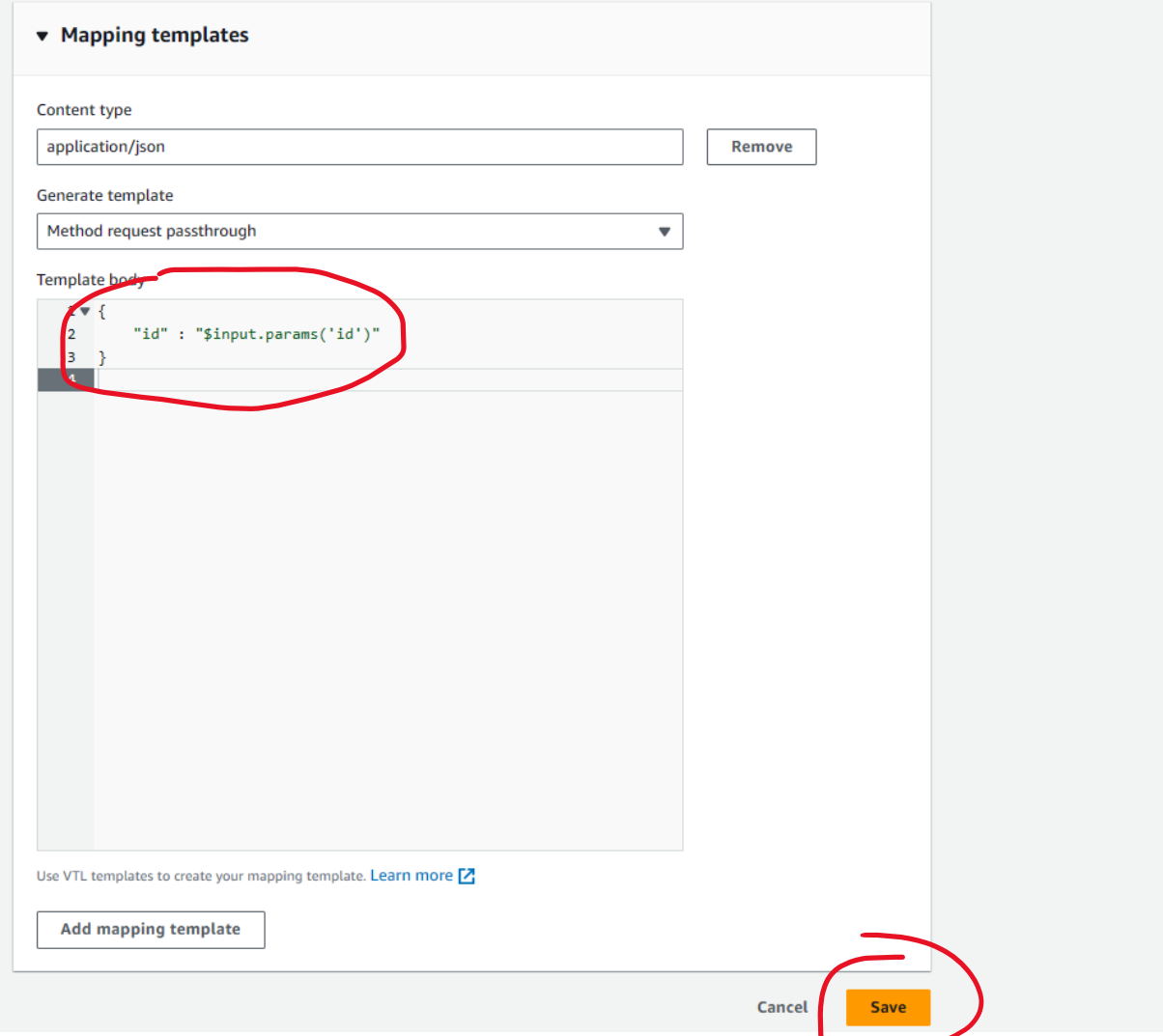


Click on add mapping template.

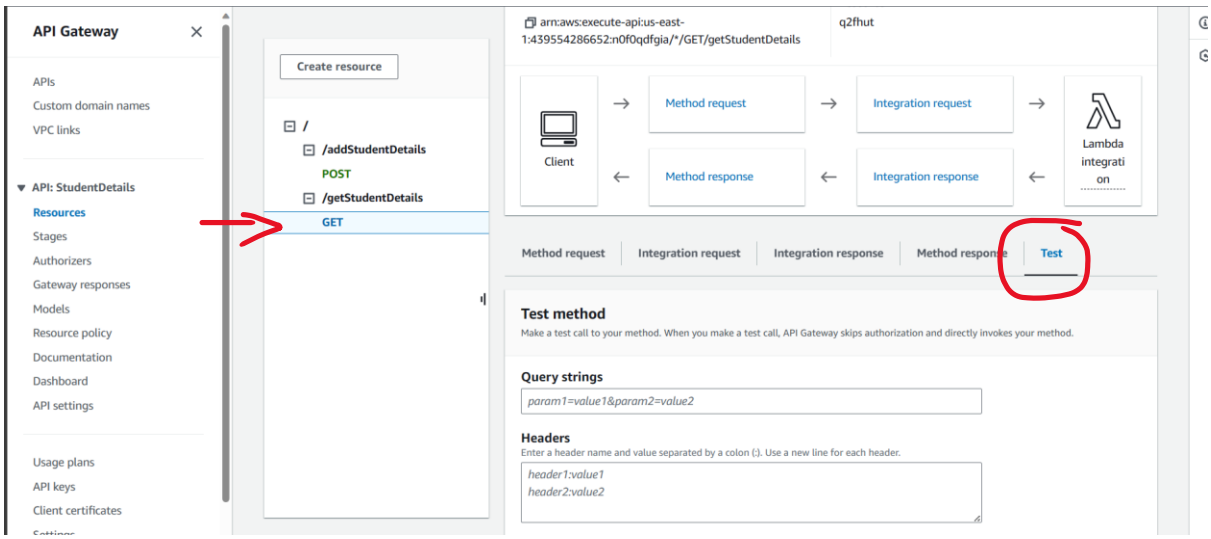


Replace the Template body with the below one :

```
{
   "id" : "$input.params('id')"
}
```

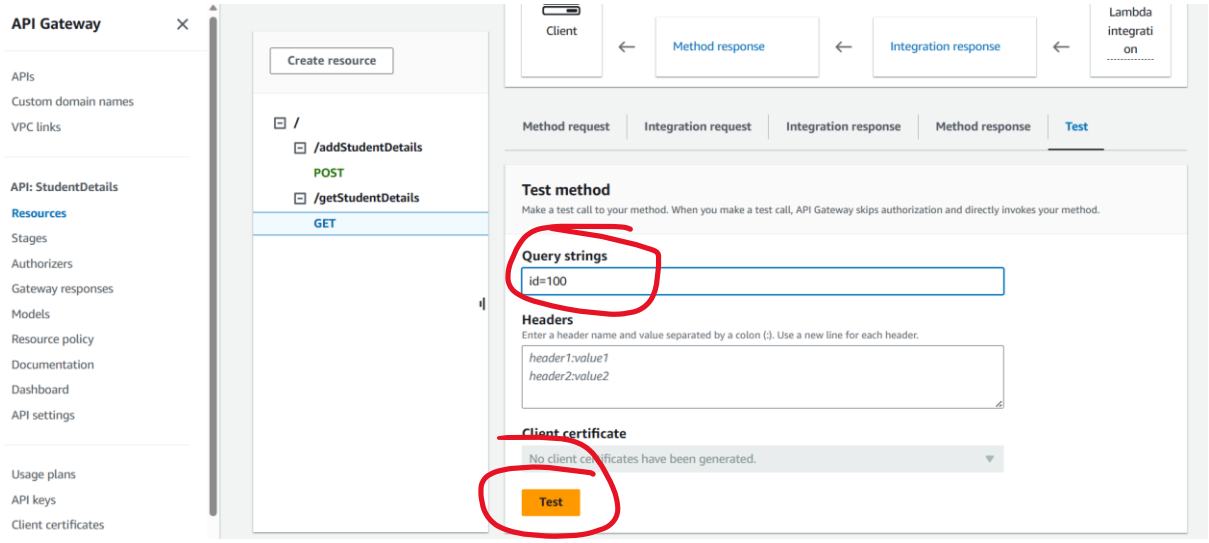


Click on save.

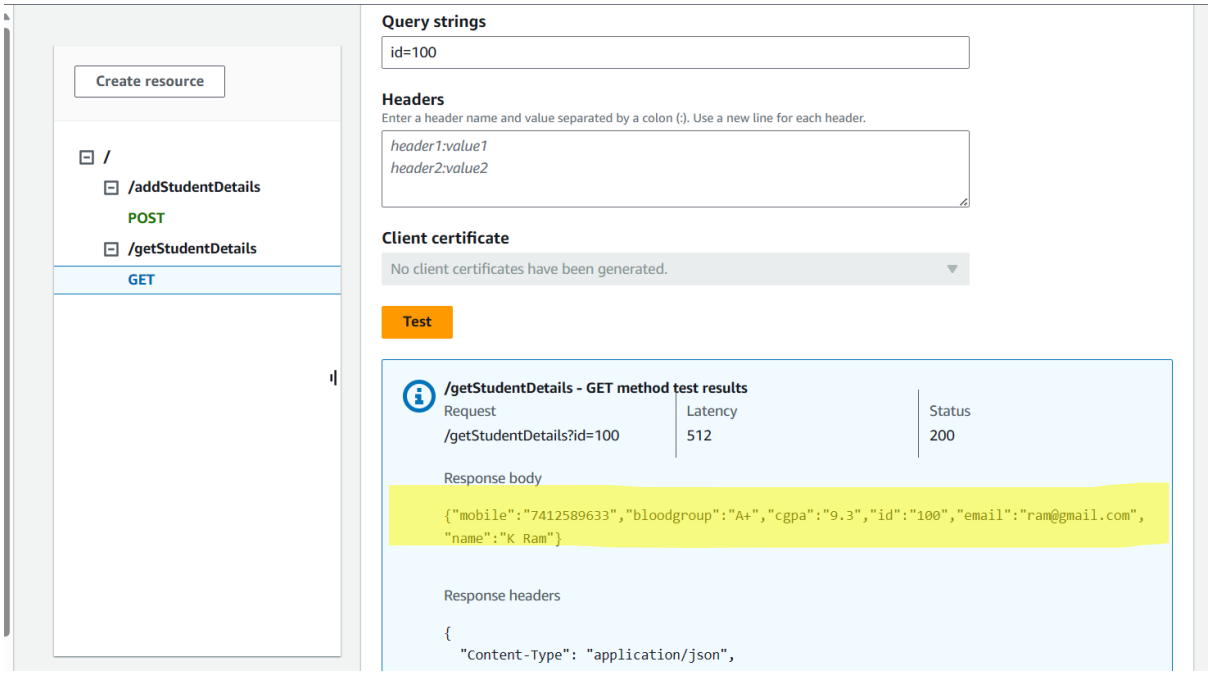


Click on Test.

In the place of Query Strings add
Id=100 (any id which is present in dynamodb table StudentDetails)



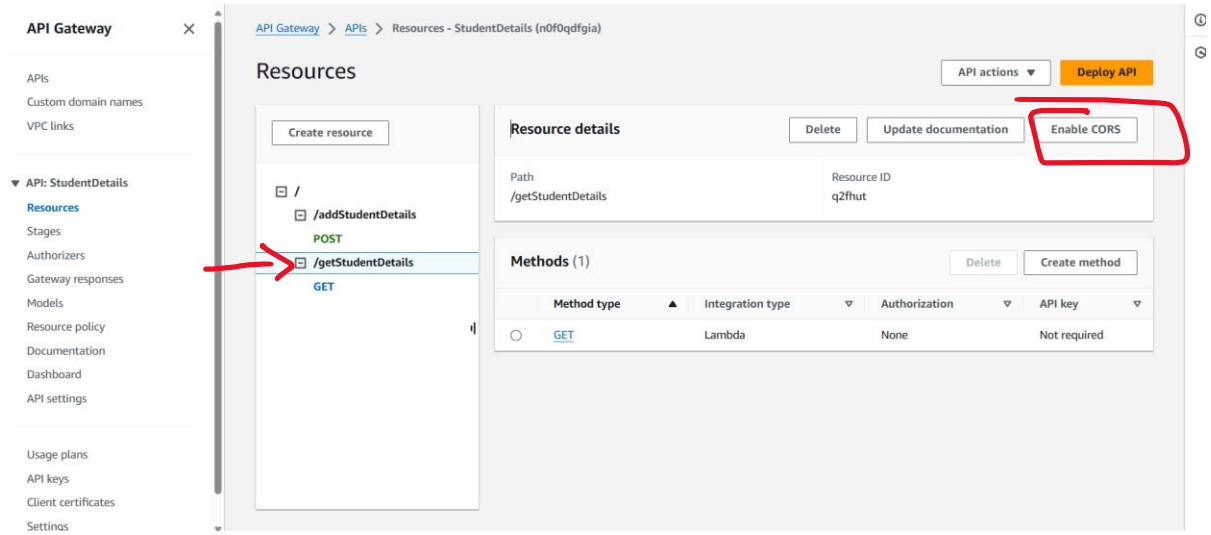
Click on **Test**.



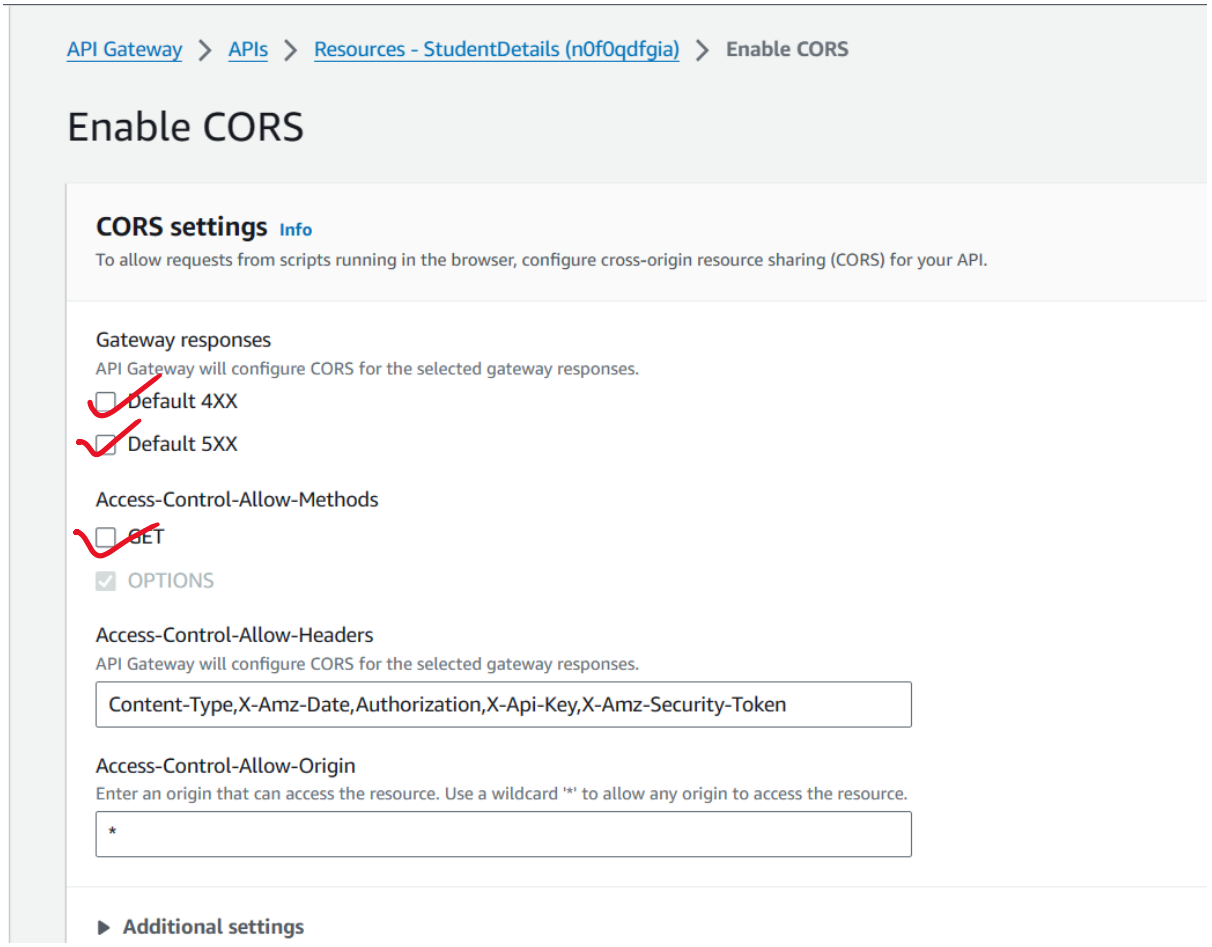
If any error in response check the above steps again.

