

**DynamoDB Setup:**

→ Search DynamoDB in Aws Management Console and click on DynamoDB service.

The screenshot shows the AWS Management Console search results for 'dynamodb'. The 'DynamoDB' service card is highlighted with a red circle. Below it, the 'Dashboard' page for DynamoDB is displayed. The dashboard includes sections for Alarms (0), DAX clusters (0), and a prominent orange 'Create table' button. The right side of the screen shows a sidebar with application management options and a cost summary chart.

DynamoDB > Tables > Create table

## Create table

**Table details** Info

DynamoDB is a schemaless database that requires only a table name and a primary key when you create the table.

**Table name**  
 This will be used to identify your table.  
 Between 3 and 255 characters, containing only letters, numbers, underscores (\_), hyphens (-), and periods (.)  
 StudentDetails

**Partition key**  
 The partition key is part of the table's primary key. It is a hash value that is used to retrieve items from your table and allocate data across hosts for scalability and availability.  
 String ▾  
 1 to 255 characters and case sensitive.

**Sort key - optional**  
 You can use a sort key as the second part of a table's primary key. The sort key allows you to sort or search among all items sharing the same partition key.  
 String ▾  
 1 to 255 characters and case sensitive.

**Table Name : StudentDetails**

**Table details** Info

DynamoDB is a schemaless database that requires only a table name and a primary key when you create the table.

**Table name**  
 This will be used to identify your table.  
 Between 3 and 255 characters, containing only letters, numbers, underscores (\_), hyphens (-), and periods (.)  
 StudentDetails

**Partition key**  
 The partition key is part of the table's primary key. It is a hash value that is used to retrieve items from your table and allocate data across hosts for scalability and availability.  
 String ▾  
 id  
 1 to 255 characters and case sensitive.

**Sort key - optional**  
 You can use a sort key as the second part of a table's primary key. The sort key allows you to sort or search among all items sharing the same partition key.  
 String ▾  
 Enter the sort key name  
 1 to 255 characters and case sensitive.

**Table settings**

**Partition key : id**

**Scroll Down and click create table**

## Home Assignment-2 Cloud and Serverless Computing

Screenshot of the AWS DynamoDB 'Create Table' wizard.

**Table Configuration:**

Setting	Value	Editable after creation
Table class	DynamoDB Standard	Yes
Capacity mode	Provisioned	Yes
Provisioned read capacity	5 RCU	Yes
Provisioned write capacity	5 WCU	Yes
Auto scaling	On	Yes
Local secondary indexes	-	No
Global secondary indexes	-	Yes
Encryption key management	Owned by Amazon DynamoDB	Yes
Deletion protection	Off	Yes

**Tags:**  
Tags are pairs of keys and optional values, that you can assign to AWS resources. You can use tags to control access to your resources or track your AWS spending.

No tags are associated with the resource.

**Action Buttons:**

- Add new tag
- Create table

A red circle highlights the 'Create table' button.

Screenshot of the AWS DynamoDB 'Tables' page.

**Tables Overview:**

Name	Status	Partition key	Sort key	Indexes	Deletion protection	Read capacity mode	Write capacity mode	Total size	Table class
<a href="#">EmployeeDetails</a>	Active	Email (S)	-	0	Off	Provisioned (1)	Provisioned (1)	301 bytes	Standard
<a href="#">product</a>	Active	id (S)	-	0	Off	Provisioned (1)	Provisioned (1)	23 bytes	Standard
<a href="#">research</a>	Active	s3obj_id (S)	-	0	Off	Provisioned (1)	Provisioned (1)	9 kilobytes	Standard
<a href="#">student</a>	Active	id (S)	-	0	Off	Provisioned (1)	Provisioned (1)	627 bytes	Standard
<a href="#">StudentDetails</a>	Active	id (S)	-	0	Off	Provisioned (5)	Provisioned (5)	0 bytes	Standard
<a href="#">weather</a>	Active	sc (S)	-	0	Off	Provisioned (1)	Provisioned (1)	4.5 kilobytes	Standard

A red circle highlights the 'StudentDetails' table row.

**Click on StudentDetails Table**

## Home Assignment-2 Cloud and Serverless Computing

The screenshot shows the AWS DynamoDB console. On the left, the navigation pane includes 'Dashboard', 'Tables', 'Update settings', 'Explore items', 'PartiQL editor', 'Backups', 'Exports to S3', 'Imports from S3', 'Integrations', 'Reserved capacity', and 'Settings'. Under 'DAX', there are 'Clusters', 'Subnet groups', 'Parameter groups', and 'Events'. The main area shows 'Tables (6)' with 'StudentDetails' selected. The 'StudentDetails' table details are displayed, including 'General information' (Partition key: id (String), Sort key: -, Capacity mode: Provisioned, Table status: Active), 'Alarms' (No active alarms), and 'Point-in-time recovery (PITR)'. Below this is an 'Items summary' section with 'Item count', 'Table size', and 'Average item size'. At the top right of the main area, there is a red circle around the 'Explore table items' button.

Click on Explore table items.

The screenshot shows the 'Explore items' view for the StudentDetails table. It features a 'Scan or query items' section with 'Scan' selected. Below it are 'Filters' and a 'Run' button. A message indicates 'Completed. Read capacity units consumed: 0.5'. The 'Items returned (0)' section shows a table with one row: 'No items'. At the bottom right of this section, there is a red circle around the 'Create item' button.

Click on Create item.

The screenshot shows the 'Create item' form for the StudentDetails table. The 'Attributes' section lists 'id - Partition key' with a value of 'Empty value' and a type of 'String'. There is a 'Form' tab and a 'JSON view' tab at the top right. At the bottom right, there are 'Cancel' and 'Create item' buttons, with the 'Create item' button highlighted by a red circle.

Lets add an item.

**Id : 100**

Create item

You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. [Learn more](#)

Attributes		Type
Attribute name	Value	
id - Partition key	100	String

[Cancel](#) [Create item](#)

Create item

You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. [Learn more](#)

Attributes		Type
Attribute name	Value	
id - Partition key	100	String

Add new attribute ▾

- String
- Number
- Boolean
- Binary
- Null
- String set
- Number set
- Binary set
- List
- Map

### Click on String

Create item

You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. [Learn more](#)

Attributes		Type
Attribute name	Value	
id - Partition key	100	String
NewValue	Empty value	String

[Cancel](#) [Create item](#)

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 27°C Haze ENG IN 2006 11-01-2024

Replace NewValue with name

name : K Ram

DynamoDB > Explore items: StudentDetails > Create item

**Create item**

You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. [Learn more](#)

**Attributes**

Attribute name	Value	Type
id - Partition key	100	String
name	K Ram	String

Add new attribute ▾

Cancel **Create item**

**Repeat the process for the below attributes**

**email : [ram@gmail.com](mailto:ram@gmail.com)**

**mobile : 7412589633**

**cgpa : 9.3**

**bloodgroup : A+**

**Create item**

You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. [Learn more](#)

**Attributes**

Attribute name	Value	Type
id - Partition key	100	String
name	K Ram	String
email	ram@gmail.com	String
mobile	7412589633	String
cgpa	9.3	String
bloodgroup	A+	String

Add new attribute ▾

Cancel **Create item**

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

**After filling the attributes and values click on create item.**

## Home Assignment-2 Cloud and Serverless Computing

DynamoDB > Explore Items > StudentDetails

Tables (6)

Any tag key

Any tag value

Find tables by table name

EmployeeDetails

product

research

student

StudentDetails

weather

Scan or query items

Scan Query

Select a table or index

Table - StudentDetails

Select attribute projection

All attributes

Filters

Run Reset

Completed. Read capacity units consumed: 0.5

Items returned (1)

	id (String)	bloodgroup (String)	cgpa (Number)	email (String)	mobile (Number)	name (String)
100	A+	9.5	ram@gmail.com	7412589633	K Ram	

## IAM Roles

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

https://us-east-1.console.aws.amazon.com/console/home?region=us-east-1

aws Services Q IAM

Search results for 'IAM'

Services (11)

Features (21)

Resources New

Documentation (49,042)

Knowledge Articles (548)

Marketplace (725)

Blogs (1,730)

Events (12)

Tutorials (2)

**IAM** Manage access to AWS resources

Top features

Groups Users Roles Policies Access Analyzer

IAM Identity Center Manage workforce user access to multiple AWS accounts and cloud applications

Resource Access Manager Share AWS resources with other accounts or AWS Organizations

AWS App Mesh Easily monitor and control microservices

**IAM Dashboard**

**Security recommendations** 1

- ⚠ Add MFA for root user
- ✓ Root user has no active access keys

**IAM resources**

User groups	Users	Roles	Policies	Identity providers
0	2	31	13	0

**What's new** [View all](#)

click on Roles

**Roles (31) Info**

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

Role name	Trusted entities	Last activity
api_poweruser	AWS Service: apigateway	-

**Select trusted entity** [Info](#)

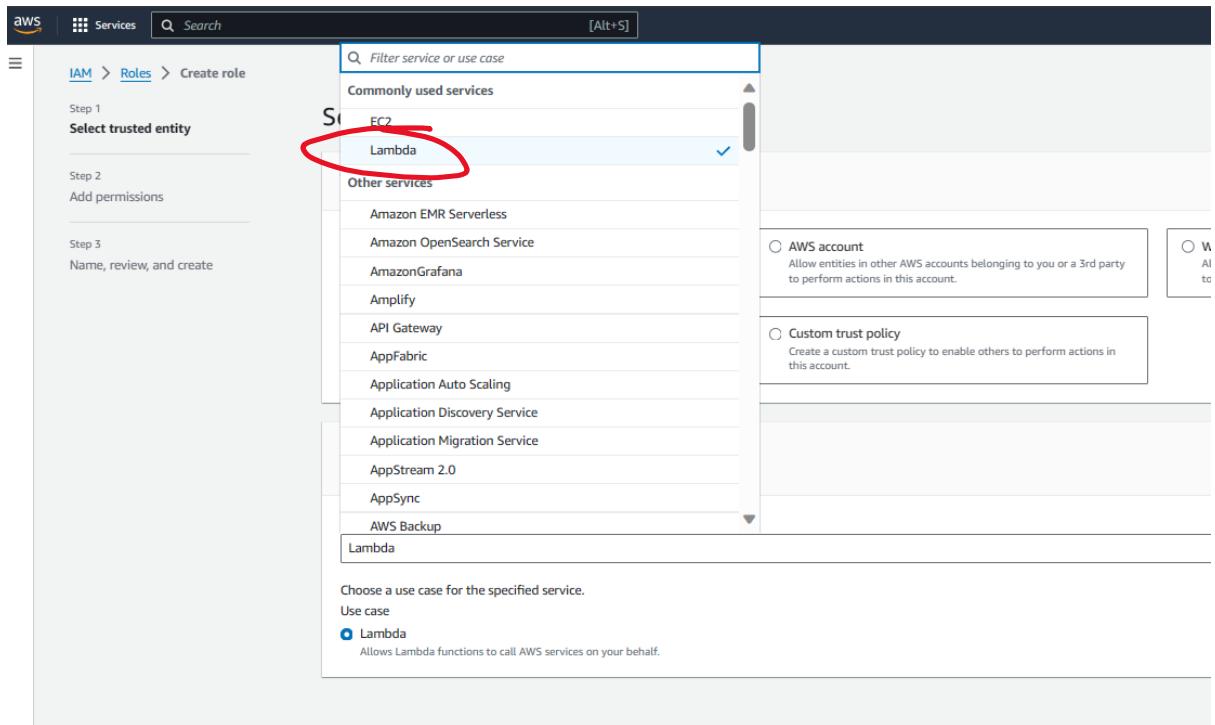
**Trusted entity type**

- 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 Allows 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.