Now, Both addStudentDetails and getStudentDetails endpoints are working . Lets **deploy** the **StudentDetails** API.

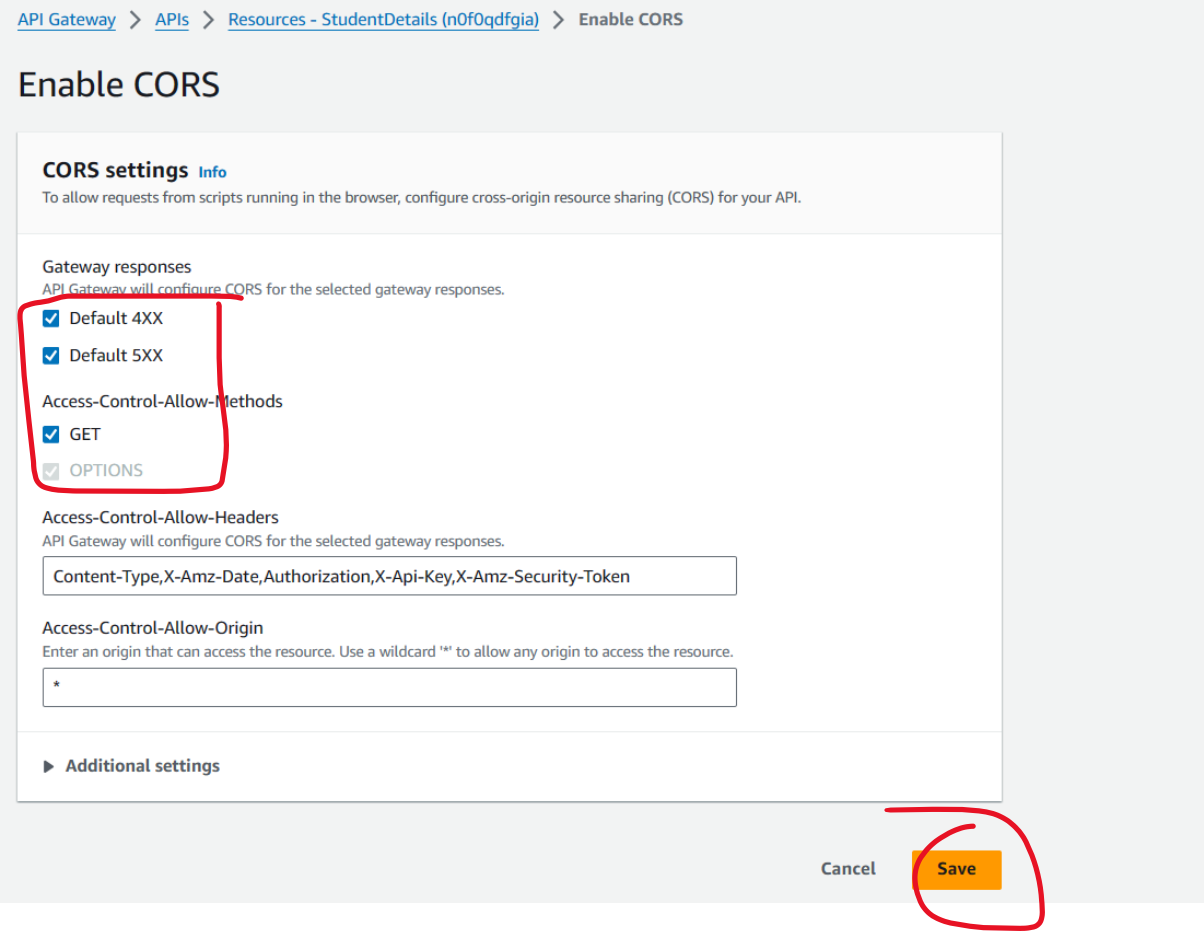
Lets 1st enable the CORS for both addStudentDetails and getStudentDetails resources, because we are going to access them form S3.

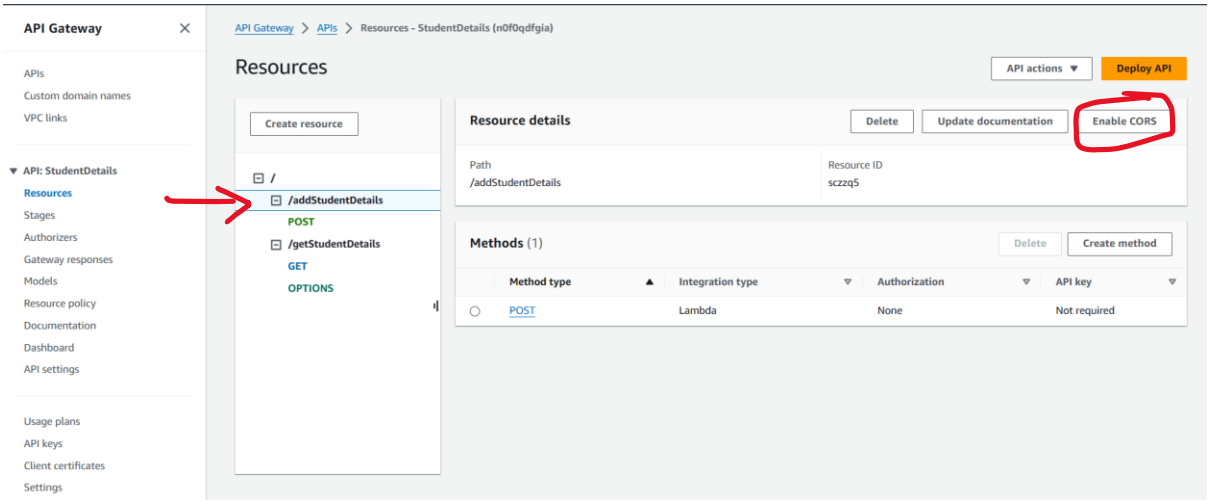


First , click on getStudentDetails resources then click on Enable CORS.

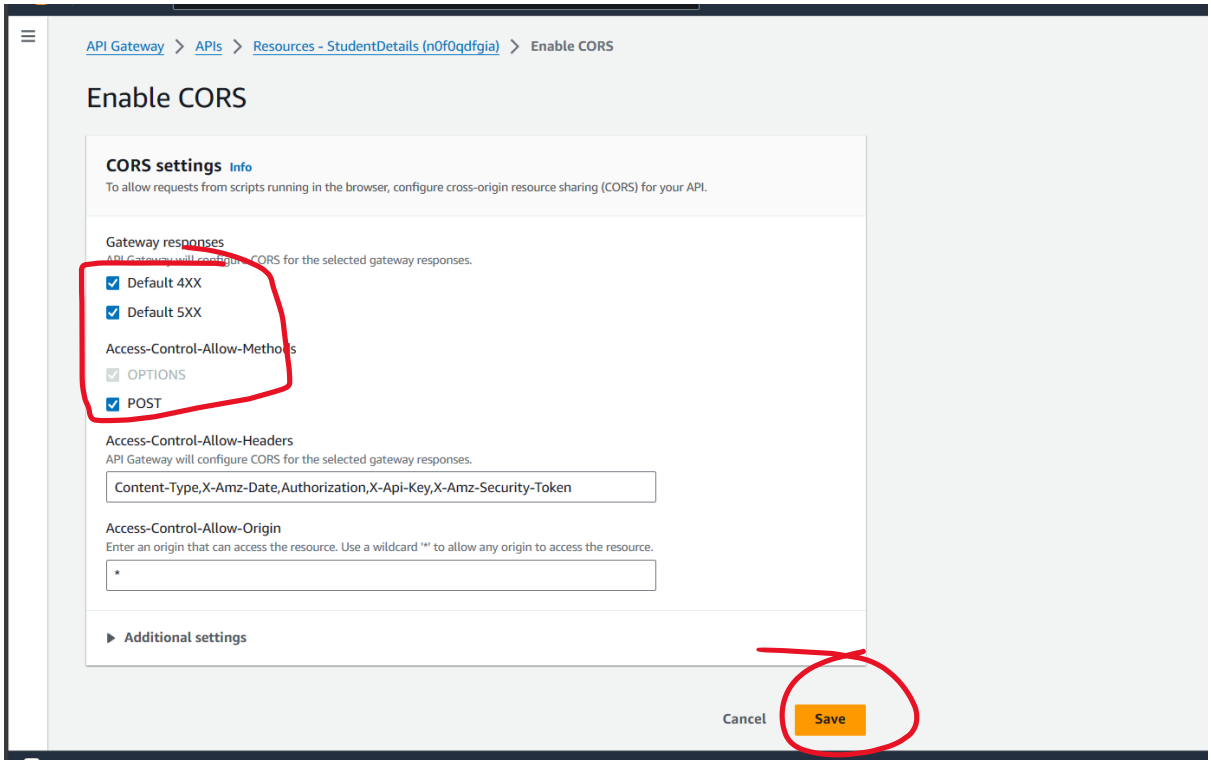


Check all the options and click on save.



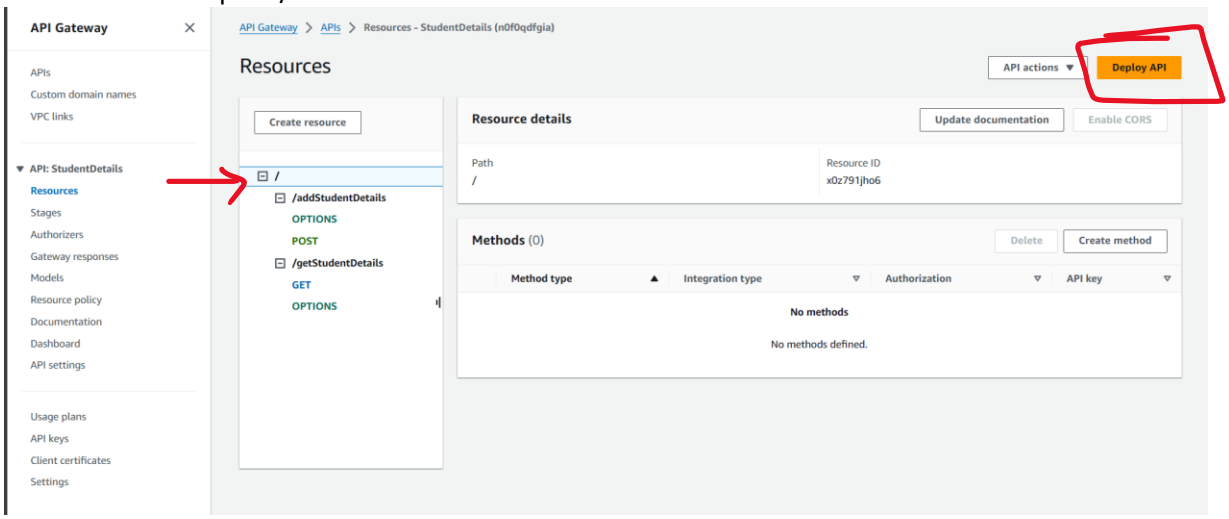


Now lets do the same for addStudentDetails resource.

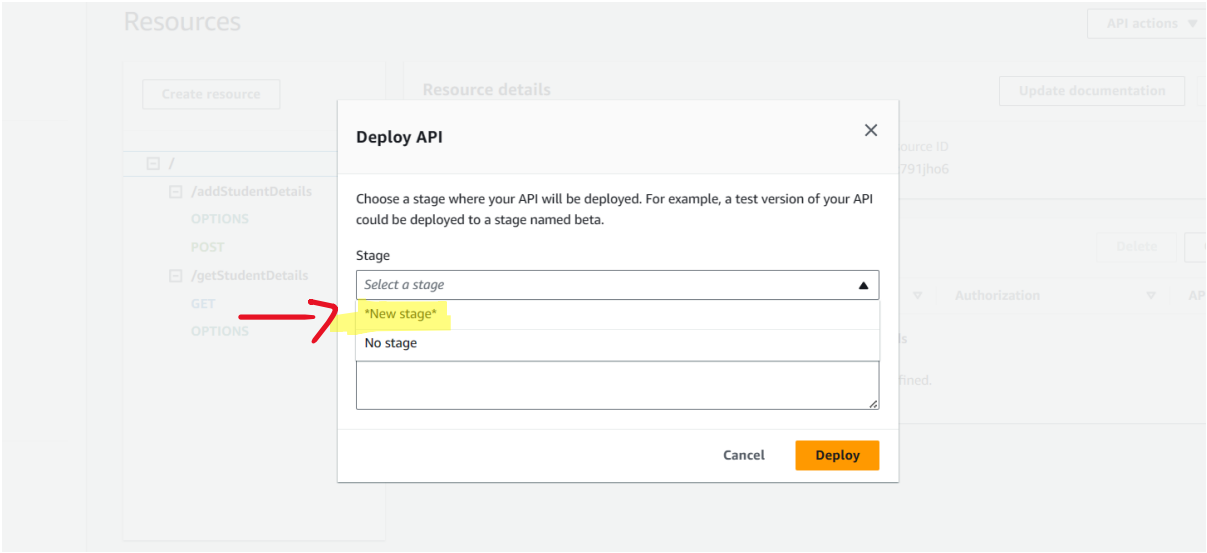


Click on save .

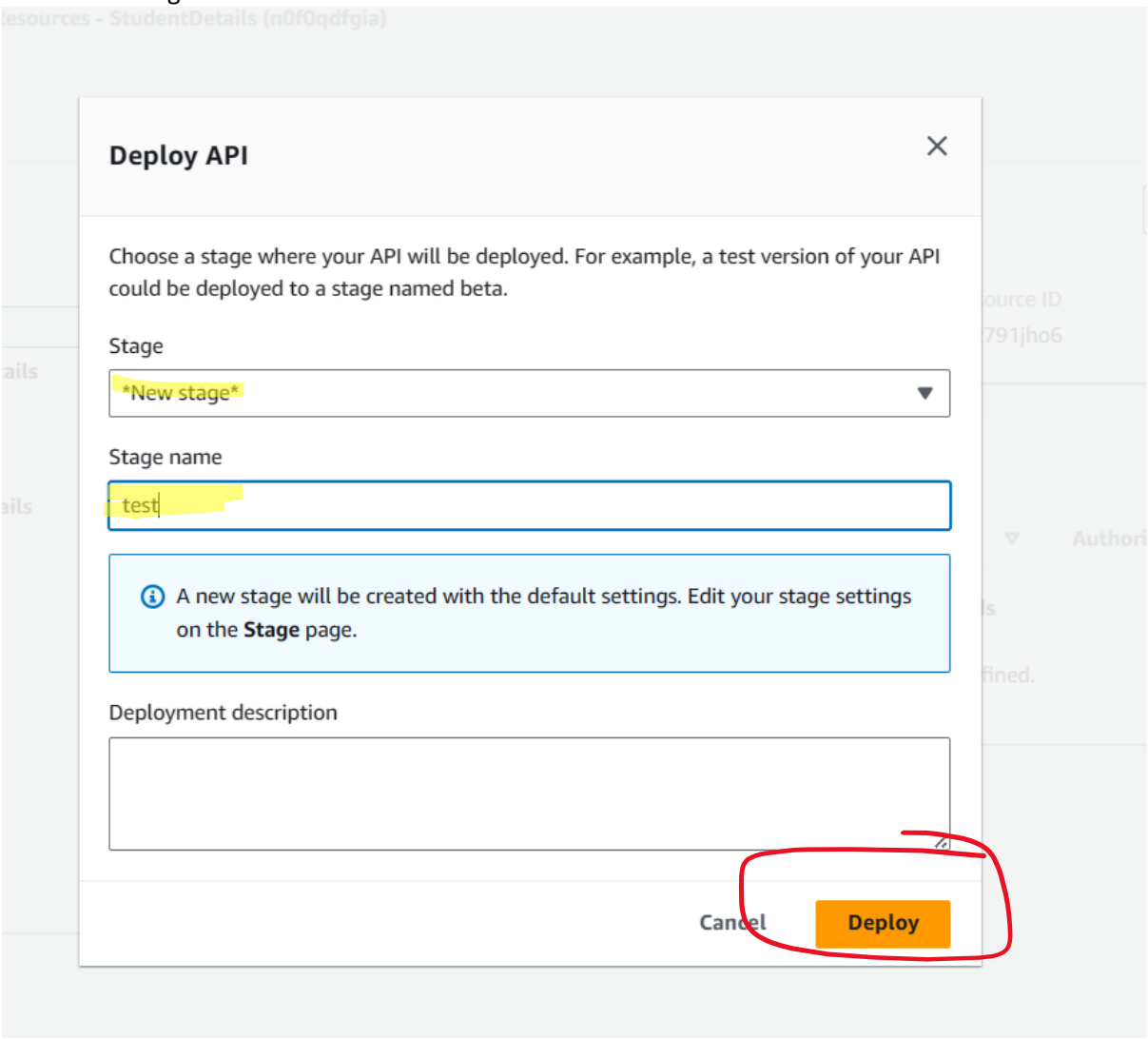
Now click on root path /



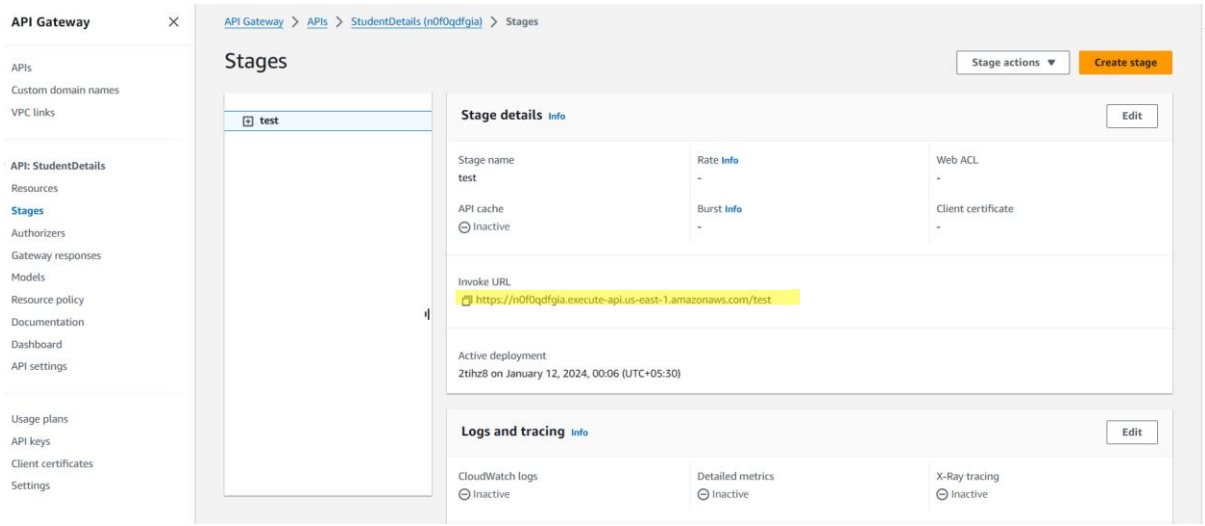
Deploy the API.



Select New stage.



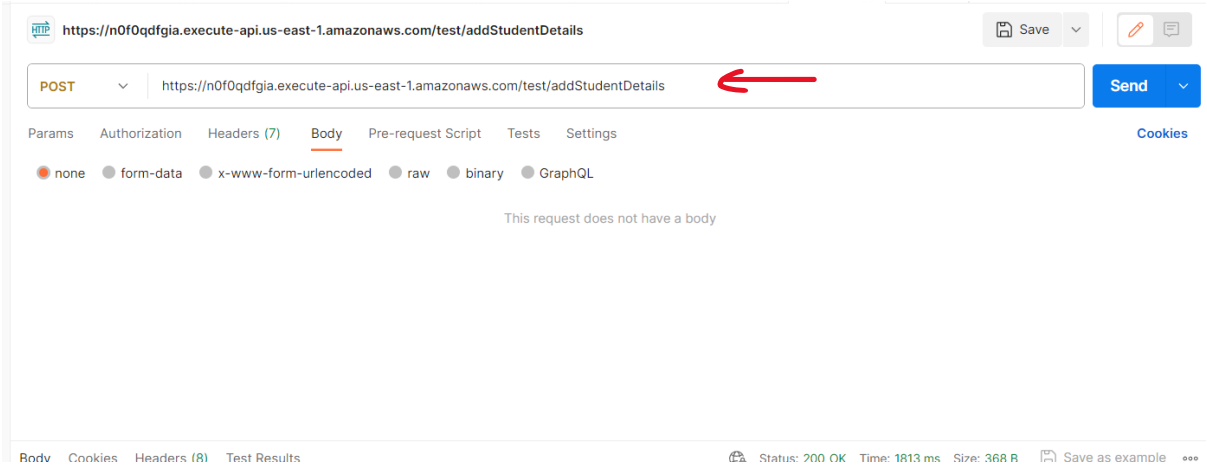
Click on **Deploy** .



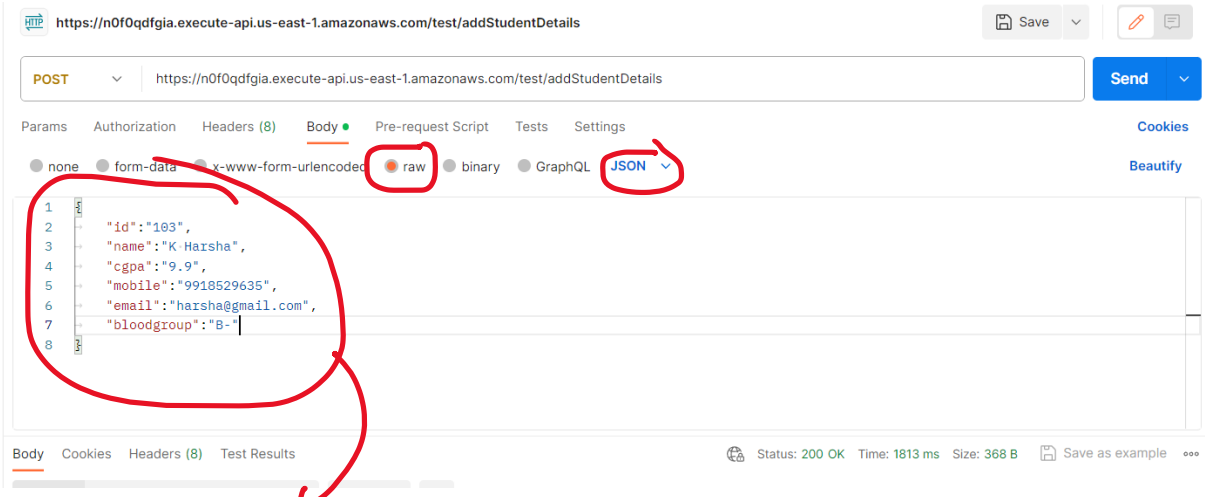
Copy the invoke URL.
And wait for 2 mins to get the API Deployed.

Now lets test the API with POSTMAN.

To add student details the endpoint will be
Invoke URL : your_api_gateway_url/addStudentDetails
And method will POST.

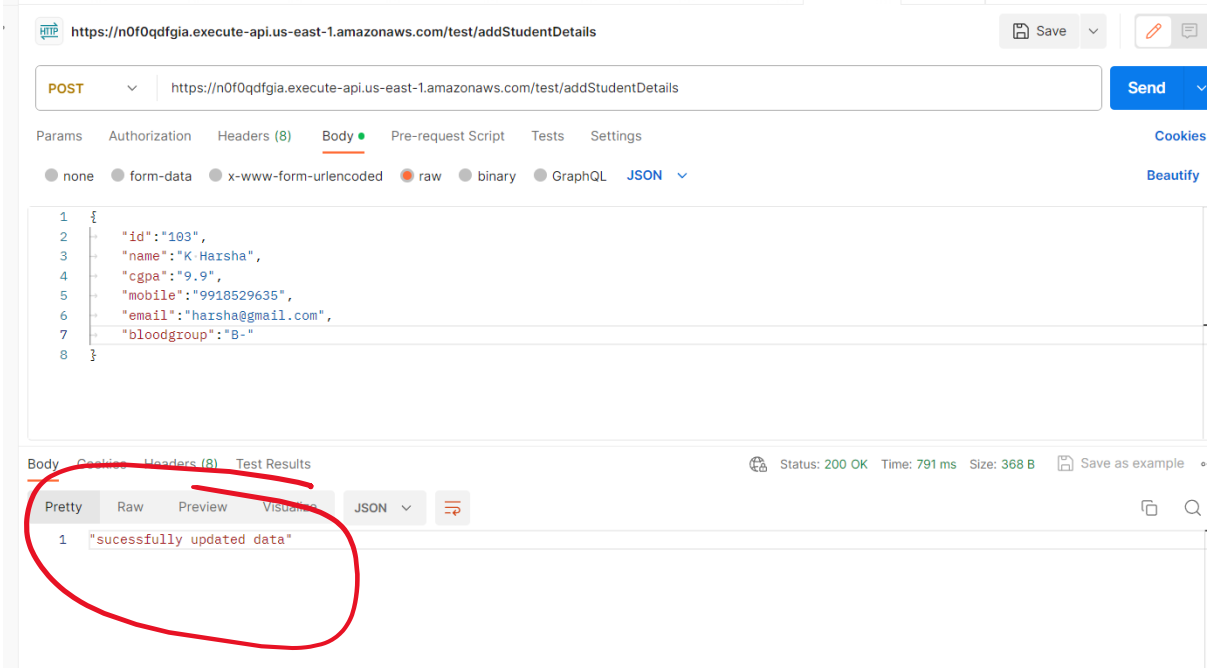


Goto raw

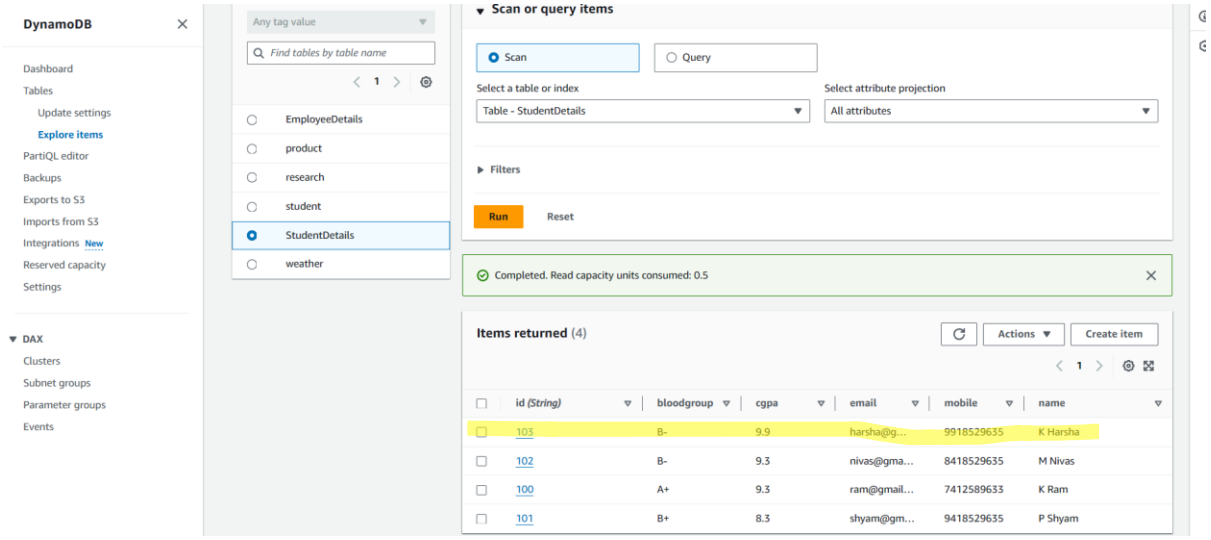



```
{
    "id":"103",
    "name":"K Harsha",
    "cgpa":"9.9",
    "mobile":"9918529635",
    "email":"harsha@gmail.com",
    "bloodgroup":"B-"
}
```

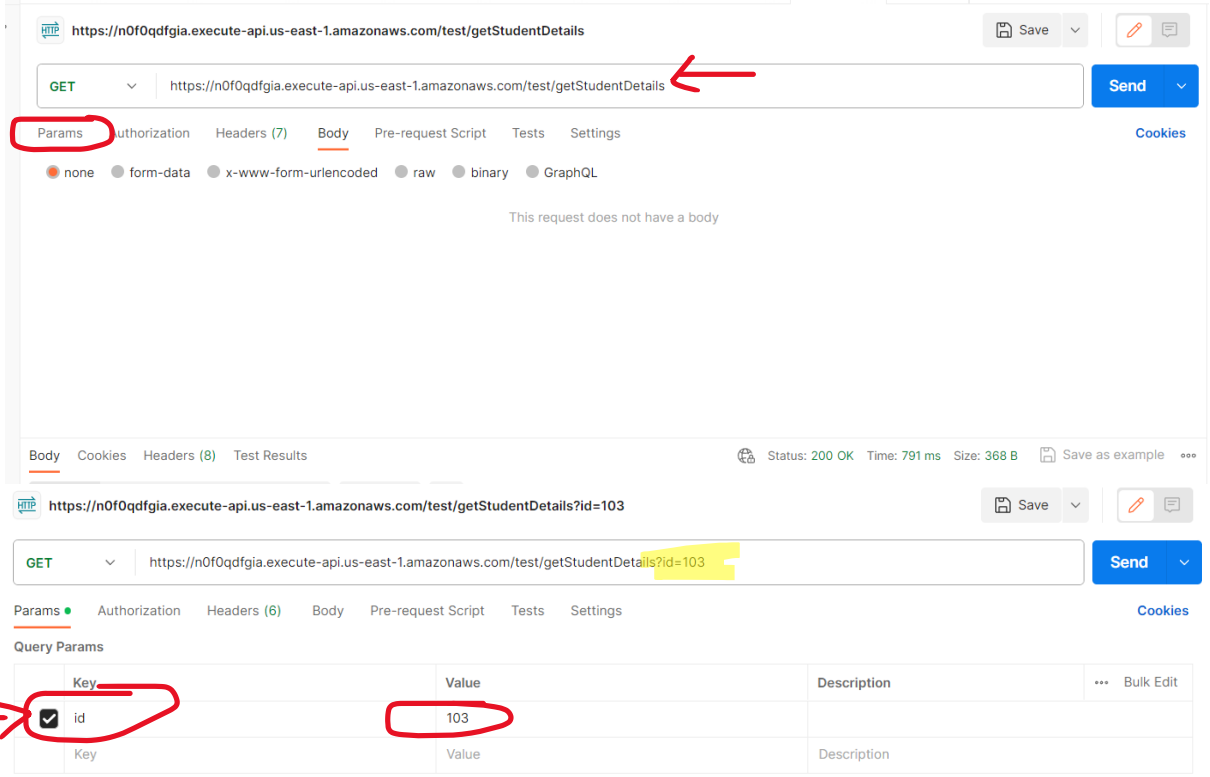
Click on Send.