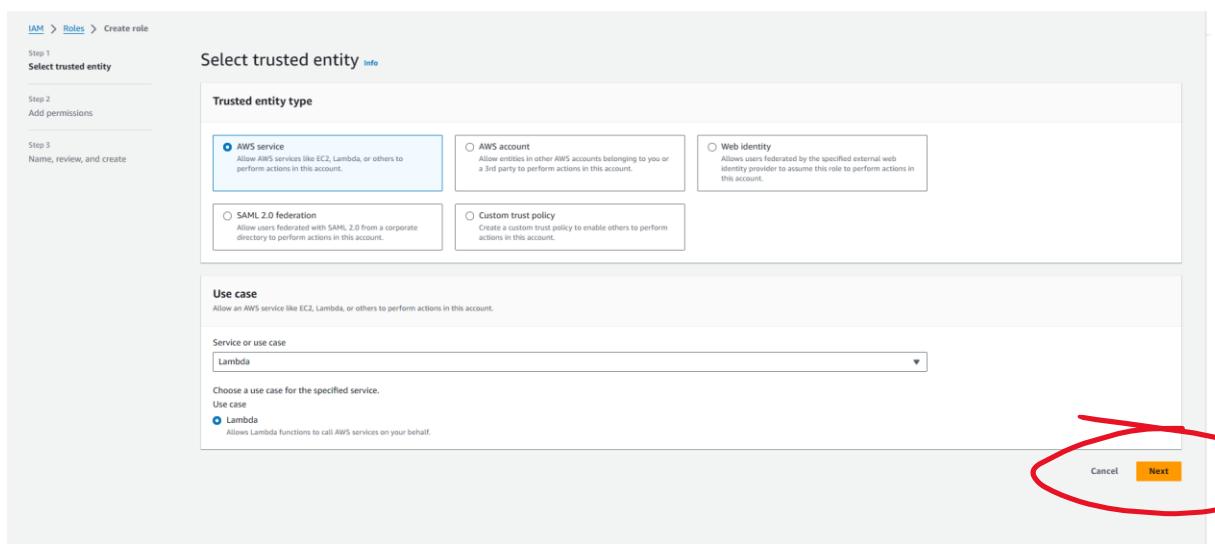
**Use case**  
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

**Service or use case**

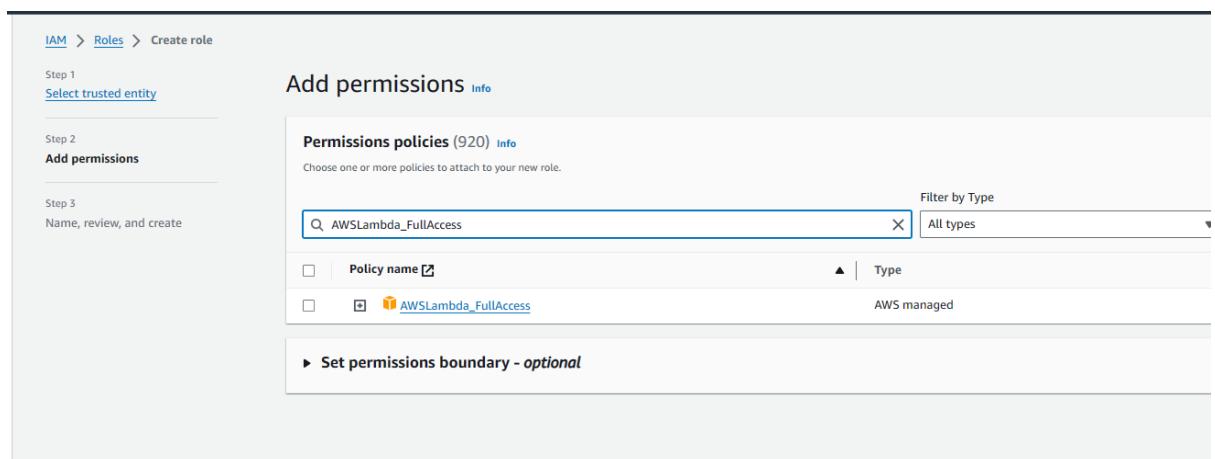
## Home Assignment-2 Cloud and Serverless Computing



The screenshot shows the AWS IAM 'Create role' wizard. The current step is 'Step 1: Select trusted entity'. A dropdown menu titled 'Filter service or use case' is open, showing a list of services. The 'Lambda' service is listed under 'Commonly used services' and is circled in red. Other services listed include EC2, Amazon EMR Serverless, Amazon OpenSearch Service, Amazon Grafana, Amplify, API Gateway, AppFabric, Application Auto Scaling, Application Discovery Service, Application Migration Service, AppStream 2.0, AppSync, and AWS Backup. Below the dropdown, a section titled 'Choose a use case for the specified service.' shows a single option: 'Use case' with 'Lambda' selected, which allows Lambda functions to call AWS services on behalf of the user.



This screenshot shows the 'Select trusted entity' step of the 'Create role' wizard. It displays the 'Trusted entity type' section with four options: 'AWS service' (selected), 'AWS account', 'Web identity', 'SAML 2.0 federation', and 'Custom trust policy'. Below this, the 'Use case' section is shown, with 'Lambda' selected. At the bottom right, there are 'Cancel' and 'Next Step' buttons, with the 'Next Step' button circled in red.



This screenshot shows the 'Add permissions' step of the 'Create role' wizard. It displays the 'Permissions policies' section with a search bar containing 'AWSLambda\_FullAccess'. A single policy, 'AWSLambda\_FullAccess', is listed and selected, indicated by a red circle. Below the policies, there is an optional section for setting a 'permissions boundary'.

Search for [AWSLambda\\_FullAccess](#). Click the check box

Screenshot of the AWS IAM 'Add permissions' step. The search bar at the top contains 'AWSLambda\_FullAccess'. A red circle labeled '1.' highlights the search bar. A red circle labeled '2.' highlights the checkbox next to 'Policy name' for the selected policy.

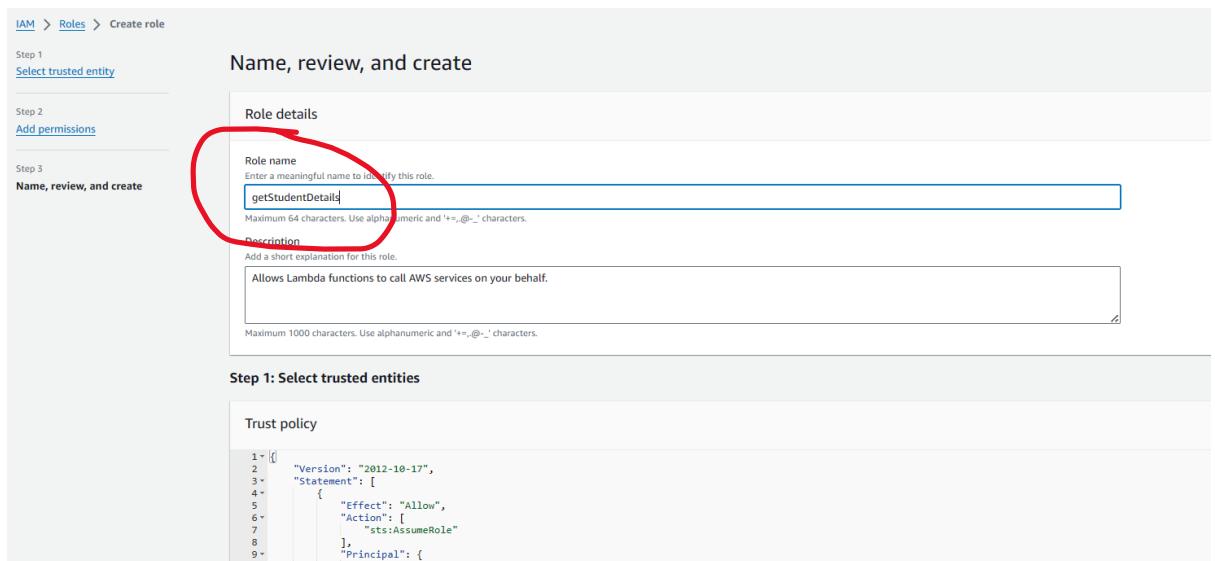
Screenshot of the AWS IAM 'Add permissions' step. The search bar at the top contains 'AmazonDynamoDBReadOnlyAccess'. A red arrow points to the search bar. A red circle labeled '1.' highlights the search bar. A red circle labeled '2.' highlights the checkbox next to 'Policy name' for the selected policy.

Search for [AmazonDynamoDBReadOnlyAccess](#). Click the check box

Screenshot of the AWS IAM 'Add permissions' step. The search bar at the top contains 'AmazonDynamoDBReadOnlyAccess'. A red circle labeled '1.' highlights the search bar. A red circle labeled '2.' highlights the checkbox next to 'Policy name' for the selected policy. A large red circle highlights the 'Next' button at the bottom right.

Click on next.

## Home Assignment-2 Cloud and Serverless Computing

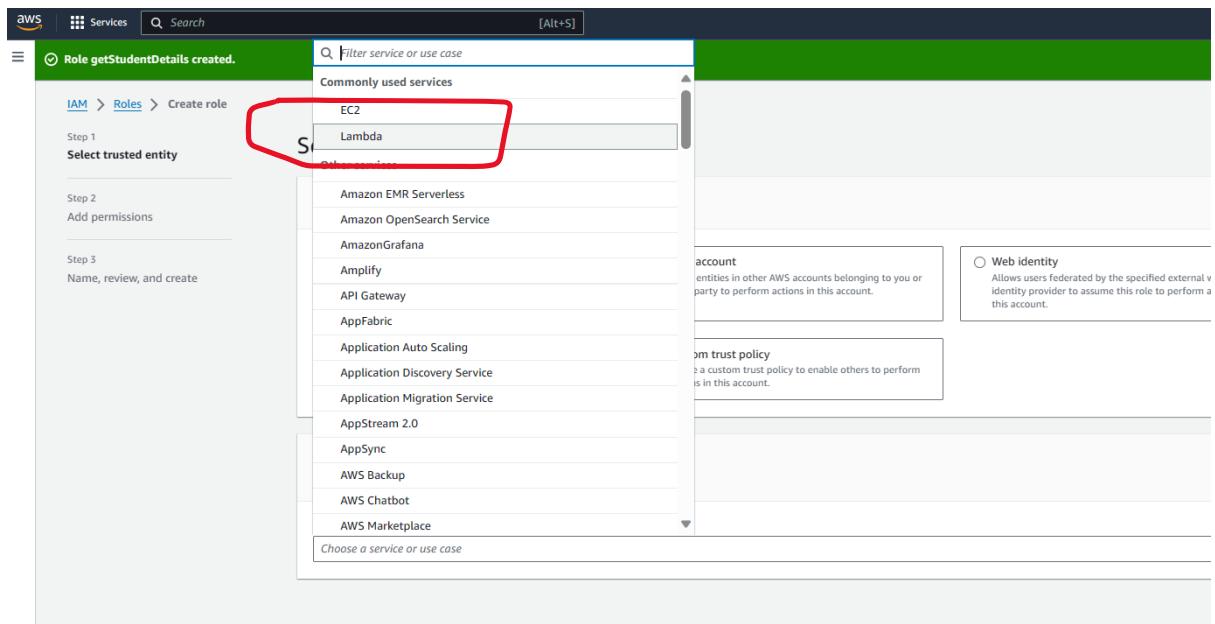


The screenshot shows the 'Name, review, and create' step of the IAM 'Create role' wizard. The 'Role name' field is highlighted with a red circle. The 'Description' field contains the text: 'Allows Lambda functions to call AWS services on your behalf.'

**Role Name : getStudentDetails**

Scroll down and click create role.

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



The screenshot shows the 'Select trusted entity' step of the IAM 'Create role' wizard. The 'Lambda' service is selected in the dropdown menu and is circled in red. Other services listed include EC2, Amazon EMR Serverless, Amazon OpenSearch Service, Amazon Grafana, Amplify, API Gateway, AppFabric, Application Auto Scaling, Application Discovery Service, Application Migration Service, AppStream 2.0, AppSync, AWS Backup, AWS Chatbot, and AWS Marketplace.

**Click next**

## Home Assignment-2 Cloud and Serverless Computing

The screenshot shows the 'Add permissions' step in the AWS IAM console. A search bar at the top contains 'AWSLambda\_FullAccess'. Below it, a table lists one result: 'AWSLambda\_FullAccess' (AWS managed). A link to 'Set permissions boundary - optional' is visible.

### Search for [AWSLambda\\_FullAccess](#)

The screenshot shows the 'Add permissions' step in the AWS IAM console. A search bar at the top contains 'AmazonS3FullAccess'. Below it, a table lists one result: 'AmazonS3FullAccess' (AWS managed). A link to 'Set permissions boundary - optional' is visible.

### Search for [AmazonS3FullAccess](#)

Click on next.

The screenshot shows the 'Name, review, and create' step in the AWS IAM console. It includes fields for 'Role details' (name: 'addStudentDetails', description: 'Allows Lambda functions to call AWS services on your behalf.'), 'Step 1: Select trusted entities' (trust policy JSON), and 'Step 2: Add permissions'.

```
1 - [ { "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "sts:AssumeRole", "Principal": { "Service": "lambda.amazonaws.com" } } ] }
```

**Role name : addStudentDetails**

Scroll down and Click on create role.

Go to Roles page and search for addStudentDetails role.

## Home Assignment-2 Cloud and Serverless Computing

The screenshot shows the AWS IAM Roles page. At the top, there is a search bar with the placeholder 'Q add' and a dropdown menu showing '1 match'. Below the search bar, a table lists roles. The first row, which contains the role name 'addStudentDetails', is highlighted with a red oval. The table also includes columns for 'Trusted entities' and 'Last activity'. A note at the bottom of the table states 'AWS Service: lambda'. Below the table, there is a section titled 'Roles Anywhere' with three options: 'Access AWS from your non AWS workloads', 'X.509 Standard', and 'Temporary credentials'.