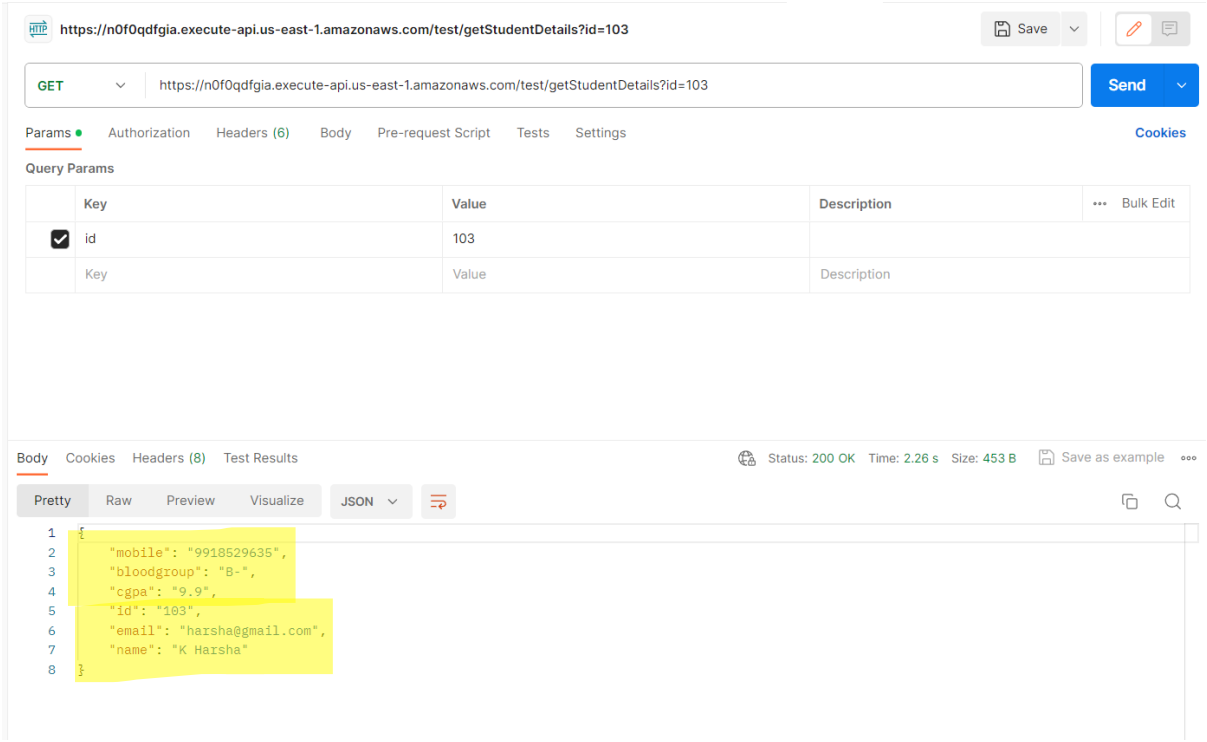
Lets check in dynamodb table **StudentDetails**.



To get student details the endpoint will be
Invoke URL : your_api_gateway_url/getStudentDetails
And method will GET.



Click on Send.



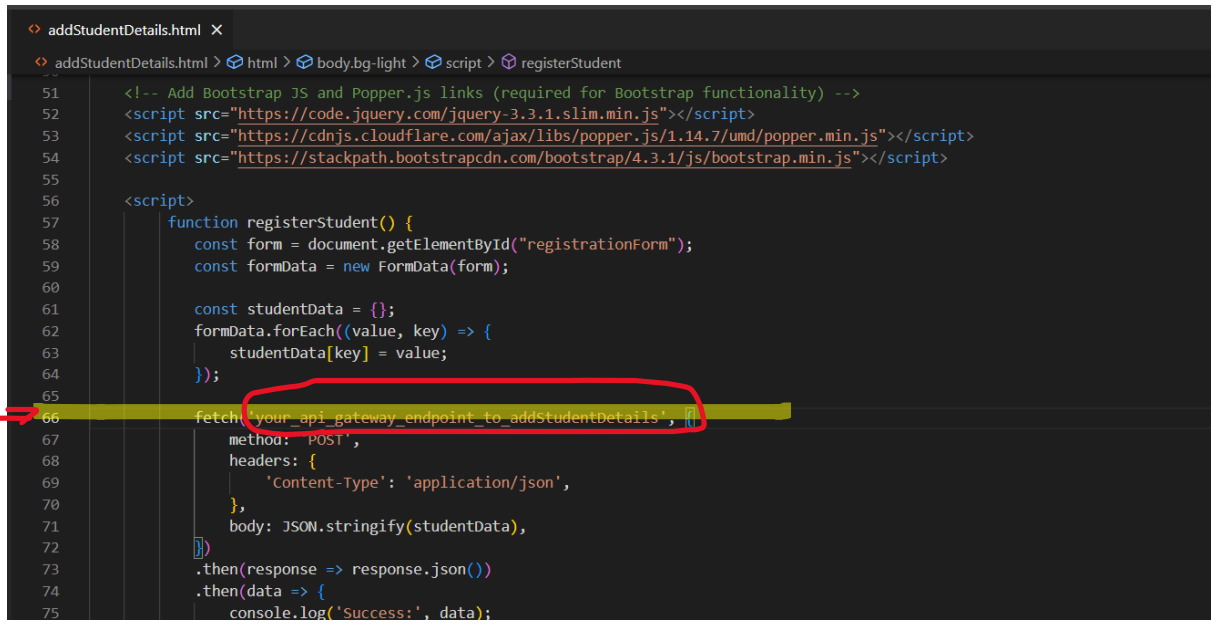
The API endpoints are working fine. Lets go to S3 and host Student Details static website.

NOTE : you can download the website files from the below link .

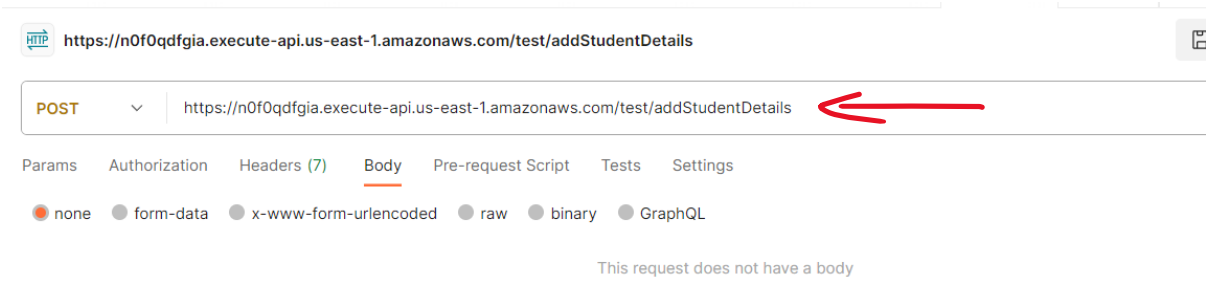
Student-Details-HA-2 - OneDrive (sharepoint.com)

After downloading the files. Place them in a folder.

Open addStudentDetails.html in notepad or vsCode or any code editor.



On line 66 you need to replace it with the addStudentDetails api gateway endpoint which is used in postman.




```
addStudentDetails.html
addStudentDetails.html > html > body.bg-light > script > registerStudent
51      <!-- Add Bootstrap JS and Popper.js links (required for Bootstrap functionality) -->
52      <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
53      <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>
54      <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>
55
56      <script>
57          function registerStudent() {
58              const form = document.getElementById("registrationForm");
59              const formData = new FormData(form);
60
61              const studentData = {};
62              formData.forEach((value, key) => {
63                  studentData[key] = value;
64              });
65
66              fetch('https://n0f0qdfgia.execute-api.us-east-1.amazonaws.com/test/addStudentDetails', {
67                  method: 'POST',
68                  headers: {
69                      'Content-Type': 'application/json',
70                  },
71                  body: JSON.stringify(studentData),
72              })
73              .then(response => response.json())
74              .then(data => {
75                  console.log('Success:', data);
76                  // You can handle success actions here
```

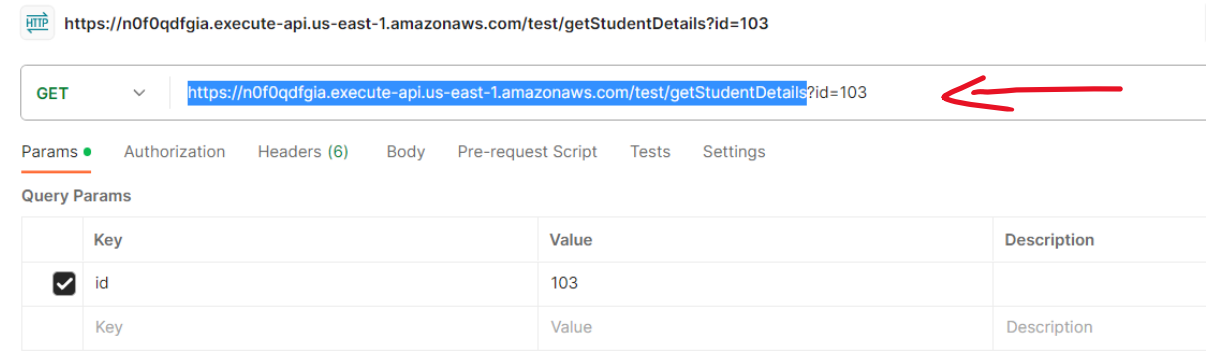
Save the file.

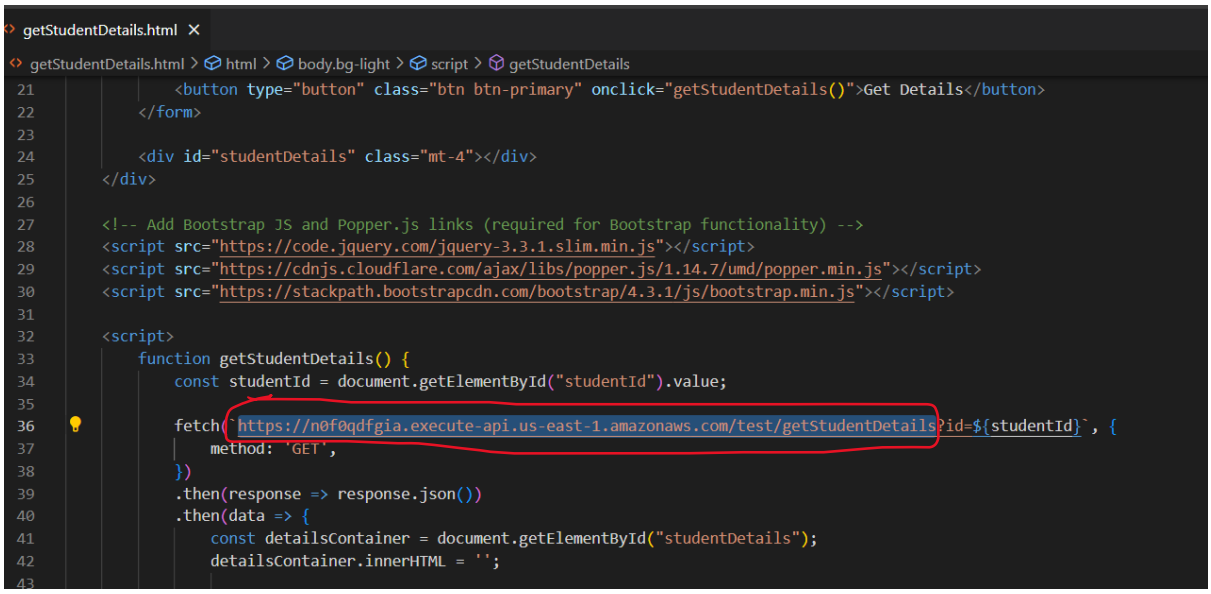
Now open getStudentDetails in notepad or vsCode or any other code editor.

```
getStudentDetails.html
getStudentDetails.html > html > body.bg-light > script > getStudentDetails
21                  <button type="button" class="btn btn-primary" onclick="getStudentDetails()">Get Details</button>
22              </form>
23
24              <div id="studentDetails" class="mt-4"></div>
25          </div>
26
27          <!-- Add Bootstrap JS and Popper.js links (required for Bootstrap functionality) -->
28          <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
29          <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>
30          <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>
31
32          <script>
33              function getStudentDetails() {
34                  const studentId = document.getElementById("studentId").value;
35
36                  fetch(`your_api_gateway_endpoint_to_getStudentDetailsById?id=${studentId}`, {
37                      method: 'GET',
38                  })
39                  .then(response => response.json())
40                  .then(data => {
41                      const detailsContainer = document.getElementById("studentDetails");
42                      detailsContainer.innerHTML = '';
43
```

On line 36 you need to replace it with the getStudentDetails api gateway endpoint which is used in postman.

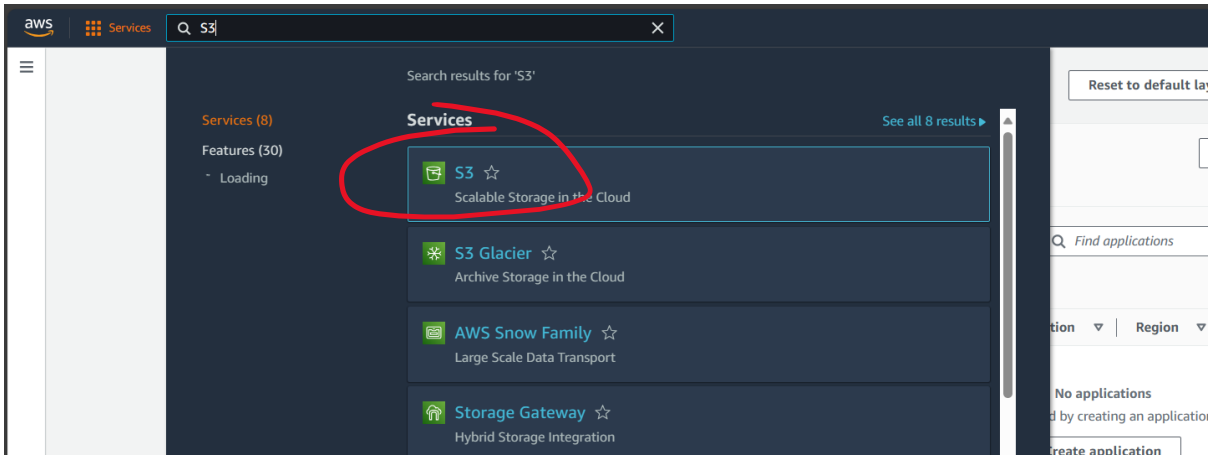
Replace only the part before question mark (?id=) .