**Click on the role addStudentDetails.**

**Scroll down and click on Add permissions.**

**click on Create Inline policy.**

The screenshot shows the 'Permissions' tab of the 'addStudentDetails' role's configuration page. It displays basic information like creation date (January 11, 2024) and ARN (arn:aws:iam::439554286652:role/addStudentDetails). The 'Permissions' tab is active, showing two managed policies: 'AmazonS3FullAccess' and 'AWSLambda\_FullAccess'. On the right side, there is a 'Permissions' panel with a 'Create inline policy' button highlighted with a red oval. Other buttons in the panel include 'Add permissions', 'Attach policies', and 'Simulate'.

## Home Assignment-2 Cloud and Serverless Computing

The screenshots illustrate the process of searching for the DynamoDB service in the AWS IAM Policy Editor.

**Screenshot 1:** The search bar is empty, showing the list of "Commonly used services".

**Screenshot 2:** The search bar contains the partial text "dyna", and the results list shows "DynamoDB" highlighted with a red box.

**Screenshot 3:** The search bar now contains the full text "DynamoDB", and the results list shows "DynamoDB" expanded, displaying its actions and resources.

**Search for putItem**

## Home Assignment-2 Cloud and Serverless Computing

Specify permissions [Info](#)  
Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.

Policy editor

DynamoDB [Allow](#) 1 Action

Specify what actions can be performed on specific resources in DynamoDB.

Actions allowed

Search: putitem

Effect: Allow

1.  PutItem

2. All

Resources

The 'all wildcard' \* may be overly permissive for the selected actions. Allowing specific ARNs for these service resources can improve security.

Request conditions - optional

+ Add more permissions

Scroll down and click on next.

Review and create [Info](#)  
Review the permissions, specify details, and tags.

Policy details

Policy name: putItemDynamodb

Permissions defined in this policy [Info](#)  
Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it.

Allow (1 of 403 services)

Service	Access level	Resource	Request condition
DynamoDB	Limited: Write	All resources	None

Create policy

Policy name : putItemDynamodb

Click on Create policy.

Identity and Access Management (IAM)

addStudentDetails [Info](#)  
Allows Lambda functions to call AWS services on your behalf.

Summary

Creation date	ARN
January 11, 2024, 20:45 (UTC+05:30)	arn:aws:iam::439554286652:role/addStudentDetails

Last activity: -

Maximum session duration: 1 hour

Permissions

Permissions policies (3) [Info](#)  
You can attach up to 10 managed policies.

Policy name	Type	Attached entities
AmazonS3FullAccess	AWS managed	4
AWSLambda_FullAccess	AWS managed	5
putItemDynamodb	Customer inline	0

Permissions boundary (not set)

## Home Assignment-2 Cloud and Serverless Computing

### Lambda

The screenshot shows the AWS Services search results page with 'lambda' typed into the search bar. The 'Lambda' service card is highlighted with a red circle, showing its icon, name, and description: 'Run code without thinking about servers'. Other services listed include CodeBuild, AWS Signer, and Amazon Inspector.

The screenshot shows the AWS Lambda Functions list page. The 'Create function' button is circled in red. The table lists five existing functions: getSmartWeather, s3\_lambda, s3\_texttract, and two unnamed entries.

Function name	Description	Package type	Runtime	Last modified
getSmartWeather	-	Zip	Node.js 16.x	14 days ago
s3_lambda	-	Zip	Python 3.9	8 days ago
s3_texttract	-	Zip	Python 3.9	8 days ago
get_weather	-	Zip	Node.js 14.x	11 days ago

The screenshot shows the 'Create function' wizard. The 'Author from scratch' option is selected and highlighted with a red circle. The 'Basic information' section shows the function name 'getStudentDetails' and runtime 'Node.js 16.x', both also circled in red.

**Function name : getStudentDetails**

**Runtime : Node.js 16.x**

**Scroll down**

## Home Assignment-2 Cloud and Serverless Computing

**Runtime Info**  
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Node.js 16.x

**Architecture Info**  
Choose the instruction set architecture you want for your function code.

x86\_64  
 arm64

**Permissions Info**  
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▶ Change default execution role

Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Node.js 16.x

**Architecture Info**  
Choose the instruction set architecture you want for your function code.

x86\_64  
 arm64

**Permissions Info**  
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▼ Change default execution role

**Execution role**  
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console [IAM console](#).

Create a new role with basic Lambda permissions  
 Use an existing role  
 Create a new role from AWS policy templates

**Existing role**  
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

getStudentDetails

▶ Advanced settings

Type **getStudentDetails** to search existing roles.

Use an existing role  
 Create a new role from AWS policy templates

getStudentDetails

getStudentDetails role must have permission to upload logs to Amazon CloudWatch Logs.

▶ Advanced settings

**Change default execution role**

**Execution role**  
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console [IAM console](#).

Create a new role with basic Lambda permissions  
 Use an existing role  
 Create a new role from AWS policy templates

**Existing role**  
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

getStudentDetails   
View the getStudentDetails role [on the IAM console](#).

▶ Advanced settings

```

        index.js
        Environment Var x
        1 exports.handler = async (event) => {
        2     // TODO implement
        3     const response = {
        4         statusCode: 200,
        5         body: JSON.stringify('Hello from Lambda!'),
        6     };
        7     return response;
        8 };
    
```

Replace the code with the below one :

```

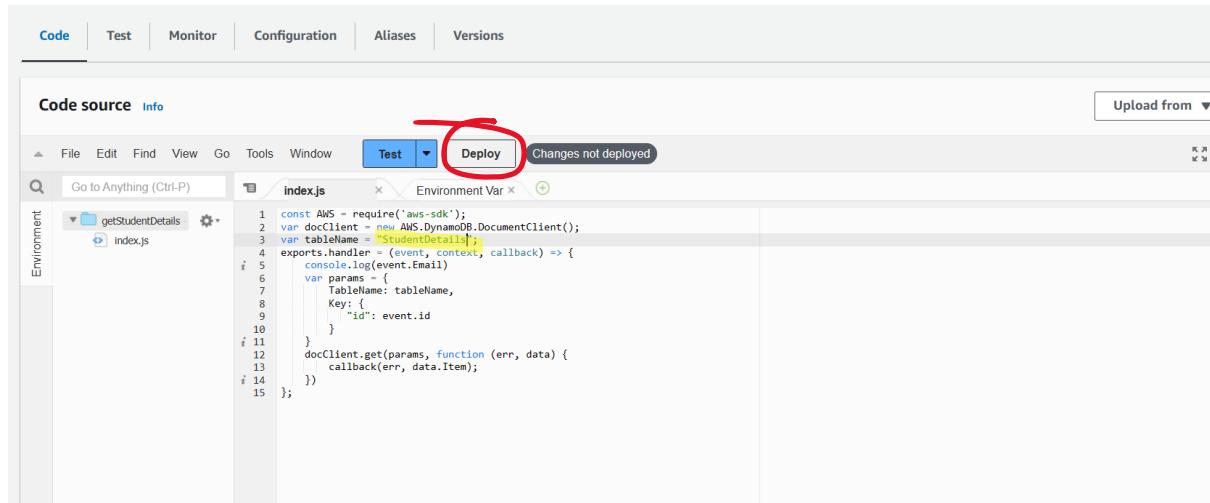
const AWS = require('aws-sdk');
var docClient = new AWS.DynamoDB.DocumentClient();
var tableName = "student";
exports.handler = (event, context, callback) => {
    console.log(event.Email)
    var params = {
        TableName: tableName,
        Key: {
            "id": event.id
        }
    }
    docClient.get(params,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

```

        index.js
        Environment Var x
        1 const AWS = require('aws-sdk');
        2 var docClient = new AWS.DynamoDB.DocumentClient();
        3 var tableName = "student";
        4 exports.handler = (event, context, callback) => {
        5     console.log(event.Email)
        6     var params = {
        7         TableName: tableName,
        8         Key: {
        9             "id": event.id
        10        }
        11    }
        12    docClient.get(params, function (err, data) {
        13        callback(err, data.Item);
        14    })
        15 };
    
```

Replace the tableName with StudentDetails

## Home Assignment-2 Cloud and Serverless Computing

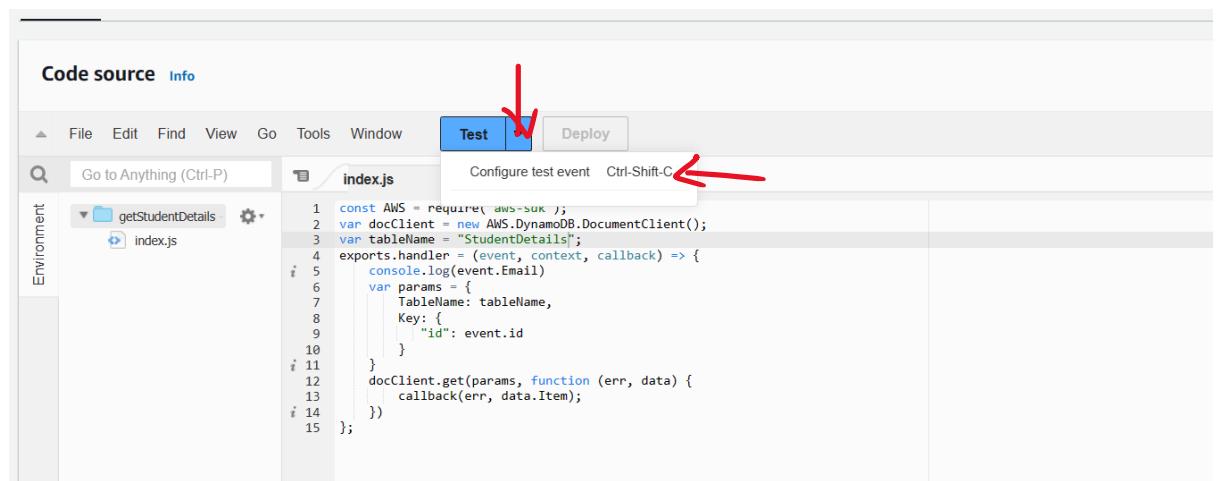


The screenshot shows the AWS Lambda code editor interface. At the top, there are tabs for 'Code', 'Test', 'Monitor', 'Configuration', 'Aliases', and 'Versions'. Below the tabs, the 'Code source' tab is selected. In the center, there is a code editor window with a file named 'index.js'. The code is as follows:

```
1 const AWS = require('aws-sdk');
2 var docClient = new AWS.DynamoDB.DocumentClient();
3 var tableName = "StudentDetails";
4 exports.handler = (event, context, callback) => {
5     console.log(event.Email)
6     var params = {
7         TableName: tableName,
8         Key: {
9             "id": event.id
10        }
11    }
12    docClient.get(params, function (err, data) {
13        callback(err, data.Item);
14    })
15};
```

At the top right of the code editor, there is a 'Deploy' button with a red circle around it. A status message 'Changes not deployed' is displayed next to it. On the far right, there is a 'Upload from' button.

**click on deploy**



This screenshot shows the same AWS Lambda code editor interface as the previous one, but with a red arrow pointing to the 'Deploy' button at the top right of the code editor. The 'Test' button is also highlighted with a blue rectangle. The code in 'index.js' remains the same as in the first screenshot.

Test event action

Create new event  Edit saved event

Event name

test

Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Event sharing settings

Private  
This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)

Shareable  
This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

Template - optional

hello-world

**Event JSON**

```

1 {
2   "id": "100"
3 }

```

**Format JSON**

```
{
  "id": "100"
}
```

Scroll down and Click on save.

Code Test Monitor Configuration Aliases Versions

**Code source** Info

File Edit Find View Go Tools Window Test Deploy

Go to Anything (Ctrl-P) index.js Environment Var +

Environment

```

1 const AWS = require('aws-sdk');
2 var docClient = new AWS.DynamoDB.DocumentClient();
3 var tableName = "StudentDetails";
4 exports.handler = (event, context, callback) => {
5   console.log(event.email)
6   var params = {
7     TableName: tableName,
8     Key: {
9       "id": event.id
10      }
11    }
12    docClient.get(params, function (err, data) {
13      if(err) {
14        callback(err, data.item);
15      }
16  });

```

Click on test.

## Home Assignment-2 Cloud and Serverless Computing

The screenshot shows the AWS Lambda Test console. A red circle highlights the "Response" object in the JSON output:

```

{
  "mobile": "7412589633",
  "bloodgroup": "A+",
  "cgpa": "9.3",
  "id": "100",
  "email": "ram@gmail.com",
  "name": "K Ram"
}
  
```

The status bar at the top right indicates "Status Succeeded" and "Max memory used: 85 MB".

If you didn't get this response, go back and check everything once again.

Now we are able to retrieve the items from dynamodb table.

Next if adding items to dynamodb.

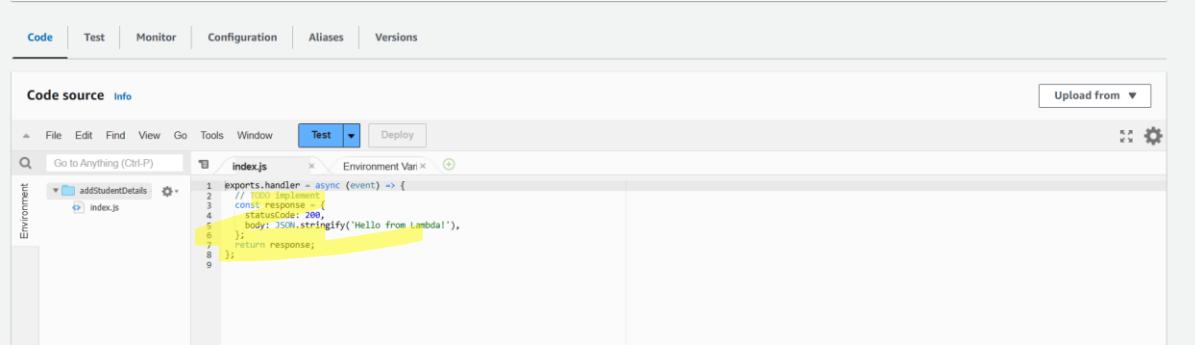
The screenshot shows the "Create function" wizard. Three steps are circled in red:

- 1.** Function name: The input field contains "addStudentDetails".
- 2.** Runtime: The dropdown menu is set to "Node.js 16.x".
- 3.** Change default execution role: The "Change default execution role" button is highlighted with a red circle.

Search for addStudentDetails role.

The screenshot shows the "Change default execution role" step. The search bar contains "addStudentDetails". The "Create function" button at the bottom right is highlighted with a red circle.

Select it and click on create function.



The screenshot shows the AWS Lambda code editor interface. At the top, there are tabs for 'Code', 'Test', 'Monitor', 'Configuration', 'Aliases', and 'Versions'. Below the tabs, there's a toolbar with 'File', 'Edit', 'Find', 'View', 'Go', 'Tools', 'Window', 'Test' (which is currently selected), and 'Deploy'. On the left, there's a sidebar with 'Environment' and a search bar 'Go to Anything (Ctrl+P)'. The main area shows a file tree with 'addStudentDetails' and 'index.js'. The 'index.js' file is open, displaying the following code:

```
1 exports.handler = async (event) => {
2     // TODO implement
3     const response = {
4         statusCode: 200,
5         body: JSON.stringify('Hello from Lambda!'),
6     };
7     return response;
8 };
```

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, "successfully updated data")
        }
    })
};
```

## Home Assignment-2 Cloud and Serverless Computing

The screenshot shows the AWS Lambda code editor interface. The tab bar at the top includes 'Code', 'Test', 'Monitor', 'Configuration', 'Aliases', and 'Versions'. Below the tabs, there's a toolbar with 'File', 'Edit', 'Find', 'View', 'Go', 'Tools', 'Window', 'Test' (which is currently selected), 'Deploy', and 'Changes not deployed'. A search bar says 'Go to Anything (Ctrl-P)'. On the left, there's a sidebar labeled 'Environment' with a folder icon 'addStudentDetails' containing 'index.js'. The main area displays the 'index.js' code:

```
1 const AWS = require("aws-sdk");
2 var docClient = new AWS.DynamoDB.DocumentClient();
3 exports.handler = (event, context, callback) => {
4     var tableName = "student";
5     var params = {
6         TableName: tableName,
7         Item: {
8             "id": event.id,
9             "name": event.name,
10            "email": event.email,
11            "mobile": event.mobile,
12            "bloodgroup": event.bloodgroup,
13            "cgpa": event.cgpa
14        }
15    };
16    docClient.put(params, function (err, data) {
17        if (err) {
18            callback(err, data);
19        } else {
20            callback(null, "sucessfully updated data")
21        }
22    })
23 });
24 }
```

**Change the tableName to StudentDetails.**

This screenshot shows the same AWS Lambda code editor interface after the 'tableName' variable has been changed to 'StudentDetails'. The 'Deploy' button in the toolbar is circled in red. The rest of the interface and code are identical to the previous screenshot.

```
1 const AWS = require("aws-sdk");
2 var docClient = new AWS.DynamoDB.DocumentClient();
3 exports.handler = (event, context, callback) => {
4     var tableName = "StudentDetails";
5     var params = {
6         TableName: tableName,
7         Item: {
8             "id": event.id,
9             "name": event.name,
10            "email": event.email,
11            "mobile": event.mobile,
12            "bloodgroup": event.bloodgroup,
13            "cgpa": event.cgpa
14        }
15    };
16    docClient.put(params, function (err, data) {
17        if (err) {
18            callback(err, data);
19        } else {
20            callback(null, "sucessfully updated data")
21        }
22    })
23 };
24 }
```

**Click on deploy.**

**Click on test**

This screenshot shows the AWS Lambda code editor interface again. A red arrow points to the 'Test' button in the toolbar. The 'Configure test event' option is visible in the dropdown menu next to the 'Test' button. The rest of the interface and code are identical to the previous screenshots.

```
1 const AWS = require("aws-sdk");
2 var docClient = new AWS.DynamoDB.DocumentClient();
3 exports.handler = (event, context, callback) => {
4     var tableName = "StudentDetails";
5     var params = {
6         TableName: tableName,
7         Item: {
8             "id": event.id,
9             "name": event.name,
10            "email": event.email,
11            "mobile": event.mobile,
12            "bloodgroup": event.bloodgroup,
13            "cgpa": event.cgpa
14        }
15    };
16    docClient.put(params, function (err, data) {
17        if (err) {
18            callback(err, data);
19        } else {
20            callback(null, "sucessfully updated data")
21        }
22    })
23 };
24 }
```

To invoke your function without saving an event, configure the JSON event, then choose Test.

Test event action

Create new event     Edit saved event

Event name  
test

Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Event sharing settings

Private  
This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)

Shareable  
This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

Template - optional

hello-world

Event JSON

```
1 {  
2   "id": "101",  
3   "name": "P Shyam",  
4   "cgpa": "8.3",  
5   "mobile": "9418529635",  
6   "email": "shyam@gmail.com",  
7   "bloodgroup": "B+"  
8 }
```

Format JSON

Cancel       

{

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

Click on save.

## Home Assignment-2 Cloud and Serverless Computing

The test event test was successfully saved.

Code source Info

File Edit Find View Go Tools Window Test Deploy

index.js Environment Variables

```
const AWS = require('aws-sdk');
var docClient = new AWS.DynamoDB.DocumentClient();
exports.handler = (event, context, callback) => {
  var tableName = "StudentDetails";
  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, "successfully updated data");
    }
  });
};
```

Click on test.

Code source Info

File Edit Find View Go Tools Window Test Deploy

index.js Environment Variables Execution result

Execution results

Test Event Name  
test

Response  
"successfully updated data"

Function Logs

```
START RequestId: 8c043931-fae7-4fc3-996d-f8f1b2ee1566 Version: $LATEST
END RequestId: 8c043931-fae7-4fc3-996d-f8f1b2ee1566
REPORT RequestId: 8c043931-fae7-4fc3-996d-f8f1b2ee1566 Duration: 799.36 ms Billed Duration: 800 ms Memory Size: 128 MB
```

Request ID  
8c043931-fae7-4fc3-996d-f8f1b2ee1566

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

DynamoDB

Tables (6)

EmployeeDetails product research student StudentDetails weather

StudentDetails

Scan or query items

Scan Query

Select a table or index Table - StudentDetails Select attribute projection All attributes

Completed. Read capacity units consumed: 0.5

Items returned (2)

	id (String)	bloodgroup	cgpa	email	mobile	name
100	100	A+	9.3	ram@gmail.com	7412589633	K Ram
101	101	B+	8.3	shyam@gmail.com	9412529635	P Shyam

**Api Gateway**

The screenshot shows the AWS Management Console search results for 'api'. The 'API Gateway' service is highlighted with a red oval. On the main search results page, there is another red oval around the 'Create API' button in the top right corner of the main content area.

**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

**Import** **Build**

API Gateway > APIs > Create API > Create REST API

## Create REST API

**API details**

New API  
Create a new REST API.

Clone existing API  
Create a copy of an API in this AWS account.

Import API  
Import an API from an OpenAPI definition.

Example API  
Learn about API Gateway with an example API.

**API name**  
StudentDetails

**Description - optional**

**API endpoint type**  
Regional

**Create API**

API name : **StudentDetails**

Click on create

API Gateway

APIs  
Custom domain names  
VPC links

▼ API: StudentDetails  
Resources  
Stages  
Authorizers  
Gateway responses  
Models  
Resource policy

API Gateway > APIs > Resources - StudentDetails (n0f0qdfgia)

### Resources

**Create resource**

/

**Resource details**

Path /

**Methods (0)**

Method type	Integration type
-------------	------------------

Click on create resource

API Gateway > APIs > Resources - StudentDetails (n0f0qdfgia) > Create resource

### Create resource

**Resource details**

Proxy resource [Info](#)  
Proxy resources handle requests to all sub-resources. To create a proxy resource use a path parameter that ends with a plus sign, for example {proxy+}.

Resource path: /

Resource name: addStudentDetails  

CORS (Cross Origin Resource Sharing) [Info](#)  
Create an OPTIONS method that allows all origins, all methods, and several common headers.

Cancel Create resource  

Resource name : **addStudentDetails**

API Gateway > APIs > Resources - StudentDetails (n0f0qdfgia)

### Resources

		API actions ▾			Deploy API												
		Resource details															
		Path	Resource ID	Delete Update documentation Enable CORS													
/		/addStudentDetails	sczzq5	<span style="border: 1px solid orange; border-radius: 5px; padding: 2px;">Create resource</span>													
		<b>Methods (0)</b> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Method type</th> <th>Integration type</th> <th>Authorization</th> <th>API key</th> </tr> </thead> <tbody> <tr> <td colspan="4" style="text-align: center;">No methods</td> </tr> <tr> <td colspan="4" style="text-align: center;">No methods defined.</td> </tr> </tbody> </table>			Method type	Integration type	Authorization	API key	No methods				No methods defined.				<span style="border: 2px solid red; border-radius: 50%; padding: 2px;"> </span>
Method type	Integration type	Authorization	API key														
No methods																	
No methods defined.																	

Click on create method .

Method type : POST

Integration type : Lambda

## Create method

**Method details**

Method type: POST

Integration type:

- Lambda function: Integrate your API with a Lambda function. (selected)
- HTTP: Integrate with an existing HTTP endpoint.
- Mock: Generate a response based on API Gateway mappings and transformations.

Lambda proxy integration

Lambda function: Integrate your API with a Lambda function. (selected)

HTTP: Integrate with an existing HTTP endpoint.

Mock: Generate a response based on API Gateway mappings and transformations.

Lambda proxy integration

Send the request to your Lambda function as a structured event.

Lambda function: us-east-1 ▾ Choose a Lambda function or enter its ARN

Grant API Gateway permission to invoke your Lambda function. To turn off, update the function's resource policy yourself, or provide an invoke role that API Gateway uses to invoke your function.

Default timeout: The default timeout is 29 seconds.

**Create method**

Search for **addStudentDetails** lambda function.

## Home Assignment-2 Cloud and Serverless Computing

The screenshot shows the AWS Lambda proxy integration configuration screen. The 'Lambda proxy integration' option is selected. The 'Send the request to' field contains 'addStudent'. The 'Lambda function' dropdown is set to 'us-east-1'. A search bar shows 'addStudent' with a red box around it. Below the search bar, an error message says 'Invalid Lambda function or Lambda function ARN.' A note below says 'Grant API Gateway permission to invoke your Lambda function. To turn off, update the function's resource policy yourself, or provide an invoke role that API Gateway uses to invoke your function.' The 'Create method' button is highlighted with a red circle.

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

The screenshot shows the AWS API Gateway Resources page. The left sidebar shows a tree structure with '/addStudentDetails' selected. The main panel shows the '/addStudentDetails - POST - Method execution' configuration. It includes fields for ARN (arn:aws:execute-api:us-east-1:439554286652:n0f0qdfgia/\*POST/addStudentDetails) and Resource ID (sczzq5). A flow diagram shows the request path from Client to Method request, then to Integration request, then to Integration response, and finally to Lambda integration. Below the diagram, tabs for Method request, Integration request, Integration response, Method response, and Test are shown, with 'Test' circled in red. A 'Test method' section allows entering query strings like 'param1=value1&param2=value2'.

Click on **Test**.

## Home Assignment-2 Cloud and Serverless Computing

Method request | Integration request | Integration response | Method response | **Test**

**Test method**  
Make a test call to your method. When you make a test call, API Gateway skips authorization and directly invokes your method.

**Query strings**  
param1=value1&param2=value2

**Headers**  
Enter a header name and value separated by a colon (:). Use a new line for each header.  
header1:value1  
header2:value2

**Client certificate**  
No client certificates have been generated.

**Request body**

```
1 {  
2     "id":"102",  
3     "name":"M Nivas",  
4     "cgpa":"9.3",  
5     "mobile":"8418529635",  
6     "email":"nivas@gmail.com",  
7     "bloodgroup":"B-"  
8 }
```

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

Scroll down and click on **Test**.