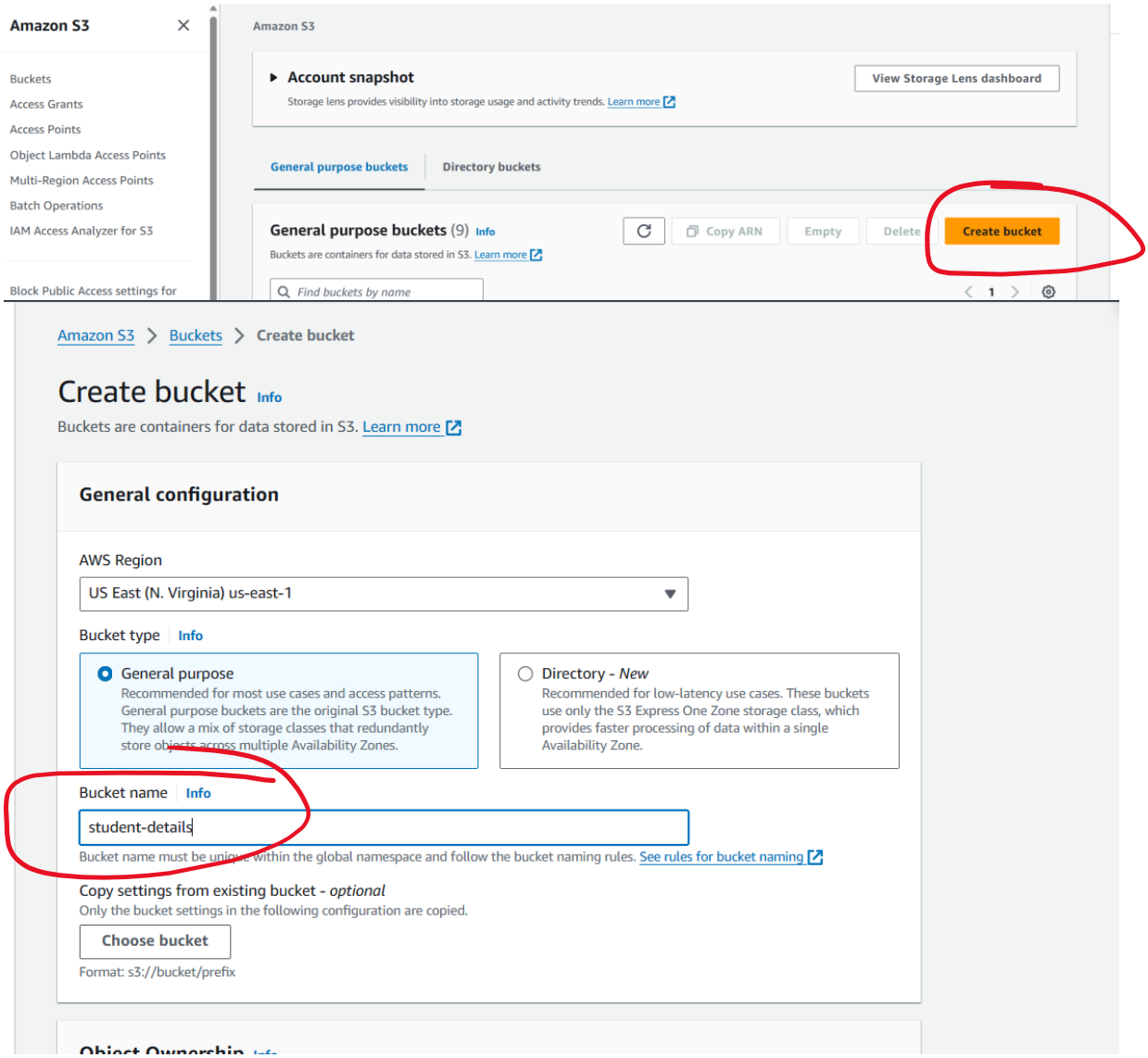




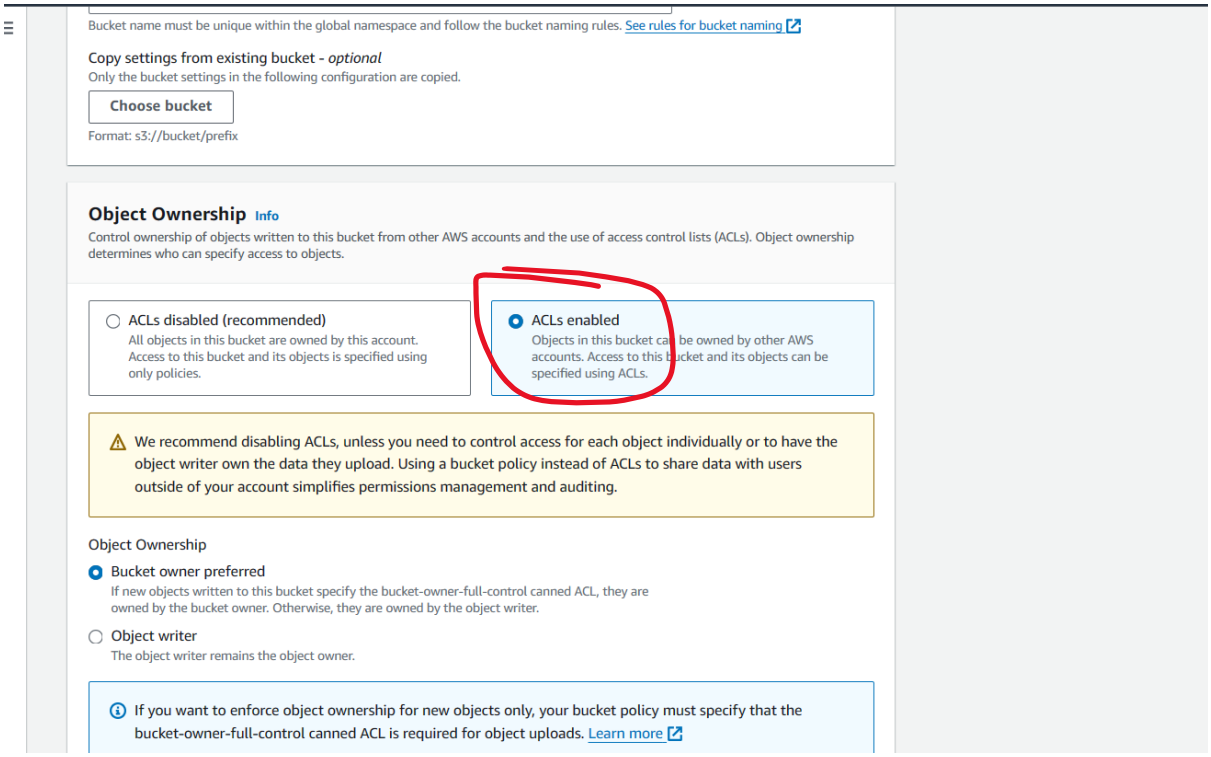
Save the file.

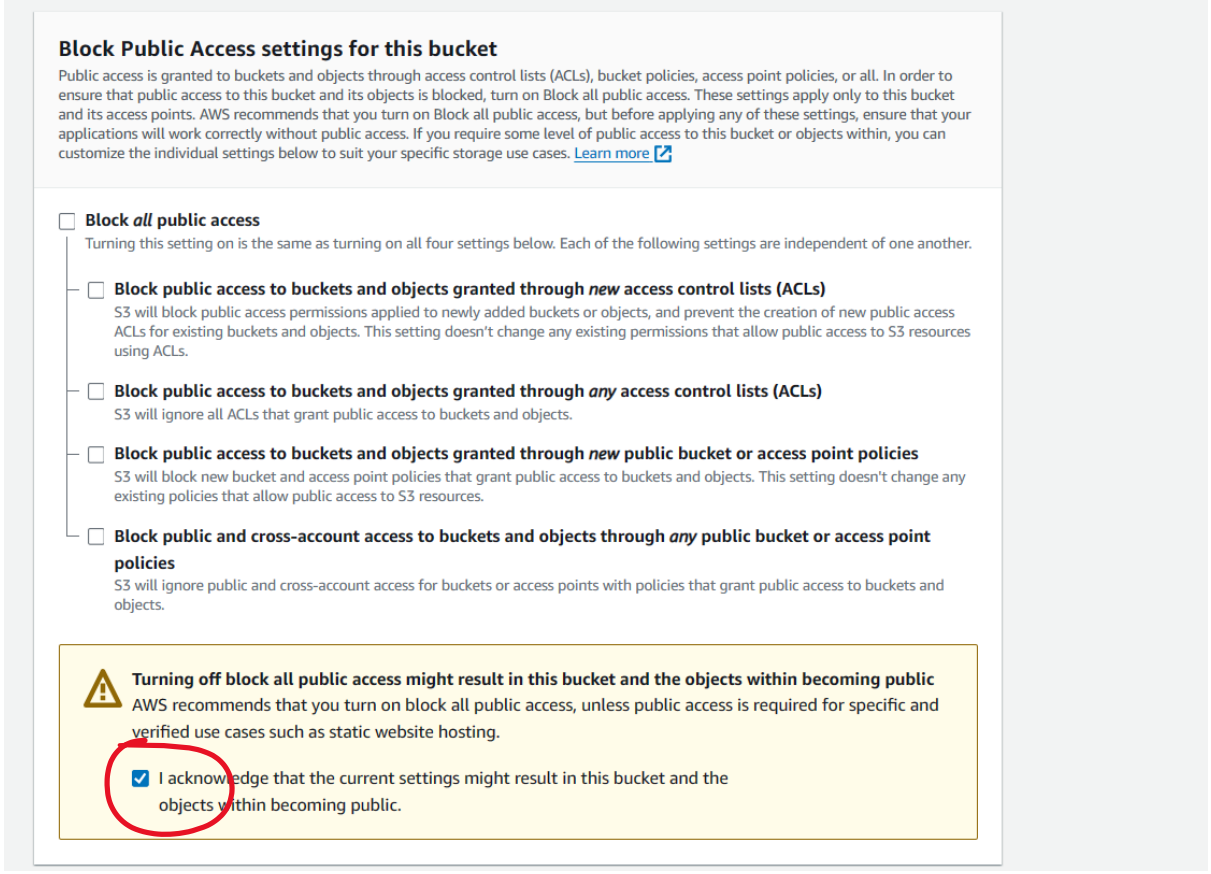
S3

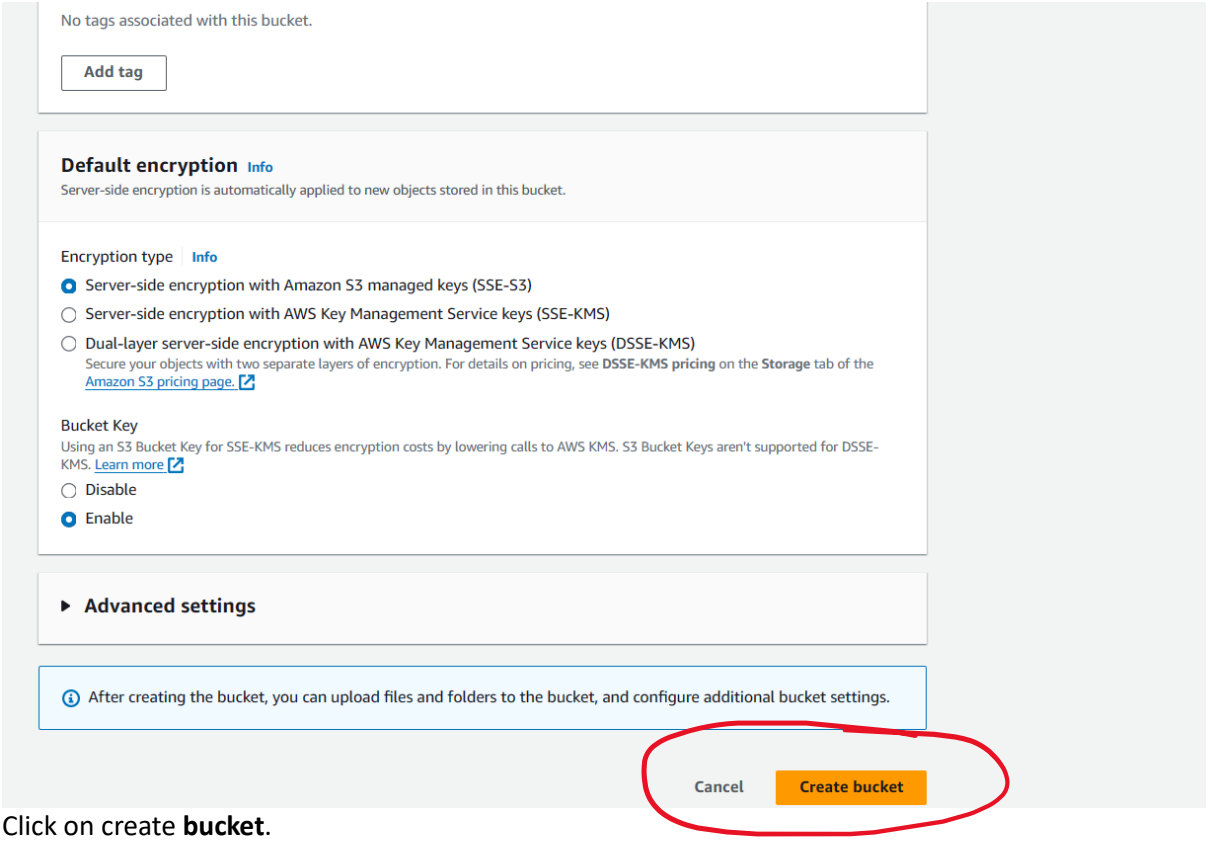




You can use any valid name.

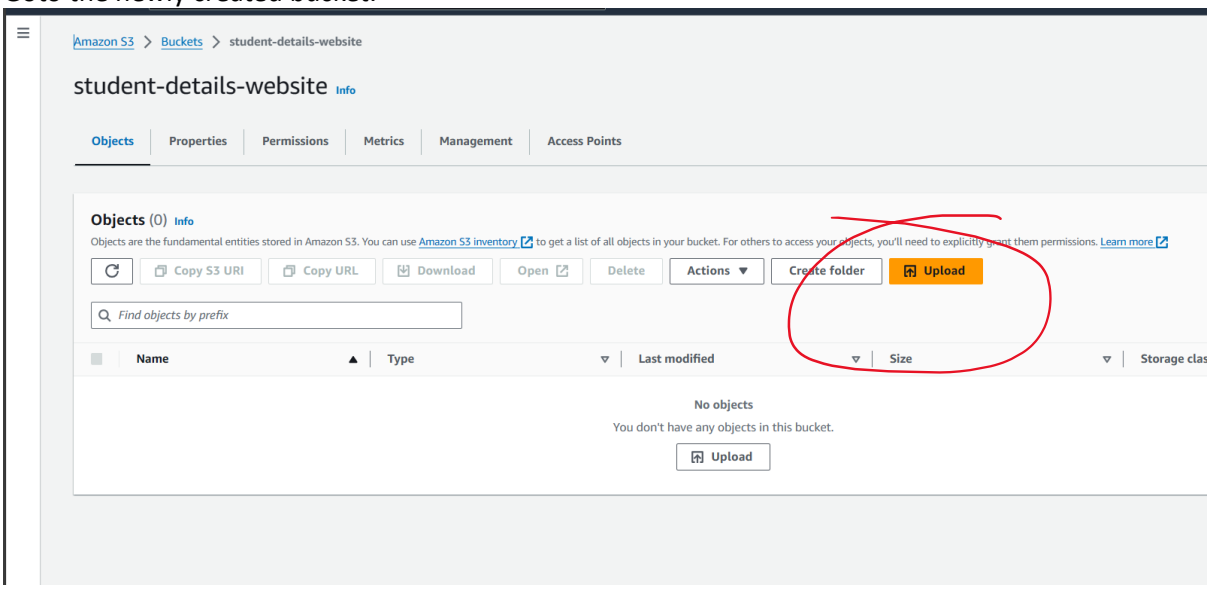

Bucket name must be unique within the global namespace and follow the bucket naming rules. See rules for bucket naming
Copy settings from existing bucket - optional
Only the bucket settings in the following configuration are copied.
Choose bucket
Format: s3://bucket/prefix
Object Ownership Info
Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.
ACLs disabled (recommended)
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.
ACLs enabled
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.
We recommend disabling ACLs, unless you need to control access for each object individually or to have the object writer own the data they upload. Using a bucket policy instead of ACLs to share data with users outside of your account simplifies permissions management and auditing.
Object Ownership
Bucket owner preferred
If new objects written to this bucket specify the bucket-owner-full-control canned ACL, they are owned by the bucket owner. Otherwise, they are owned by the object writer.
Object writer
The object writer remains the object owner.
If you want to enforce object ownership for new objects only, your bucket policy must specify that the bucket-owner-full-control canned ACL is required for object uploads. Learn more


Block Public Access settings for this bucket
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more
Block all public access
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.
Block public access to buckets and objects granted through new access control lists (ACLs)
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
Block public access to buckets and objects granted through any access control lists (ACLs)
S3 will ignore all ACLs that grant public access to buckets and objects.
Block public access to buckets and objects granted through new public bucket or access point policies
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
Block public and cross-account access to buckets and objects through any public bucket or access point policies
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.
Turning off block all public access might result in this bucket and the objects within becoming public
AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.
I acknowledge that the current settings might result in this bucket and the objects within becoming public.