Create resource

/ /addStudentDetails POST

**Test**

**/addStudentDetails - POST method test results**

Request	Latency	Status
/addStudentDetails	510	200

**Response body**

```
"sucessfully updated data"
```

**Response headers**

```
{  
    "Content-Type": "application/json",  
    "X-Amzn-Trace-Id": "Root=1-65a02f6b-60e592bda6ea7a27af9706bb;Sampled=0;lineage=50a18364:0"  
}
```

**Log**

```
Execution log for request eb1623c9-ae79-4603-98cf-3b72a4c4cb92  
Thu Jan 11 18:11:55 UTC 2024 : Starting execution for request: eb1623c9-ae79-4603-98cf-3b72a4c4cb92  
Thu Jan 11 18:11:55 UTC 2024 : HTTP Method: POST. Resource Path: /addStudentDetails
```

Lets check in dynamodb table **StudentDetails**.

## Home Assignment-2 Cloud and Serverless Computing

**DynamoDB**

- Dashboard
- Tables
  - Update settings
  - Explore items**
  - PartiQL editor
  - Backups
  - Exports to S3
  - Imports from S3
  - Integrations **New**
  - Reserved capacity
  - Settings
- DAX
  - Clusters
  - Subnet groups
  - Parameter groups
  - Events

**Scan or query items**

**Scan**      **Query**

Select a table or index  
Table - StudentDetails

Select attribute projection  
All attributes

Filters

**Run**      **Reset**

Completed. Read capacity units consumed: 0.5

	<b>id (String)</b>	<b>bloodgroup</b>	<b>cgpa</b>	<b>email</b>	<b>mobile</b>	<b>name</b>
102	B+	9.3	nivas@gma...	8418529635	M Nivas	
100	A+	9.3	ram@gmail...	7412589633	K Ram	
101	B+	8.3	shyam@gm...	9418529635	P Shyam	

Lets create another resource to get the student details.

**API Gateway**

- APIs
- Custom domain names
- VPC links
- API: StudentDetails**
  - Resources
  - Stages
  - Authorizers
  - Gateway responses
  - Models
  - Resource policy
  - Documentation
  - Dashboard
  - API settings
- Usage plans
- API keys
- Client certificates
- Settings

**Resources**

**/addStudentDetails - POST - Method execution**

ARN: arn:aws:execute-api:us-east-1:439554286652:n0f0qdfgia/\*POST/addStudentDetails

Resource ID: sczzq5

Method request → Integration request → Integration response → Lambda integration

Method request | Integration request | Integration response | Method response | Test

**Method request settings**

Authorization: NONE      API key required: False

Click on / to go to root path .

**API Gateway**

- APIs
- Custom domain names
- VPC links
- API: StudentDetails**
  - Resources**
  - Stages
  - Authorizers
  - Gateway responses
  - Models
  - Resource policy
  - Documentation
  - Dashboard
  - API settings
- Usage plans
- API keys

**Resources**

**Resource details**

Path: /      Resource ID: x0z791jh06

**Methods (0)**

Method type	Integration type	Authorization	API key
No methods			
No methods defined.			

Now click on create resource

API Gateway > APIs > Resources - StudentDetails (n0f0qdfgia) > Create resource

## Create resource

**Resource details**

Proxy resource [Info](#)  
Proxy resources handle requests to all sub-resources. To create a proxy resource use a path parameter that ends with a plus sign, for example {proxy+}.

Resource path  Resource name

CORS (Cross Origin Resource Sharing) [Info](#)  
Create an OPTIONS method that allows all origins, all methods, and several common headers.

Cancel **Create resource**

**API Gateway**

Successfully created resource '/getStudentDetails'

API Gateway > APIs > Resources - StudentDetails (n0f0qdfgia)

**Resources**

API: StudentDetails  
Resources  
Stages  
Authorizers  
Gateway responses  
Models  
Resource policy  
Documentation  
Dashboard  
API settings

Custom domain names  
VPC links

Usage plans  
API keys  
Client certificates  
Settings

**Resource details**

Path /getStudentDetails Resource ID q2fhut

**Methods (0)**

Method type Integration type Authorization API key

No methods

No methods defined.

**Create method**

Click on create method in **getStudentDetails**.

Method Type : GET

Integration type : Lambda

### Create method

**Method details**

Method type: GET (circled)

Integration type:

- Lambda function (circled) Integrate your API with a Lambda function.
- HTTP Integrate with an existing HTTP endpoint.
- Mock Generate a response based on API Gateway mappings and transformations.
- AWS service Integrate with an AWS Service.
- VPC link Integrate with a resource that isn't accessible over the public internet.
- Lambda proxy integration Send the request to your Lambda function as a structured event.

Then in Lambda function option search for getStudentDetails Lambda function.

Integrate with AWS services

Integrate with a resource that isn't accessible over the public internet.

Lambda proxy integration: getStudentDe (highlighted with yellow box and circled) Send the request to `arn:aws:lambda:us-east-1:439554286652:function:getStudentDetails`

Lambda function: Provide the Lambda function name: `getstudentdetails`

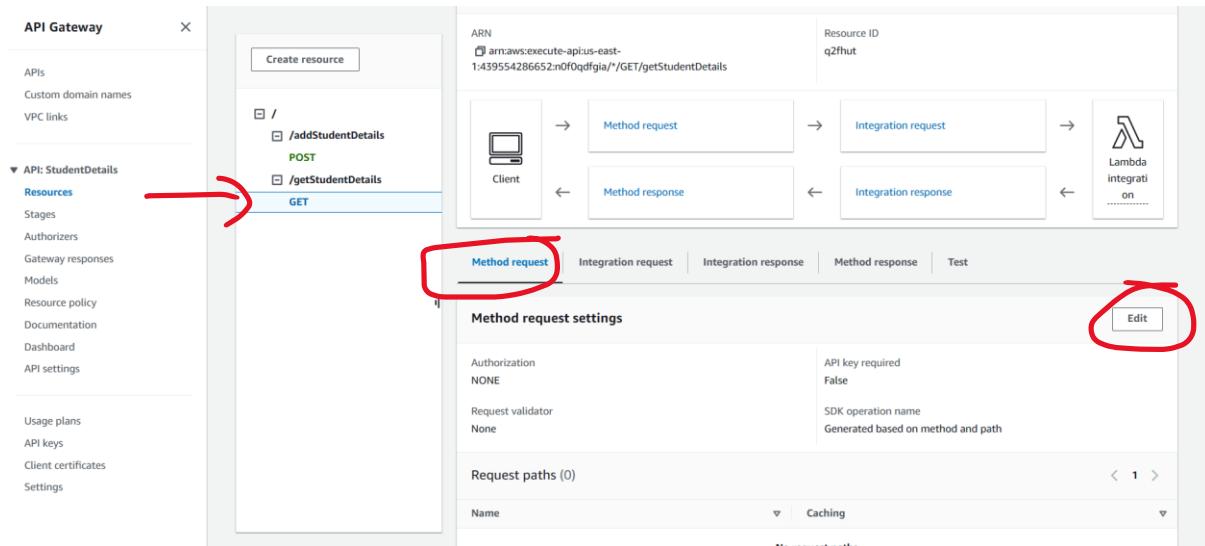
Region: us-east-1 (dropdown) Search bar: getStudentDe (highlighted with red box) Error message: Invalid Lambda function or Lambda function ARN.

Grant API Gateway permission to invoke your Lambda function. To turn off, update the function's resource policy yourself, or provide an invoke role that API Gateway uses to invoke your function.

Default timeout: The default timeout is 30 seconds.

Next , Click on **create method**.

## Home Assignment-2 Cloud and Serverless Computing



Go to Method request.

Click on Edit.

The screenshot shows the 'Edit method request' dialog. At the top, it says 'Edit method request'. Below that is a section titled 'Method request settings' with fields for Authorization (set to 'None'), Request validator (set to 'None'), and a checkbox for 'API key required' which is unchecked. There's also an 'Operation name - optional' field with the value 'GetPets'. Below this, there's a section titled 'URL query string parameters' which is expanded, indicated by a red circle around the title. Underneath it, it says 'No query strings found' and has a button labeled 'Add query string' with a red arrow pointing to it. At the bottom, there's another section titled 'HTTP request headers'.

Click on Add query string.

## Home Assignment-2 Cloud and Serverless Computing

□ API key required

Operation name - optional  
GetPets

▼ URL query string parameters

Name	Required	Caching
id	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Add query string

Scroll down and click on save .

Method request | Integration request | Integration response | Method response | Test

**Method request settings**

Authorization	API key required
NONE	False
Request validator	SDK operation name
None	Generated based on method and path

Request paths (0)

No request paths

No request paths defined

URL query string parameters (1)

Name	Required	Caching
id	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

HTTP request headers (0)

Click on Integration Request.

Click on edit.

API Gateway > APIs > Resources - StudentDetails (n0f0qdfgia)

Resources

/getStudentDetails - GET - Method execution

ARN: arn:aws:execute-api:us-east-1:439554286652:n0f0qdfgia/\*GET/getStudentDetails

Resource ID: q2Rfut

Client → Method request → Integration request → Lambda integration

Method request → Integration request → Integration response → Lambda integration

Integration request

Edit

Integration request settings

Integration type: Lambda

Region: us-east-1

Lambda function

Scroll down to **Mapping templates** option.

When no template matches the request content-type header

**⚠ If you set the request body passthrough to When no templates matches the request content-type header, API Gateway will pass all request payloads directly to the endpoint without transformation, and will transform any matches for the incoming content type. To secure your integration, select When there are no templates defined (recommended).**

- ▶ URL path parameters
- ▶ URL query string parameters
- ▶ URL request headers parameters
- ▼ Mapping templates

No mapping templates associated with this integration.

Add mapping template

Cancel Save

Click on add mapping template.

Content type

application/json

Remove

Generate template

Method request passthrough

Template body

```

1 ## See https://docs.aws.amazon.com/apigateway/latest
   /developerguide/api-gateway-mapping-template-reference
   .html
2 ## This template will pass through all parameters including
   path, querystring, header, stage variables, and context
   through to the integration endpoint via the body/payload
3 #set($allParams = $input.params())
4 {
5   "body-json" : $input.json('$'),
6   "params" : {
7     #foreach($type in $allParams.keySet())
8       #set($params = $allParams.get($type))
9     #type" : {
10       #foreach($paramName in $params.keySet())
11         "$paramName" : "$util.escapeJavaScript($params.get(
12           $paramName))"
12         #if($foreach.hasNext),#end

```

Replace the Template body with the below one :

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

**Mapping templates**

Content type: application/json Remove

Generate template: Method request passthrough

Template body:

```

1 {
2     "id" : "$input.params('id')"
3 }
4
    
```

Use VTL templates to create your mapping template. [Learn more](#)

Add mapping template

Save Cancel

Click on save.

**API Gateway**

APIs  
Custom domain names  
VPC links

**API: StudentDetails**

- Resources →
- Stages
- Authorizers
- Gateway responses
- Models
- Resource policy
- Documentation
- Dashboard
- API settings

Usage plans  
API keys  
Client certificates

Create resource

```

    / 
    /addStudentDetails POST
    /getStudentDetails GET
    
```

arn:aws:execute-api:us-east-1:439554286652:n0f0qqdfgia/\*:GET/getStudentDetails q2ffhut

Method request → Integration request → Integration response → Lambda integration

Client ← Method response ← Integration response ← Lambda integration

Method request | Integration request | Integration response | Method response Test

**Test method**  
Make a test call to your method. When you make a test call, API Gateway skips authorization and directly invokes your method.

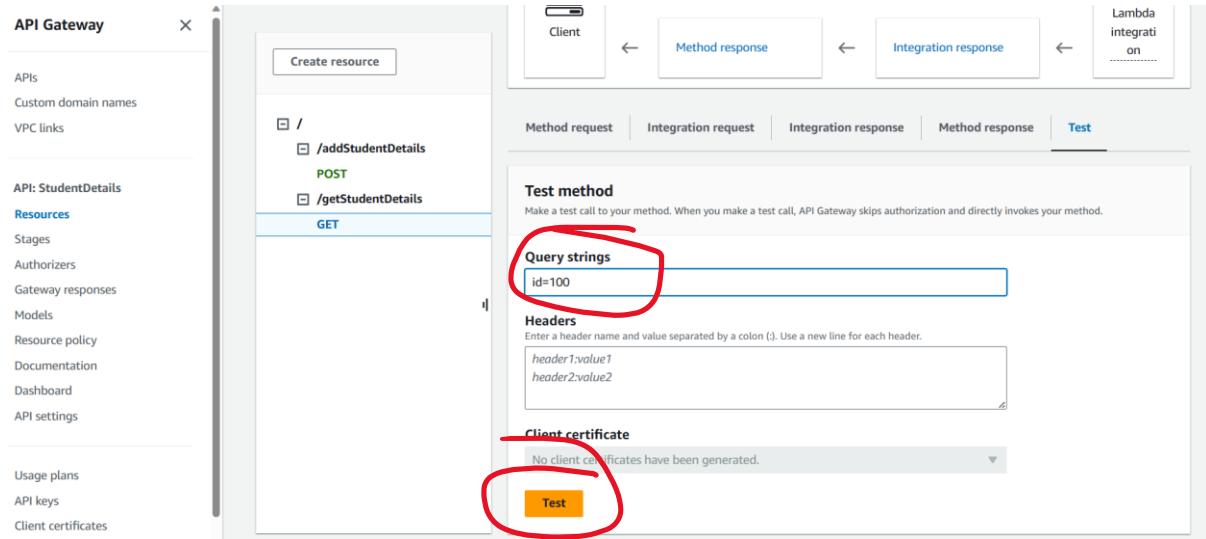
**Query strings**  
param1=value1&param2=value2

**Headers**  
Enter a header name and value separated by a colon (:). Use a new line for each header.  
header1:value1  
header2:value2

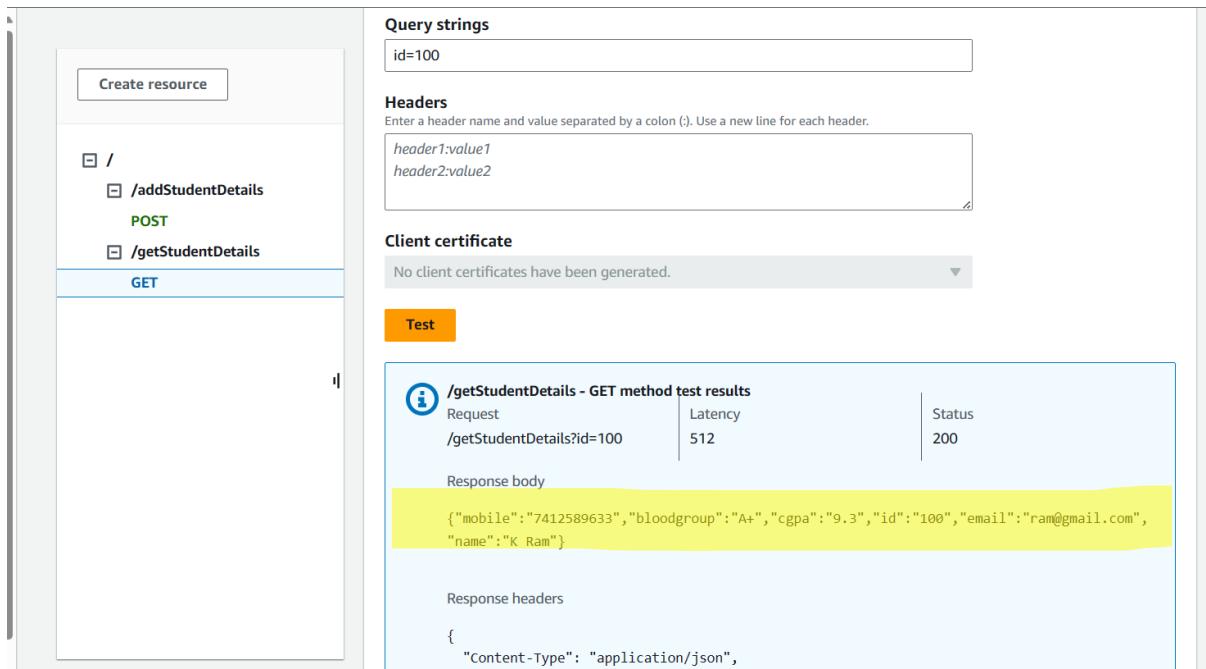
Click on Test.

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

## Home Assignment-2 Cloud and Serverless Computing



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

## Home Assignment-2 Cloud and Serverless Computing

The screenshot shows the AWS API Gateway interface. On the left, there's a sidebar with options like APIs, Custom domain names, VPC links, and a detailed section for 'API: StudentDetails' which includes Resources, Stages, Authorizers, Gateway responses, Models, Resource policy, Documentation, Dashboard, and API settings. The main area is titled 'Resources' and shows a tree structure. Under the root node, there's a 'POST' method for '/addStudentDetails' and a 'GET' method for '/getStudentDetails'. A red arrow points to the '/getStudentDetails' resource. In the top right corner of the main area, there are buttons for 'API actions', 'Deploy API', and 'Enable CORS', with the 'Enable CORS' button highlighted by a red box.

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

[API Gateway](#) > [APIs](#) > [Resources - StudentDetails \(n0f0qdfgia\)](#) > [Enable CORS](#)

### Enable CORS

#### CORS settings Info

To allow requests from scripts running in the browser, configure cross-origin resource sharing (CORS) for your API.

##### Gateway responses

API Gateway will configure CORS for the selected gateway responses.

- Default 4XX
- Default 5XX

##### Access-Control-Allow-Methods

- GET
- OPTIONS

##### Access-Control-Allow-Headers

API Gateway will configure CORS for the selected gateway responses.

Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-Token

##### Access-Control-Allow-Origin

Enter an origin that can access the resource. Use a wildcard '\*' to allow any origin to access the resource.

\*

#### ► Additional settings

Check all the options and click on save.

API Gateway > APIs > Resources - StudentDetails (n0f0qdffg) > Enable CORS

## Enable CORS

**CORS settings** Info

To allow requests from scripts running in the browser, configure cross-origin resource sharing (CORS) for your API.

**Gateway responses**  
API Gateway will configure CORS for the selected gateway responses.

Default 4XX  
 Default 5XX

**Access-Control-Allow-Methods**

GET  
 OPTIONS

**Access-Control-Allow-Headers**  
API Gateway will configure CORS for the selected gateway responses.

Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-Token

**Access-Control-Allow-Origin**  
Enter an origin that can access the resource. Use a wildcard '\*' to allow any origin to access the resource.

\*

**Additional settings**

Cancel **Save**

API Gateway API Gateway > APIs > Resources - StudentDetails (n0f0qdffg) API actions Deploy API

**Resources**

Create resource

Path /  
Resource ID sczzq5

**Methods (1)**

Method type	Integration type	Authorization	API key
POST	Lambda	None	Not required

**Enable CORS**

**API: StudentDetails**

Resources (highlighted by a red arrow)

Stages  
Authorizers  
Gateway responses  
Models  
Resource policy  
Documentation  
Dashboard  
API settings

Usage plans  
API keys  
Client certificates  
Settings

Now lets do the same for addStudentDetails resource.

**CORS settings Info**

To allow requests from scripts running in the browser, configure cross-origin resource sharing (CORS) for your API.

**Gateway responses**

API Gateway will configure CORS for the selected gateway responses.

- Default 4XX
- Default 5XX

**Access-Control-Allow-Methods**

API Gateway will configure CORS for the selected gateway responses.

- OPTIONS
- POST

**Access-Control-Allow-Headers**

API Gateway will configure CORS for the selected gateway responses.

Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-Token

**Access-Control-Allow-Origin**

Enter an origin that can access the resource. Use a wildcard (\*) to allow any origin to access the resource.

\*

**Additional settings**

Cancel **Save**

Click on save .

Now click on root path /

**API Gateway**

**Resources**

Create resource

**Resource details**

Path / Resource ID x02791jho6

**Methods (0)**

No methods

Method type Integration type Authorization API key

API actions Deploy API

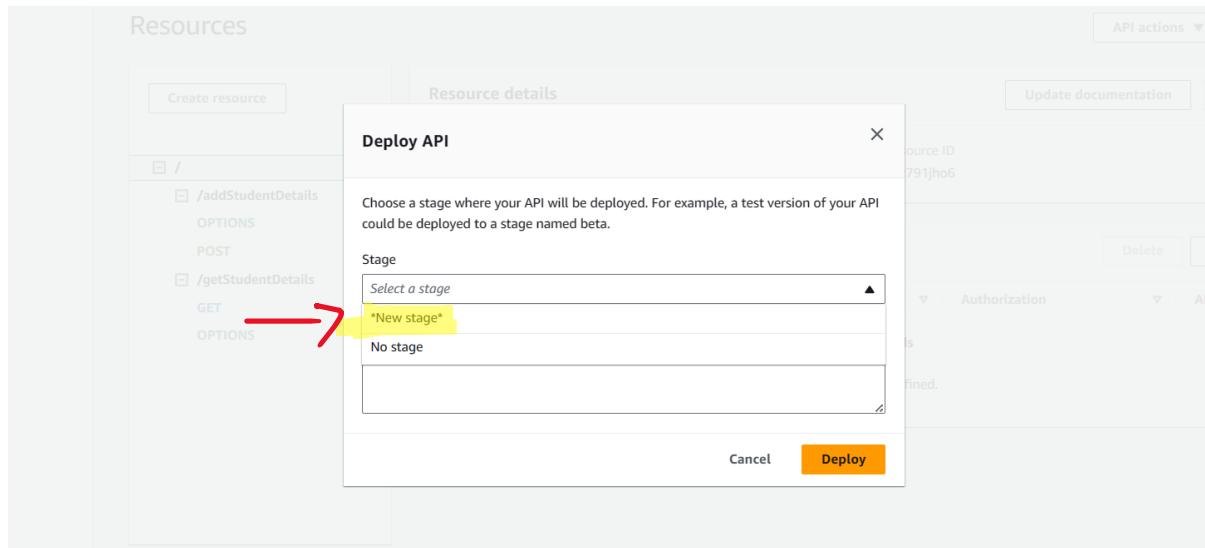
Update documentation Enable CORS

**API: StudentDetails**

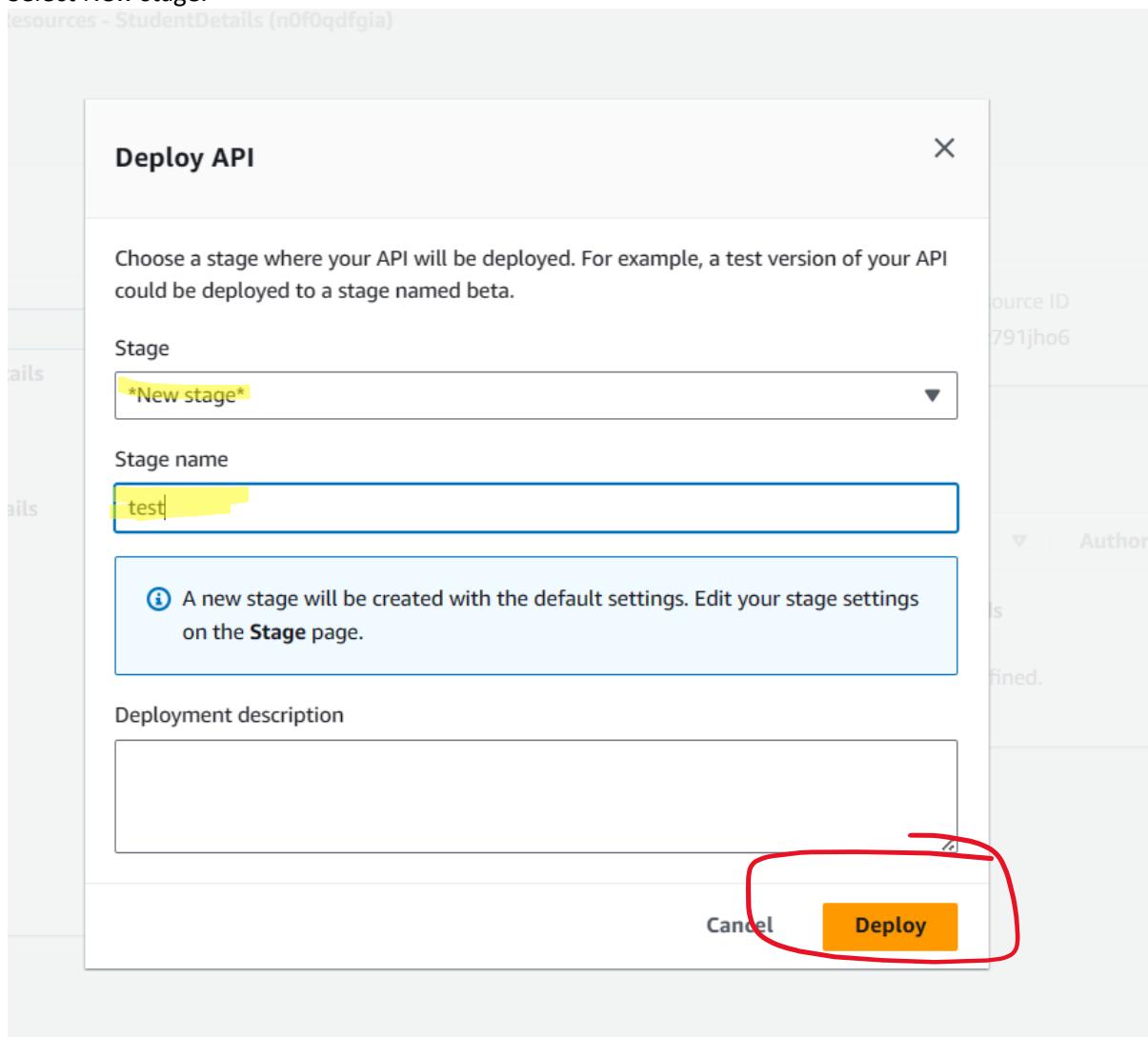
- Resources
- Stages
- Authorizers
- Gateway responses
- Models
- Resource policy
- Documentation
- Dashboard
- API settings

Usage plans API keys Client certificates Settings

Deploy the API.