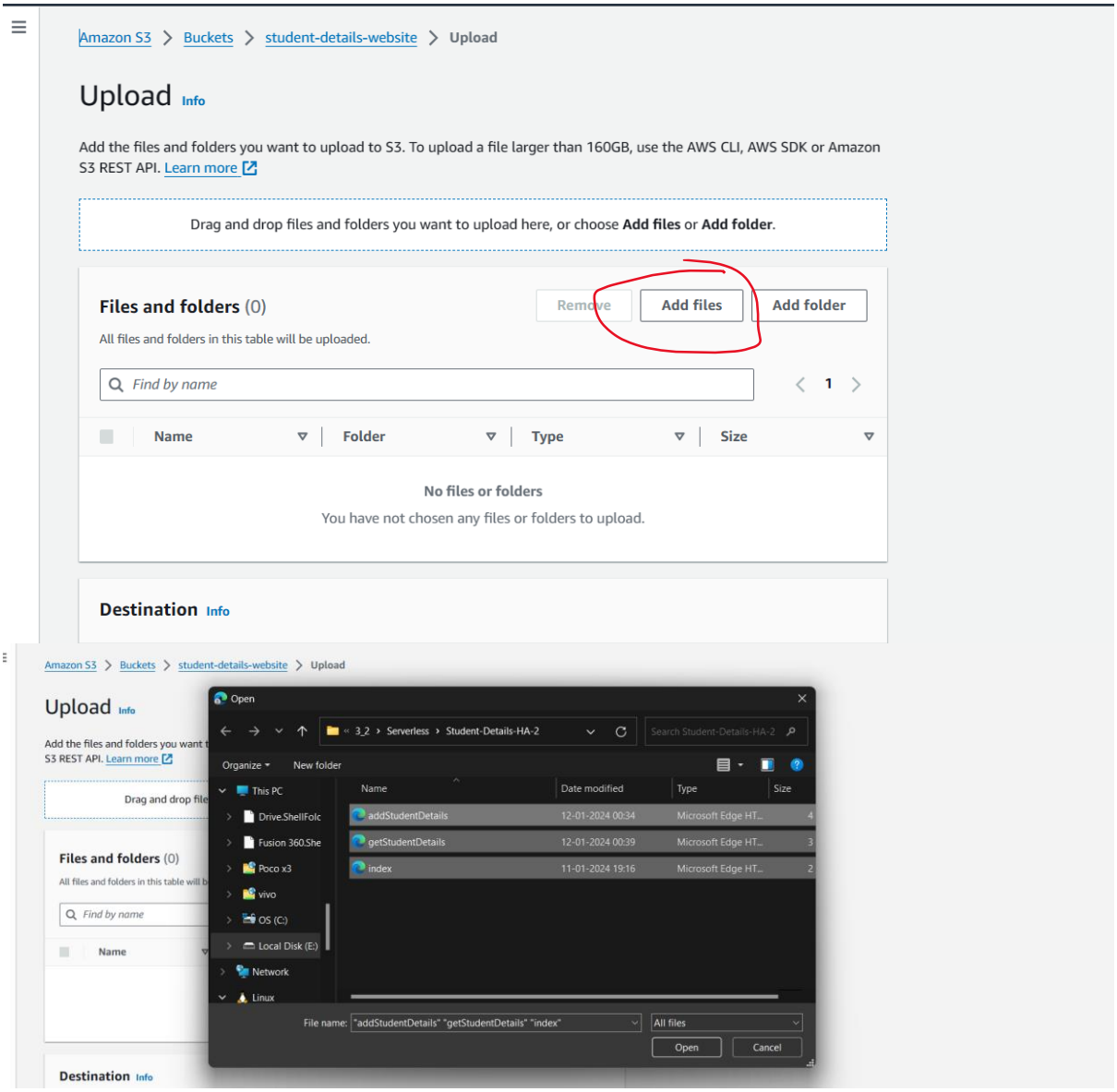


Click on create **bucket**.

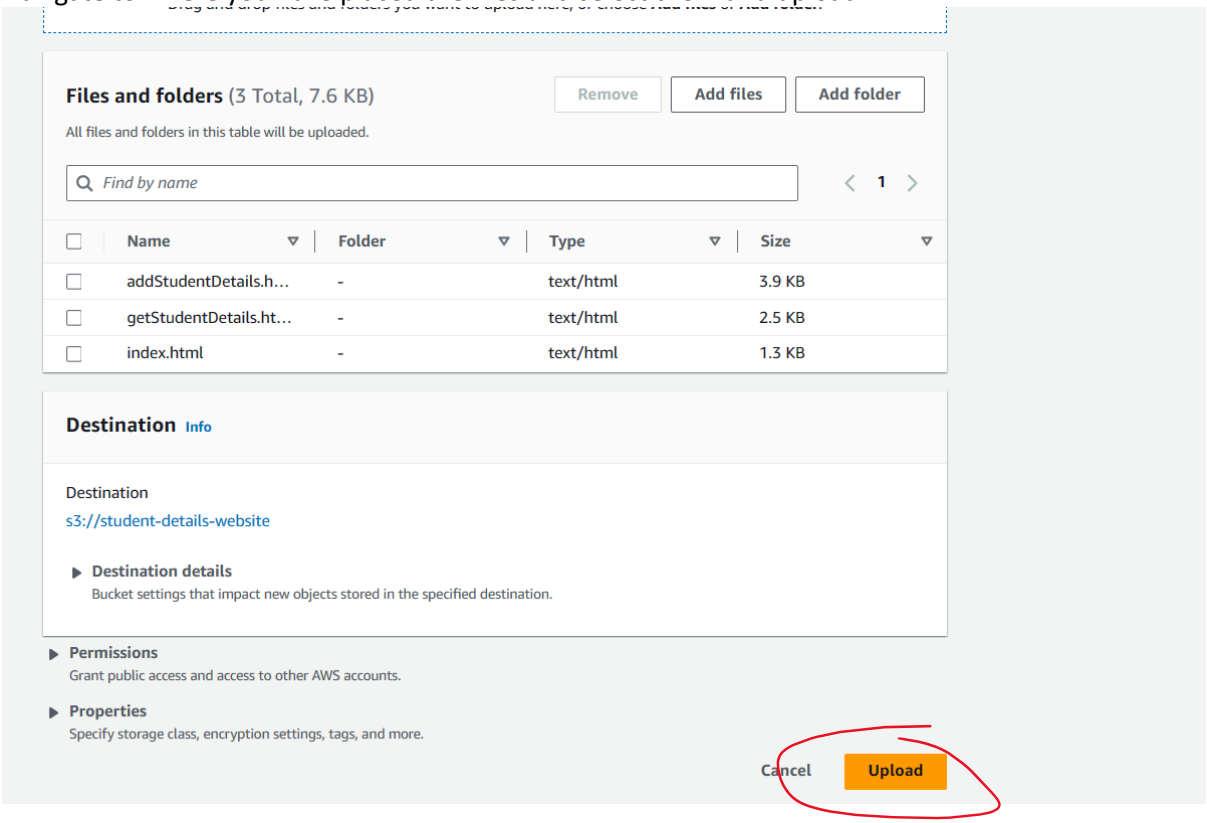
Goto the newly created bucket.

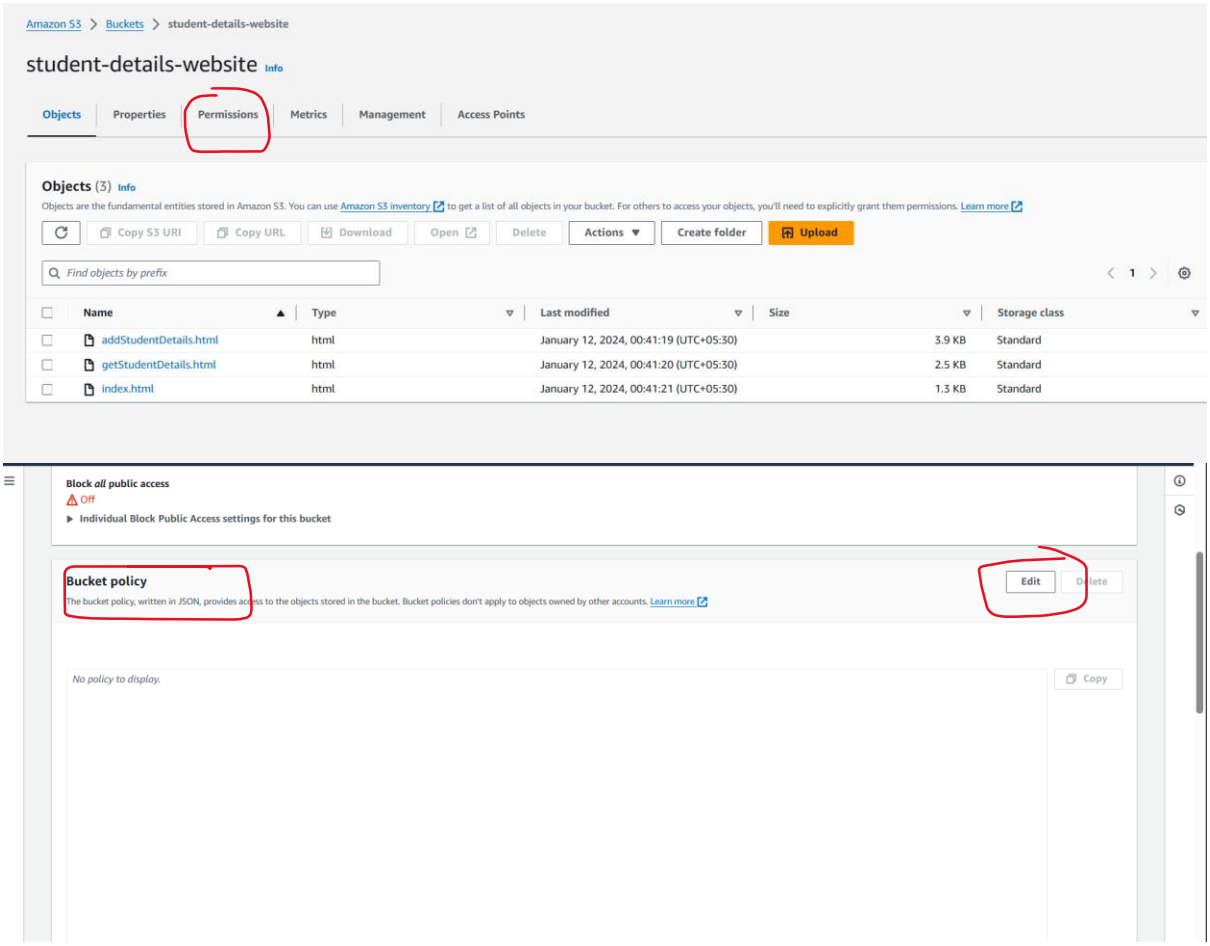


Click on upload.

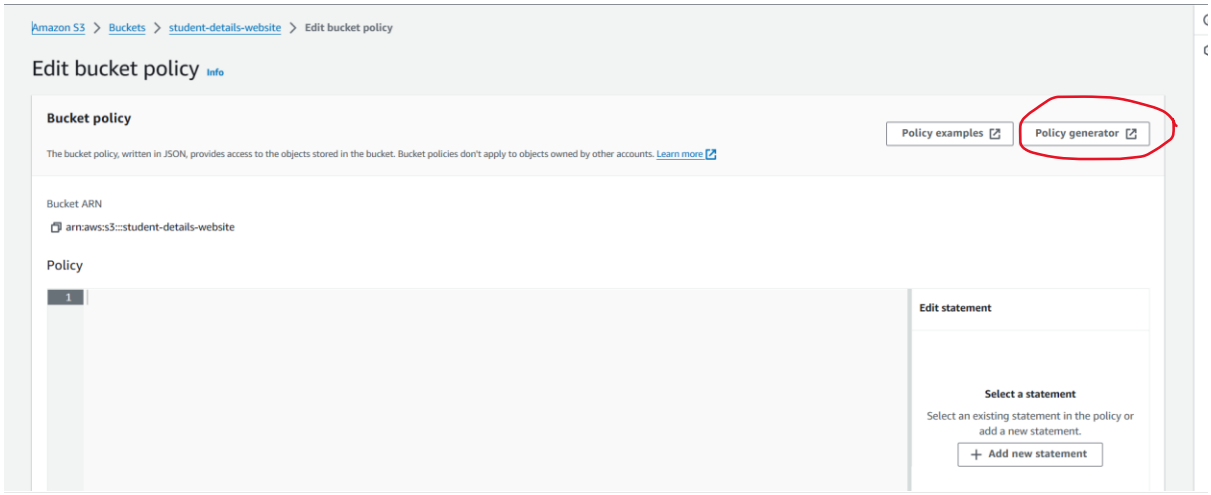

Amazon S3 > Buckets > student-details-website > Upload
Upload Info
Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more
Drag and drop files and folders you want to upload here, or choose Add files or Add folder.
Files and folders (0)
Remove
Add files
Add folder
All files and folders in this table will be uploaded.
Find by name
1
Name
Folder
Type
Size
No files or folders
You have not chosen any files or folders to upload.
Destination Info
Open
3_2 > Serverless > Student-Details-HA-2
Search Student-Details-HA-2
Organize
New folder
This PC
Drive.ShellFolc
Fusion 360.She
Poco x3
vivo
OS (C:)
Local Disk (E:)
Network
Linux
Name
Date modified
Type
Size
addStudentDetails
12-01-2024 00:34
Microsoft Edge HT...
getStudentDetails
12-01-2024 00:39
Microsoft Edge HT...
index
11-01-2024 19:16
Microsoft Edge HT...
File name: "addStudentDetails" "getStudentDetails" "index"
All files
Open
Cancel

Navigate to where you have placed the files and select them and upload.





Click on edit.



Step 1: Select Policy Type

A Policy is a container for permissions. The different types of policies you can create are an IAM Policy, an S3 Bucket Policy, an SNS Topic Policy, a VPC Endpoint Policy, and an SQS Queue Policy.

Select Type of Policy S3 Bucket Policy

Step 2: Add Statement(s)

A statement is the formal description of a single permission. See a description of elements that you can use in statements.

Effect Allow Deny

Principal *

Use a comma to separate multiple values.

AWS Service Amazon S3 All Services ('*')

Use multiple statements to add permissions for more than one service.

Actions 1 Action(s) Selected All Actions ('*')

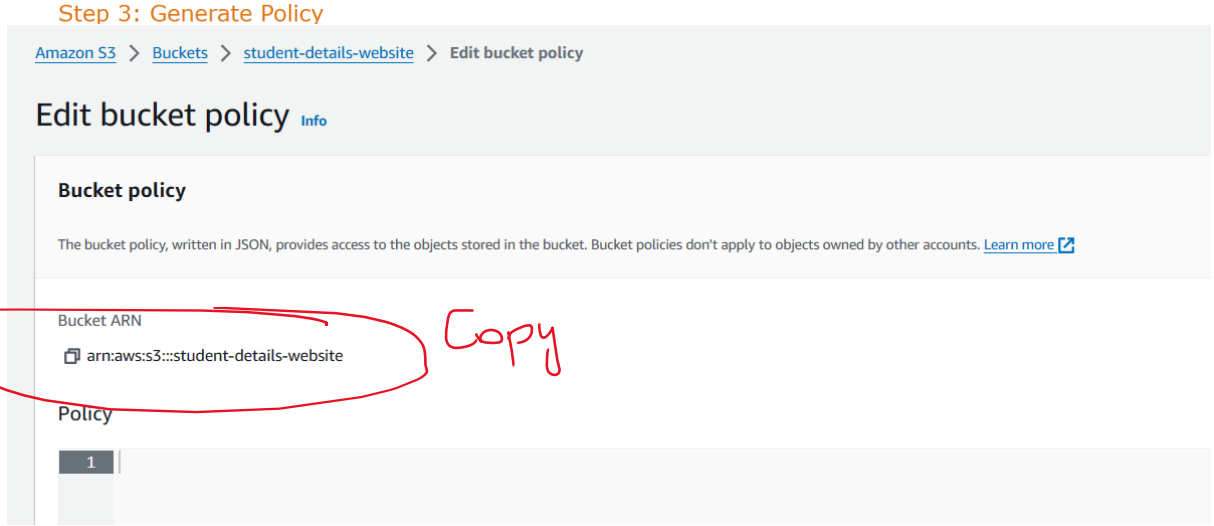
- GetMultiRegionAccessPointPolicyStatus
- GetMultiRegionAccessPointRoutes
- GetObject
- GetObjectAcl
- GetObjectAttributes
- GetObjectLegalHold
- GetObjectRetention
- GetObjectTagging

Amazon Resource Name (ARN)

{BucketName}/${KeyName}.

d. You must enter a valid ARN.