Select New stage.



Click on Deploy .

## Home Assignment-2 Cloud and Serverless Computing

The screenshot shows the AWS API Gateway interface. On the left, there's a sidebar with navigation links for APIs, VPC links, API resources, stages, authorizers, gateway responses, models, resource policy, documentation, dashboard, and API settings. The main area is titled 'Stages' and shows a single stage named 'test'. The 'Stage details' section includes fields for Stage name (test), Rate Info, Burst Info, Web ACL, Client certificate, and an Invoke URL (https://n0f0qdfqja.execute-api.us-east-1.amazonaws.com/test). Below this is an 'Active deployment' section showing '2ihz8 on January 12, 2024, 00:06 (UTC+05:30)'. The 'Logs and tracing' section includes CloudWatch logs, Detailed metrics, and X-Ray tracing, all set to inactive.

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.

The screenshot shows the POSTMAN interface. The URL in the address bar is https://n0f0qdfqja.execute-api.us-east-1.amazonaws.com/test/addStudentDetails. A red arrow points to the 'Send' button. The request method is set to 'POST'. The Headers tab shows several headers: Content-Type, Content-Length, Host, User-Agent, and X-Amzn-Trace-Id. The Body tab is selected and contains the JSON payload: {"name": "John", "age": 30, "city": "New York"}. The response status is 200 OK with a timestamp of 2024-01-12T00:06:30Z and a size of 368 B.

Goto raw

## Home Assignment-2 Cloud and Serverless Computing

The screenshot shows the Postman interface with a red arrow pointing from the JSON editor area down to the response body. The JSON editor shows the input data, and the response body shows the successful update message.

https://n0f0qdfgia.execute-api.us-east-1.amazonaws.com/test/addStudentDetails

POST https://n0f0qdfgia.execute-api.us-east-1.amazonaws.com/test/addStudentDetails

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1
2   "id":"103",
3   "name":"K-Harsha",
4   "cgpa":"9.9",
5   "mobile":"9918529635",
6   "email":"harsha@gmail.com",
7   "bloodgroup":"B-"
8 }
```

Body Cookies Headers (8) Test Results

Status: 200 OK Time: 1813 ms Size: 368 B Save as example ...

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

Click on Send.

The screenshot shows the Postman interface after sending the request. A red arrow points from the 'Raw' tab in the response preview to the message "successfully updated data".

https://n0f0qdfgia.execute-api.us-east-1.amazonaws.com/test/addStudentDetails

POST https://n0f0qdfgia.execute-api.us-east-1.amazonaws.com/test/addStudentDetails

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "id": "103",
3   "name": "K-Harsha",
4   "cgpa": "9.9",
5   "mobile": "9918529635",
6   "email": "harsha@gmail.com",
7   "bloodgroup": "B-"
8 }
```

Body Cookies Headers (8) Test Results

Status: 200 OK Time: 791 ms Size: 368 B Save as example ...

Pretty Raw Preview Visualize JSON

```
1 "successfully updated data"
```

Lets check in dynamodb table **StudentDetails**.

## Home Assignment-2 Cloud and Serverless Computing

The screenshot shows the AWS DynamoDB console. On the left, there's a sidebar with options like Dashboard, Tables, Update settings, Explore items, PartiQL Editor, Backups, Exports to S3, Imports from S3, Integrations, Reserved capacity, and Settings. Under DAX, there are Clusters, Subnet groups, Parameter groups, and Events. The main area is titled "Scan or query items" and shows a table named "StudentDetails". It has two tabs: "Scan" (selected) and "Query". Below the table, it says "Completed. Read capacity units consumed: 0.5". The table has columns: id (String), bloodgroup, cgpa, email, mobile, and name. There are four items returned:

	id (String)	bloodgroup	cgpa	email	mobile	name
1	103	B-	9.9	harsha@g...	9918529635	K Harsha
2	102	B-	9.3	nivas@gma...	8418529635	M Nivas
3	100	A+	9.5	ram@gmail...	7412589633	K Ram
4	101	B+	8.5	shyam@gm...	9418529635	P Shyam

To get student details the endpoint will be

Invoke URL : your\_api\_gateway\_url/getStudentDetails

And method will GET.

The screenshot shows the Postman API client. It displays two requests:

- Request 1:** GET https://n0f0qdfgia.execute-api.us-east-1.amazonaws.com/test/getStudentDetails  
Params: id=103 (highlighted with a red arrow)
- Request 2:** GET https://n0f0qdfgia.execute-api.us-east-1.amazonaws.com/test/getStudentDetails?id=103  
Params: id=103 (highlighted with a red arrow)

Both requests show a status of 200 OK with a response size of 368 B. The "Body" tab is selected for both requests, showing the response body as JSON:

```
[{"id": 103, "bloodgroup": "B-", "cgpa": 9.9, "email": "harsha@g...", "mobile": "9918529635", "name": "K Harsha"}, {"id": 102, "bloodgroup": "B-", "cgpa": 9.3, "email": "nivas@gma...", "mobile": "8418529635", "name": "M Nivas"}, {"id": 100, "bloodgroup": "A+", "cgpa": 9.5, "email": "ram@gmail...", "mobile": "7412589633", "name": "K Ram"}, {"id": 101, "bloodgroup": "B+", "cgpa": 8.5, "email": "shyam@gm...", "mobile": "9418529635", "name": "P Shyam"}]
```

Click on Send.

## Home Assignment-2 Cloud and Serverless Computing

The screenshot shows a Postman interface with the following details:

- URL:** https://n0f0qdfgia.execute-api.us-east-1.amazonaws.com/test/getStudentDetails?id=103
- Method:** GET
- Params:** id = 103
- Body:** Status: 200 OK, Time: 2.26 s, Size: 453 B
- Response Body (Pretty JSON):**

```
1 {
2     "mobile": "9918529635",
3     "bloodgroup": "B-",
4     "cgpa": "9.9",
5     "id": "103",
6     "email": "harsha@gmail.com",
7     "name": "K Harsha"
8 }
```

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 \(sharepoint.com\)](#)

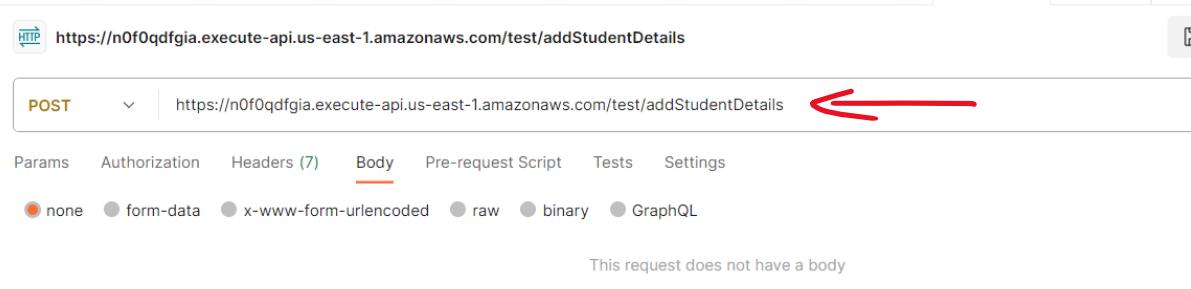
After downloading the files. Place them in a folder.

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

```
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('your_api_gateway_endpoint_to_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             })
77     }
78 </script>
```

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

## Home Assignment-2 Cloud and Serverless Computing



https://n0f0qdfgia.execute-api.us-east-1.amazonaws.com/test/addStudentDetails

POST https://n0f0qdfgia.execute-api.us-east-1.amazonaws.com/test/addStudentDetails

Params Authorization Headers (7) Body Pre-request Script Tests Settings

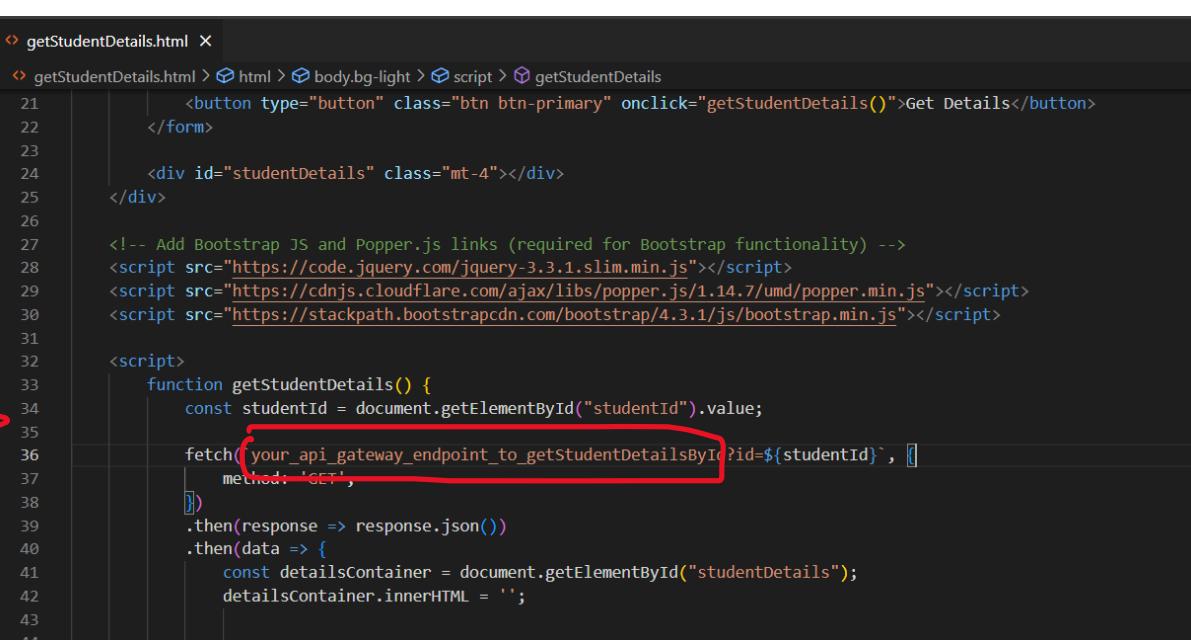
none form-data x-www-form-urlencoded raw binary GraphQL

This request does not have a body

```
> addStudentDetails.html X
< 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
77         })
78     }
79   </script>
```

Save the file.

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



```
> getStudentDetails.html X
< 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         })
44     }
45   </script>
```

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= ) .

HTTP <https://n0f0qdfgia.execute-api.us-east-1.amazonaws.com/test/getStudentDetails?id=103>

GET <https://n0f0qdfgia.execute-api.us-east-1.amazonaws.com/test/getStudentDetails?id=103>

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

Key	Value	Description
<input checked="" type="checkbox"/> id	103	
Key	Value	Description

```

getStudentDetails.html X
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('https://n0f0qdfgia.execute-api.us-east-1.amazonaws.com/test/getStudentDetails?id=${studentId}', {
37       method: 'GET'
38     })
39     .then(response => response.json())
40     .then(data => {
41       const detailsContainer = document.getElementById("studentDetails");
42       detailsContainer.innerHTML = '';
43     })
44   }

```

Save the file.

S3

aws Services  X

Search results for 'S3'

Services (8)

Features (30)

- Loading

**S3** Scalable Storage in the Cloud

S3 Glacier Archive Storage in the Cloud

AWS Snow Family Large Scale Data Transport

Storage Gateway Hybrid Storage Integration

Reset to default layout

Find applications

No applications

Create application

**Amazon S3**

Buckets  
Access Grants  
Access Points  
Object Lambda Access Points  
Multi-Region Access Points  
Batch Operations  
IAM Access Analyzer for S3

Block Public Access settings for

**Amazon S3** > **Buckets** > **Create bucket**

## Create bucket Info

Buckets are containers for data stored in S3. [Learn more](#)

**General configuration**

**AWS Region**  
US East (N. Virginia) us-east-1

**Bucket type** [Info](#)

**General purpose**  
Recommended for most use cases and access patterns.  
General purpose buckets are the original S3 bucket type.  
They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

**Directory - New**  
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

**Bucket name** [Info](#)  
**student-details**

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](#)

**You can use any valid name.**

## Home Assignment-2 Cloud and Serverless Computing

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.