Step 3: Generate Policy



Use multiple statements to add permissions for more than one service.

Actions 1 Action(s) Selected ☐ All Actions ('*')

Amazon Resource Name (ARN) 3:::student-details-website/*

ARN should follow the following format: arn:aws:s3:::${BucketName}/${KeyName}.
Use a comma to separate multiple values.

Add Conditions (Optional)

Add /* at last of the ARN

A statement is the formal description of a single permission. See a description of elements that you can use in statements.

Effect ◉ Allow ○ Deny

Principal *

Use a comma to separate multiple values.

AWS Service Amazon S3 ☐ All Services ('*')

Use multiple statements to add permissions for more than one service.

Actions 1 Action(s) Selected ☐ All Actions ('*')

Amazon Resource Name (ARN) arn:aws:s3:::student-details-

ARN should follow the following format: arn:aws:s3:::${BucketName}/${KeyName}.
Use a comma to separate multiple values.

Add Conditions (Optional)

Add Statement

Step 3: Generate Policy

A *policy* is a document (written in the Access Policy Language) that acts as a container for one or more statements.

Add one or more statements above to generate a policy.

Use a comma to separate multiple values.

Add Conditions (Optional)

Add Statement

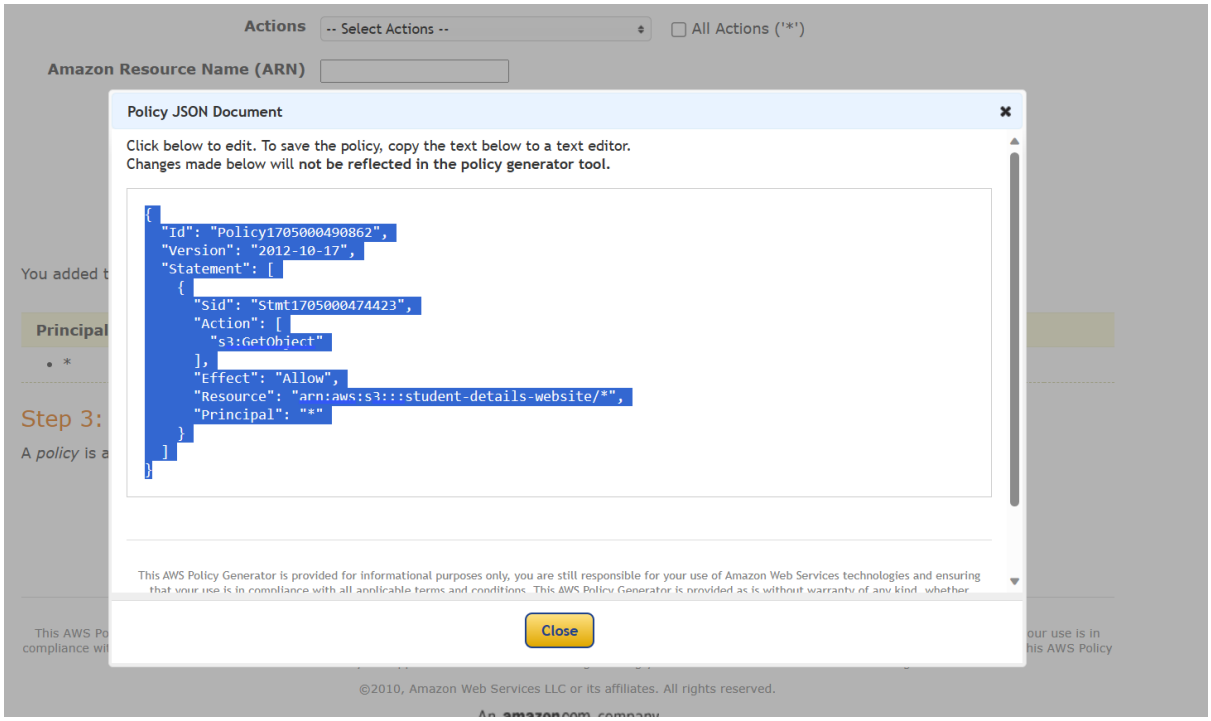
You added the following statements. Click the button below to Generate a policy.

Principal(s)	Effect	Action	Resource	Conditions
• *	Allow	• s3:GetObject	arn:aws:s3:::student-details-website/*	*None*

Step 3: Generate Policy

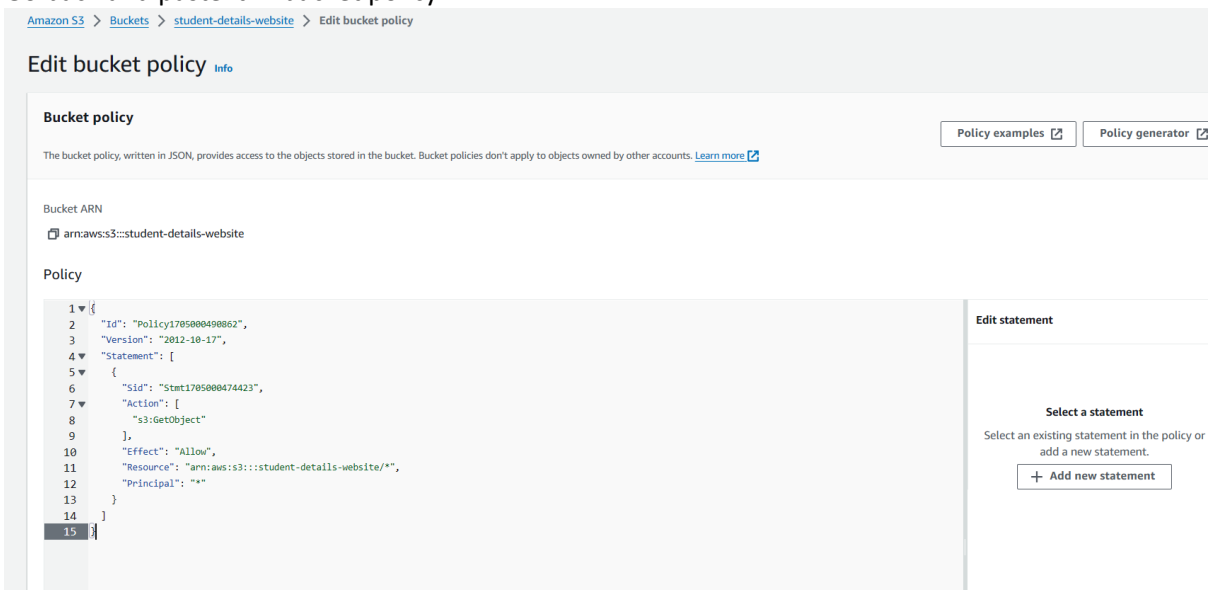
A *policy* is a document (written in the Access Policy Language) that acts as a container for one or more statements.

Generate Policy **Start Over**

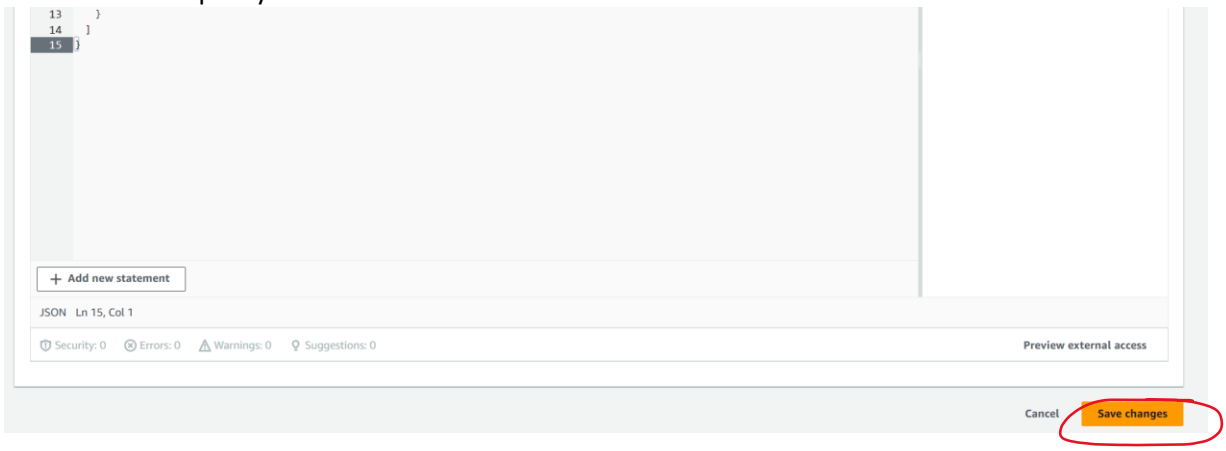


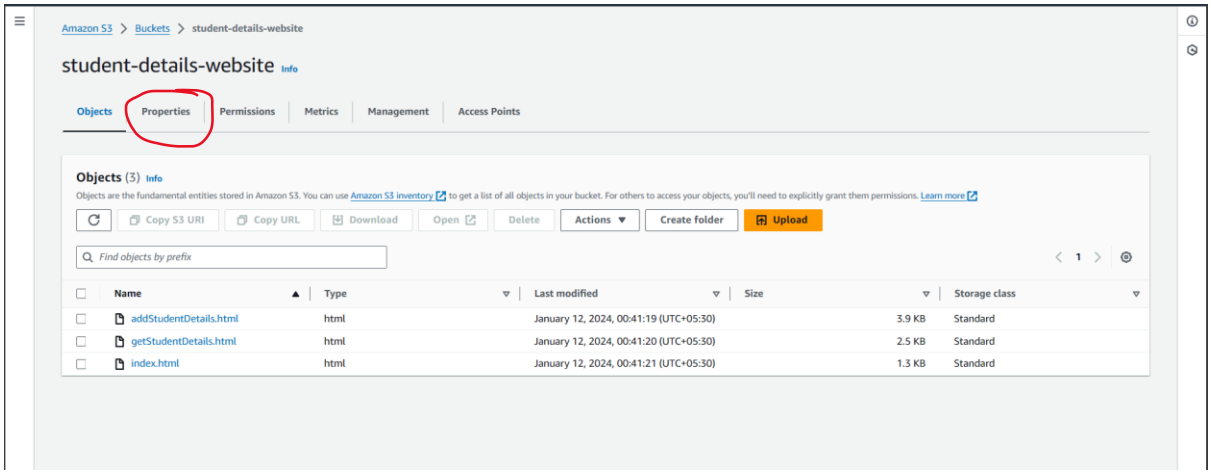
Copy the policy .

Go back and paste it in bucket policy.



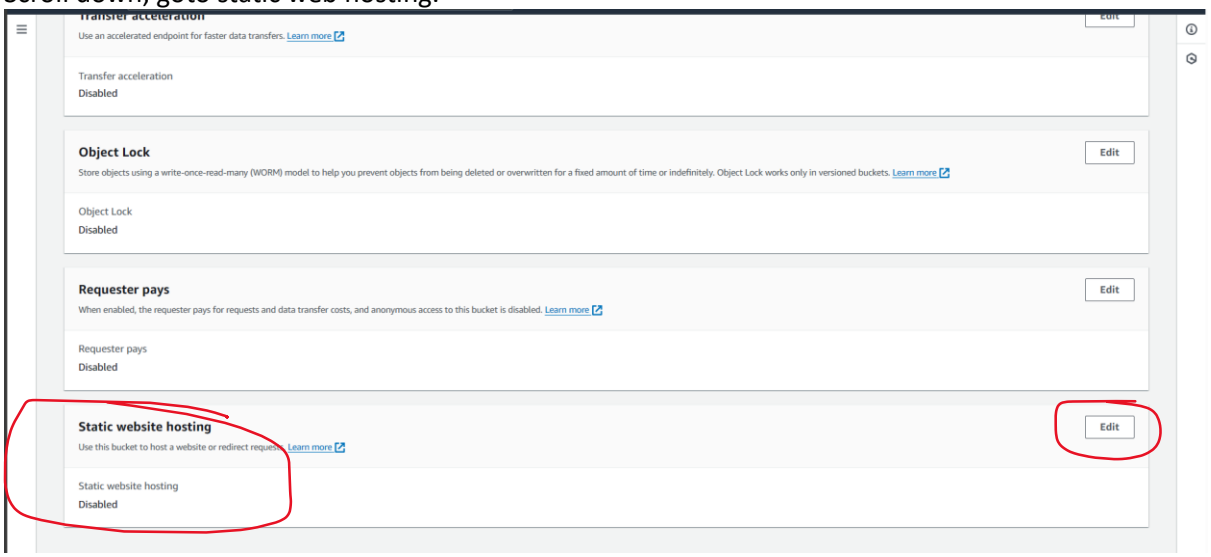
Save the bucket policy.