## Home Assignment-2 Cloud and Serverless Computing

No tags associated with this bucket.

Add tag

**Default encryption** [Info](#)  
Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type | [Info](#)  
 Server-side encryption with Amazon S3 managed keys (SSE-S3)  
 Server-side encryption with AWS Key Management Service keys (SSE-KMS)  
 Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)  
Secure your objects with two separate layers of encryption. For details on pricing, see [DSSE-KMS pricing](#) on the [Storage](#) tab of the [Amazon S3 pricing page](#).

**Bucket Key**  
Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. [Learn more](#)

Disable  
 Enable

► Advanced settings

After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

Cancel **Create bucket**

Click on create bucket.

Goto the newly created bucket.

Amazon S3 > [Buckets](#) > student-details-website

### student-details-website [Info](#)

Objects Properties Permissions Metrics Management Access Points

**Objects (0) Info**  
Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

[C](#) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) **Upload**

Find objects by prefix

Name	Type	Last modified	Size	Storage class
No objects				
You don't have any objects in this bucket.				
<a href="#">Upload</a>				

Click on upload.

## Home Assignment-2 Cloud and Serverless Computing

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)**

All files and folders in this table will be uploaded.

Find by name

Name	Folder	Type	Size
No files or folders			

You have not chosen any files or folders to upload.

**Destination Info**

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)**

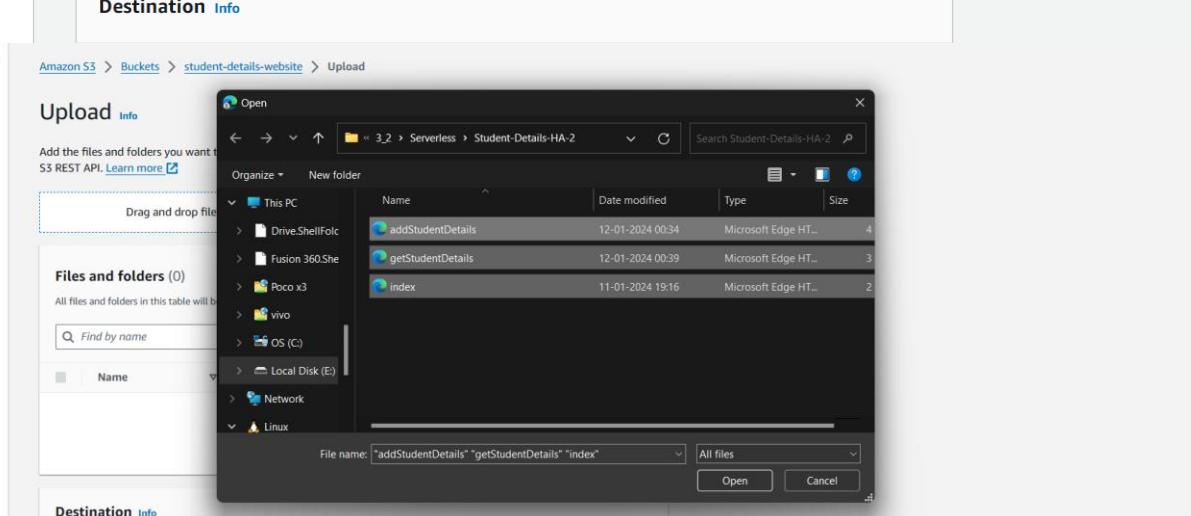
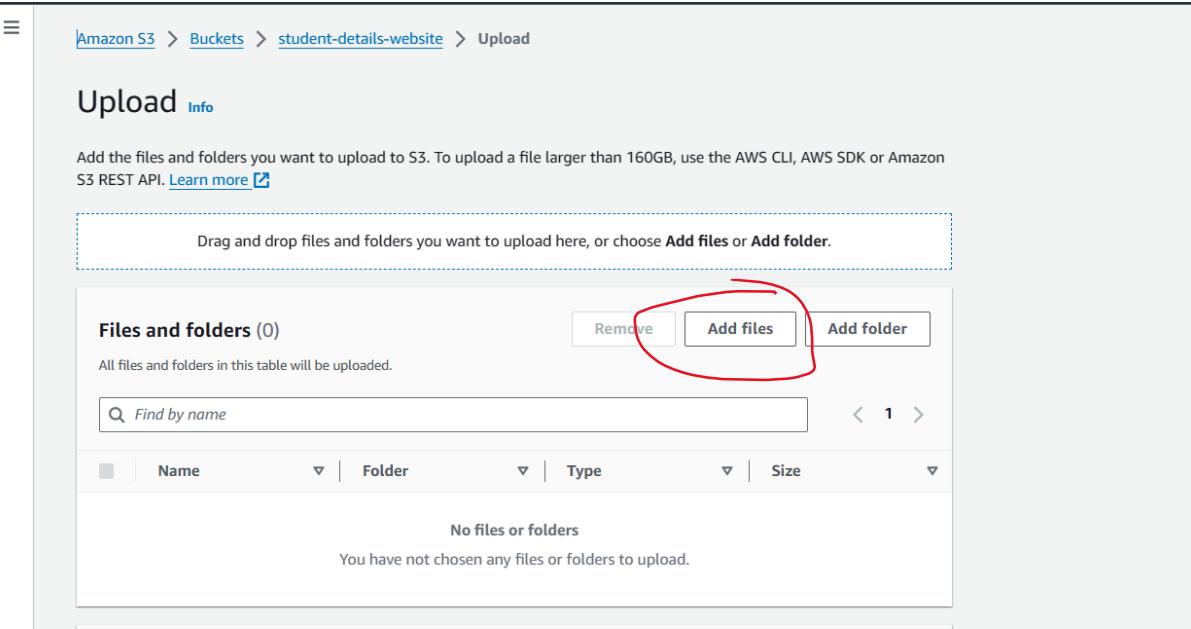
All files and folders in this table will be uploaded.

Find by name

Name	Date modified	Type	Size
addStudentDetails	12-01-2024 00:34	Microsoft Edge HT...	4
getStudentDetails	12-01-2024 00:39	Microsoft Edge HT...	3
index	11-01-2024 19:16	Microsoft Edge HT...	2

File name: "addStudentDetails" "getStudentDetails" "index"

**Destination Info**



## Home Assignment-2 Cloud and Serverless Computing

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

The screenshot shows the AWS Lambda function configuration page. In the 'File and folders' section, three files are listed: 'addStudentDetails.html' (text/html, 3.9 KB), 'getStudentDetails.html' (text/html, 2.5 KB), and 'index.html' (text/html, 1.3 KB). Below this, the 'Destination' configuration is shown, pointing to 's3://student-details-website'. Under 'Destination', there are sections for 'Destination details' (Bucket settings that impact new objects stored in the specified destination) and 'Permissions' (Grant public access and access to other AWS accounts). At the bottom right, there are 'Cancel' and 'Upload' buttons, with 'Upload' highlighted by a red oval.

The screenshot shows the AWS S3 bucket 'student-details-website'. The 'Objects' list contains three files: 'addStudentDetails.html', 'getStudentDetails.html', and 'index.html', all of which are html files. The 'Permissions' tab is highlighted by a red oval. At the top of the page, the 'Permissions' tab is also circled in red. The 'Actions' dropdown menu includes options like 'Copy S3 URI', 'Copy URL', 'Download', 'Open', 'Delete', 'Create folder', and 'Upload'.

The screenshot shows the 'Bucket policy' section of the AWS S3 bucket 'student-details-website'. A red oval highlights the 'Bucket policy' link, which is described as 'The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts.' Below this, it says 'No policy to display.' There are 'Edit' and 'Delete' buttons at the top right of the policy area, which are also highlighted by a red oval.

Click on edit.

Amazon S3 > Buckets > student-details-website > Edit bucket policy

### Edit bucket policy [Info](#)

**Bucket policy**

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. [Learn more](#)

**Bucket ARN**  
arn:aws:s3:::student-details-website

**Policy**

1

**Edit statement**

**Select a statement**  
Select an existing statement in the policy or add a new statement.

+ Add new statement

### 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 ('\*')

**Amazon Resource Name (ARN)**

GetMultiRegionAccessPointPolicyStatus  
GetMultiRegionAccessPointRoutes  
 GetObject  
GetObjectAcl  
GetObjectAttributes  
GetObjectLegalHold  
GetObjectRetention  
GetObjectTagging

{BucketName}/\${KeyName}.

d. You must enter a valid ARN.

### Step 3: Generate Policy

Amazon S3 > Buckets > student-details-website > Edit bucket policy

### Edit bucket policy [Info](#)

#### Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. [Learn more](#)

**Bucket ARN**  
arn:aws:s3:::student-details-website

Copy

#### Policy

1

## Home Assignment-2 Cloud and Serverless Computing

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

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

**Amazon Resource Name (ARN)**  \*

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**  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 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.**

This AWS Policy Generator is provided for informational purposes only. You are still responsible for your use of Amazon Web Services technologies and ensuring that your use is in accordance with the [AWS Customer Agreement](#).

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**

## Home Assignment-2 Cloud and Serverless Computing

The screenshot shows the AWS Policy Generator interface. A modal window titled "Policy JSON Document" displays the following JSON policy:

```
{ "Id": "Policy1705000490862", "Version": "2012-10-17", "Statement": [ { "Sid": "Stmt1705000474423", "Action": [ "s3:GetObject" ], "Effect": "Allow", "Resource": "arn:aws:s3:::student-details-website/*", "Principal": "*" } ] }
```

The modal includes instructions: "Click below to edit. To save the policy, copy the text below to a text editor. Changes made below will not be reflected in the policy generator tool." Below the modal, a note states: "This AWS Policy Generator is provided for informational purposes only, you are still responsible for your use of Amazon Web Services technologies and ensuring that your use is in compliance with all applicable terms and conditions. This AWS Policy Generator is provided as is without warranty of any kind, whether express or implied." A "Close" button is at the bottom right of the modal.

Copy the policy .

Go back and paste it in bucket policy.

The screenshot shows the "Edit bucket policy" page in the AWS S3 console. The URL is [Amazon S3 > Buckets > student-details-website > Edit bucket policy](#). The page title is "Edit bucket policy".

The "Bucket policy" section contains the following JSON policy:

```
1  {
2   "Id": "Policy1705000490862",
3   "Version": "2012-10-17",
4   "Statement": [
5     {
6       "Sid": "Stmt1705000474423",
7       "Action": [
8         "s3:GetObject"
9       ],
10      "Effect": "Allow",
11      "Resource": "arn:aws:s3:::student-details-website/*",
12      "Principal": "*"
13    }
14  ]
15 }
```

On the right side, there are buttons for "Policy examples" and "Policy generator". Below the policy, there is an "Edit statement" button and a "Select a statement" dropdown. A link "+ Add new statement" is visible. At the bottom, there are status indicators: "JSON Ln 15, Col 1", "Security: 0 Errors: 0 Warnings: 0 Suggestions: 0", and a "Preview external access" link. On the far right, there are "Cancel" and "Save changes" buttons, with "Save changes" being highlighted by a red oval.

Amazon S3 > Buckets > student-details-website

**student-details-website** [Info](#)

Objects (3) [Info](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 Inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

[C](#) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

[Q Find objects by prefix](#)

Name	Type	Last modified	Size	Storage class
addStudentDetails.html	html	January 12, 2024, 00:41:19 (UTC+05:30)	3.9 KB	Standard
getStudentDetails.html	html	January 12, 2024, 00:41:20 (UTC+05:30)	2.5 KB	Standard
index.html	html	January 12, 2024, 00:41:21 (UTC+05:30)	1.3 KB	Standard

Goto properties.

Scroll down, goto static web hosting.

Transfer acceleration

Use an accelerated endpoint for faster data transfers. [Learn more](#)

Transfer acceleration

Disabled

Object Lock

Store objects using a write-once-read-many (WORM) model to help you prevent objects from being deleted or overwritten for a fixed amount of time or indefinitely. Object Lock works only in versioned buckets. [Learn more](#)

Object Lock

Disabled

Requester pays

When enabled, the requester pays for requests and data transfer costs, and anonymous access to this bucket is disabled. [Learn more](#)

Requester pays

Disabled

**Static website hosting**

Use this bucket to host a website or redirect requests. [Learn more](#)

Static website hosting

Disabled

[Edit](#)

**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](#)

**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 DetailsSearch Student



### Student Details Form

ID:

Name:

CGPA:

Mobile:

Email:

Blood Group:

Submit

## Student Details Form

ID:

Name:

CGPA:

Mobile:

Email:

Blood Group:

Student details submitted successfully!

Go back and lets review search student.

## Student Details

[Add Student Details](#) [Search Student](#)

## Get Student Details

Enter Student ID:

## Get Student Details

Enter Student ID:

### Get Student Details

Enter Student ID:

**Get Details**

**ID:** 104  
**Name:** Shinchan  
**CGPA:** 7.8  
**Mobile:** 6325874199  
**Email:** shinchan@gmail.com  
**Blood Group:** AB+

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