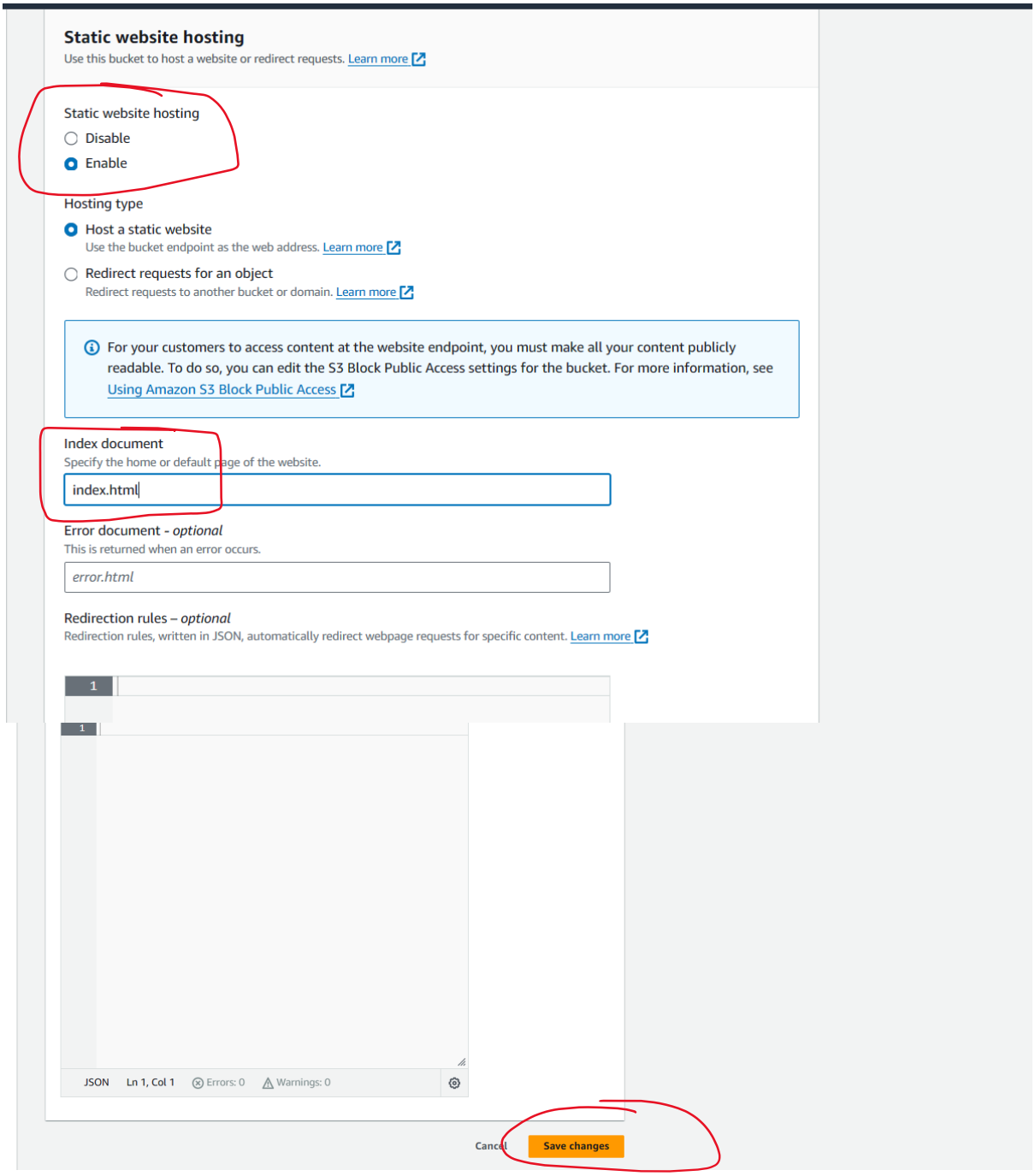


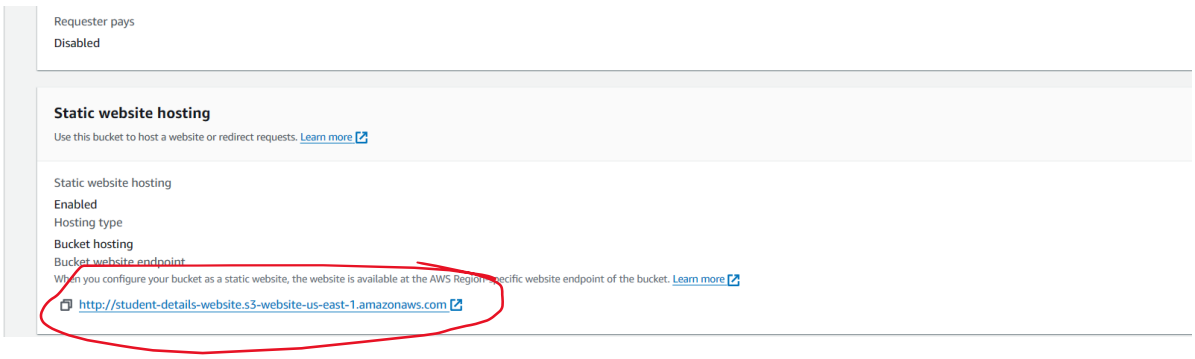


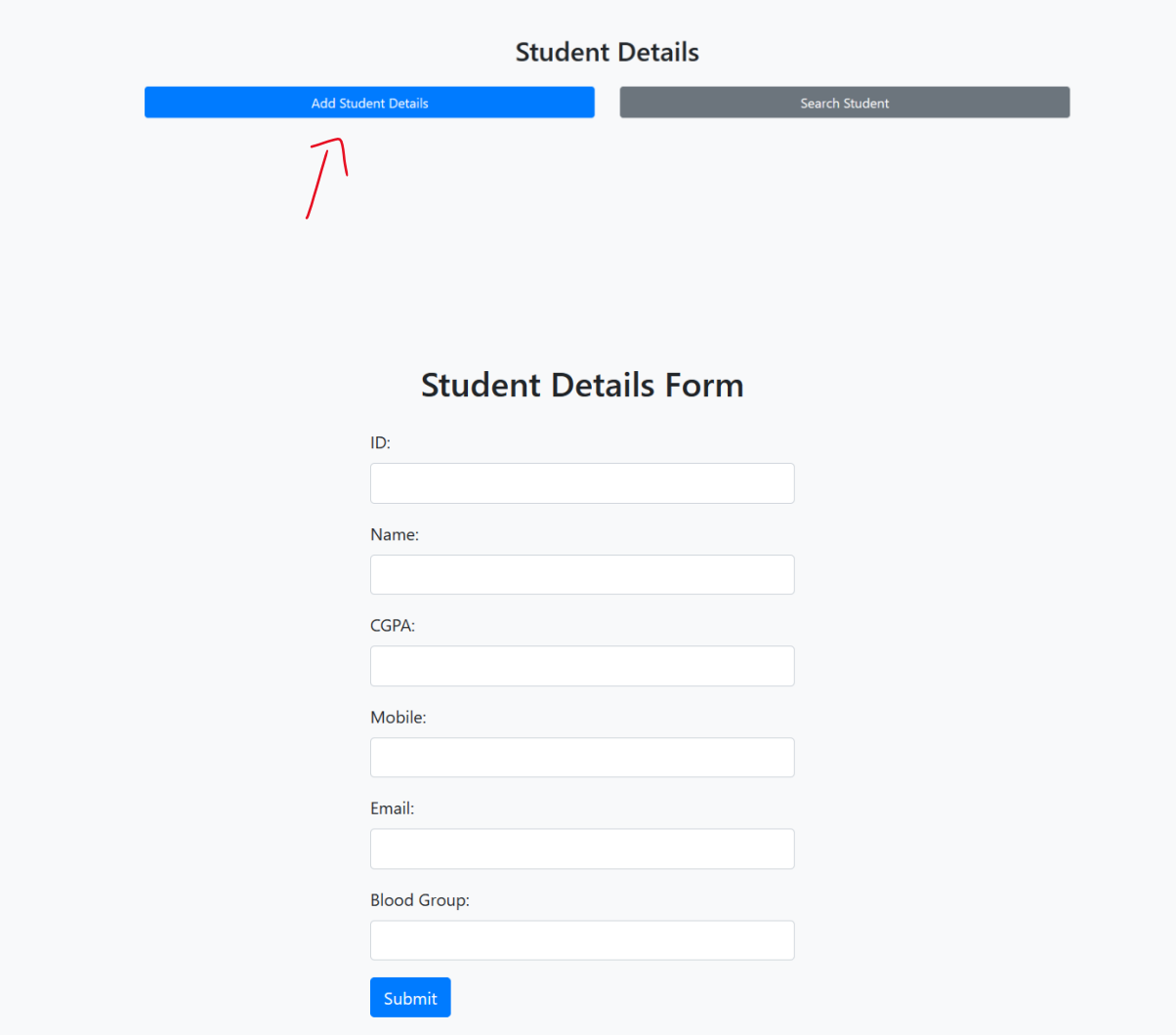
Goto properties.

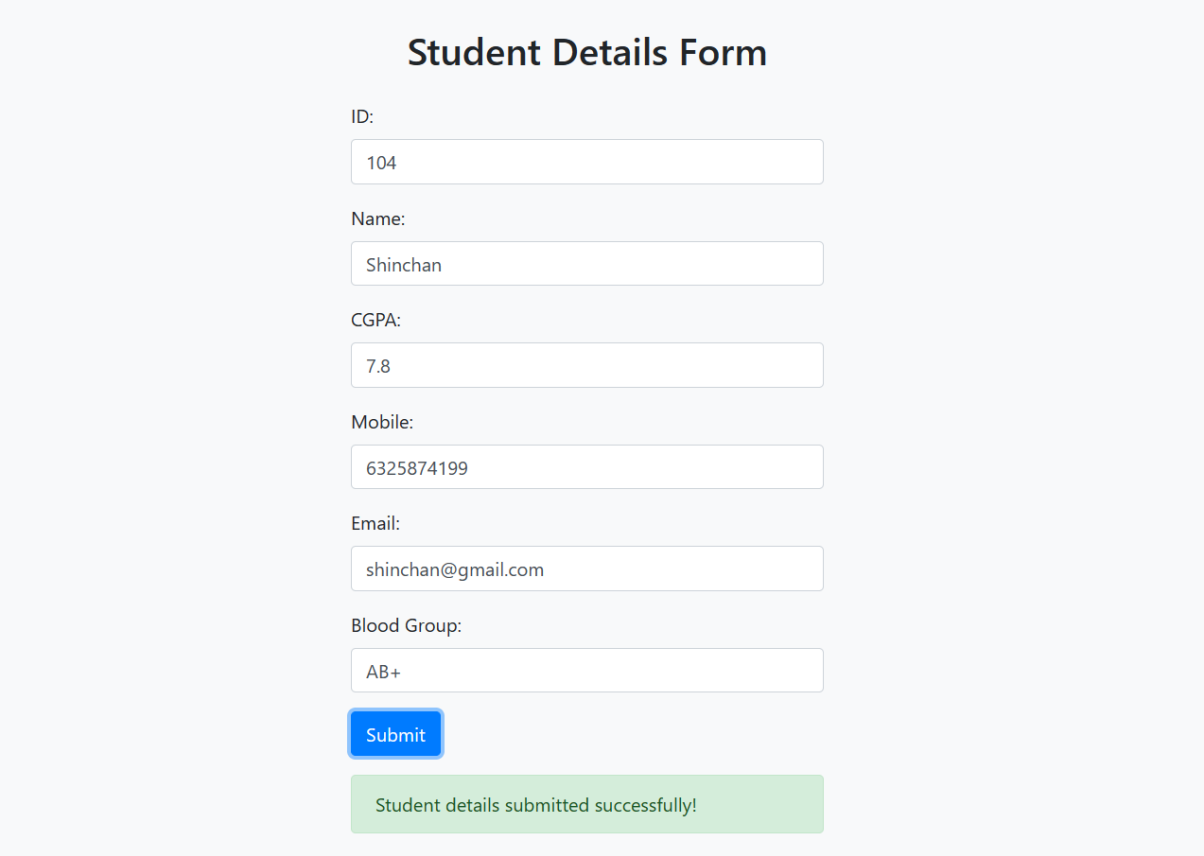
Scroll down, goto static web hosting.



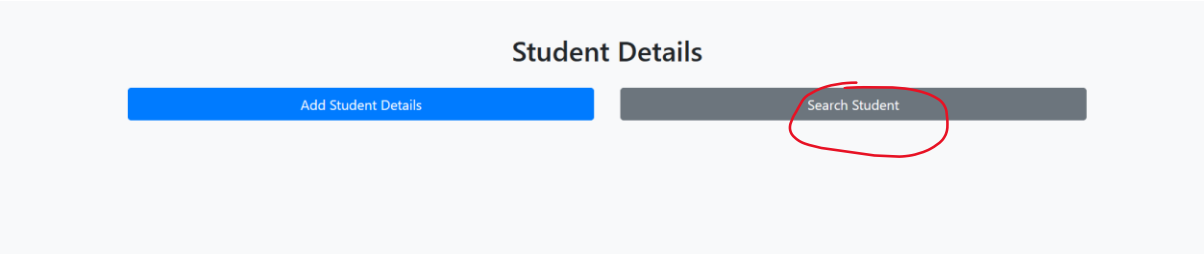

Static website hosting
Use this bucket to host a website or redirect requests. Learn more
Static website hosting
Disable
Enable
Hosting type
Host a static website
Use the bucket endpoint as the web address. Learn more
Redirect requests for an object
Redirect requests to another bucket or domain. Learn more
For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see Using Amazon S3 Block Public Access
Index document
Specify the home or default page of the website.
index.html
Error document - optional
This is returned when an error occurs.
error.html
Redirection rules – optional
Redirection rules, written in JSON, automatically redirect webpage requests for specific content. Learn more
1
1
JSON Ln 1, Col 1 Errors: 0 Warnings: 0
Cancel
Save changes

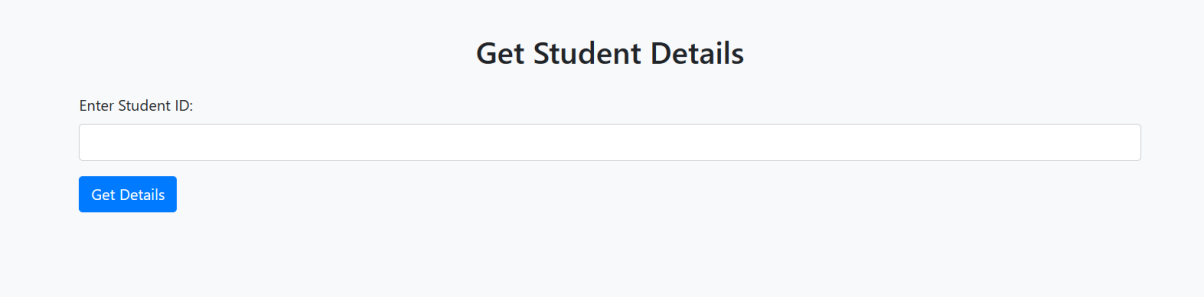

Requester pays
Disabled
Static website hosting
Use this bucket to host a website or redirect requests. Learn more
Static website hosting
Enabled
Hosting type
Bucket hosting
Bucket website endpoint
When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. Learn more
http://student-details-website.s3-website-us-east-1.amazonaws.com

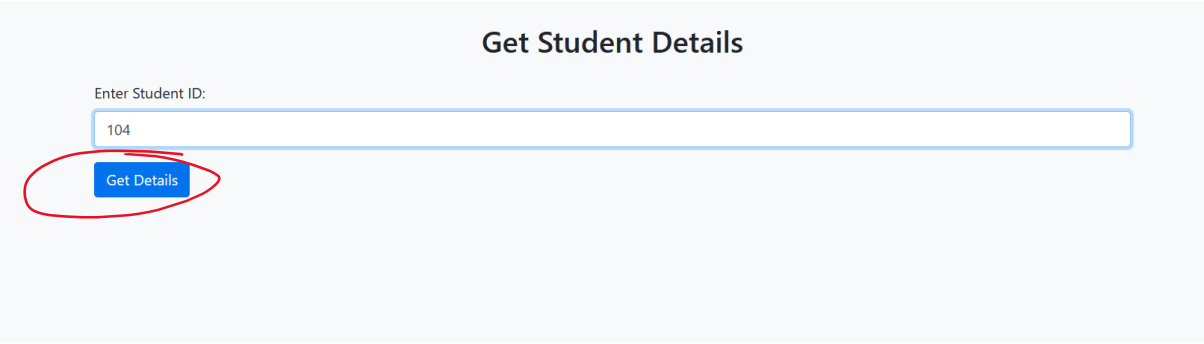

Student Details
Add Student Details
Search Student
Student Details Form
ID:
Name:
CGPA:
Mobile:
Email:
Blood Group:
Submit

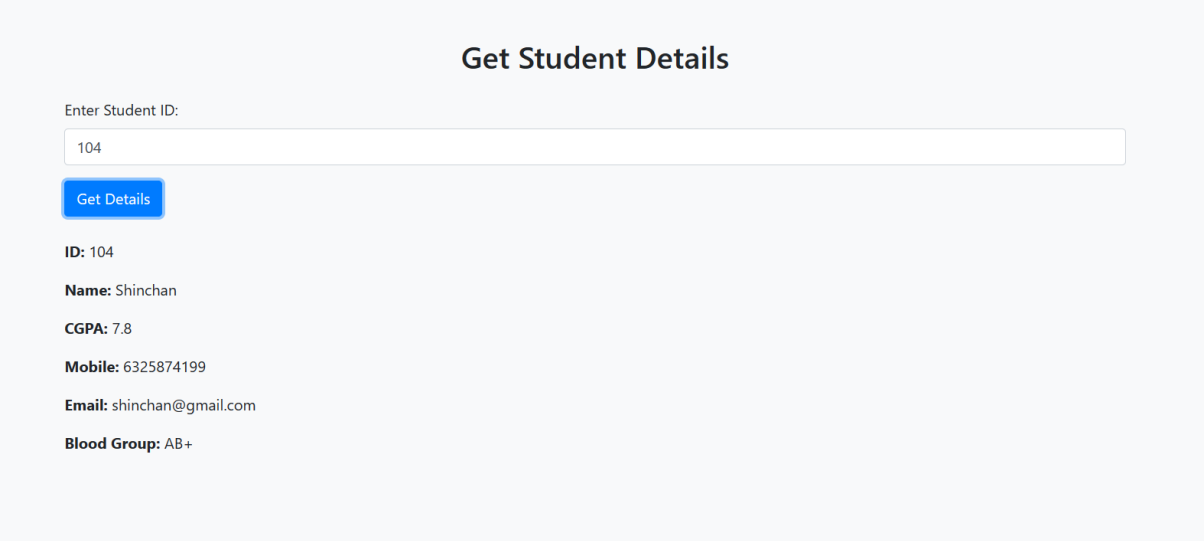


Go back and lets review search student.









Finally, you have successfully completed Home Assignment -2 😊.