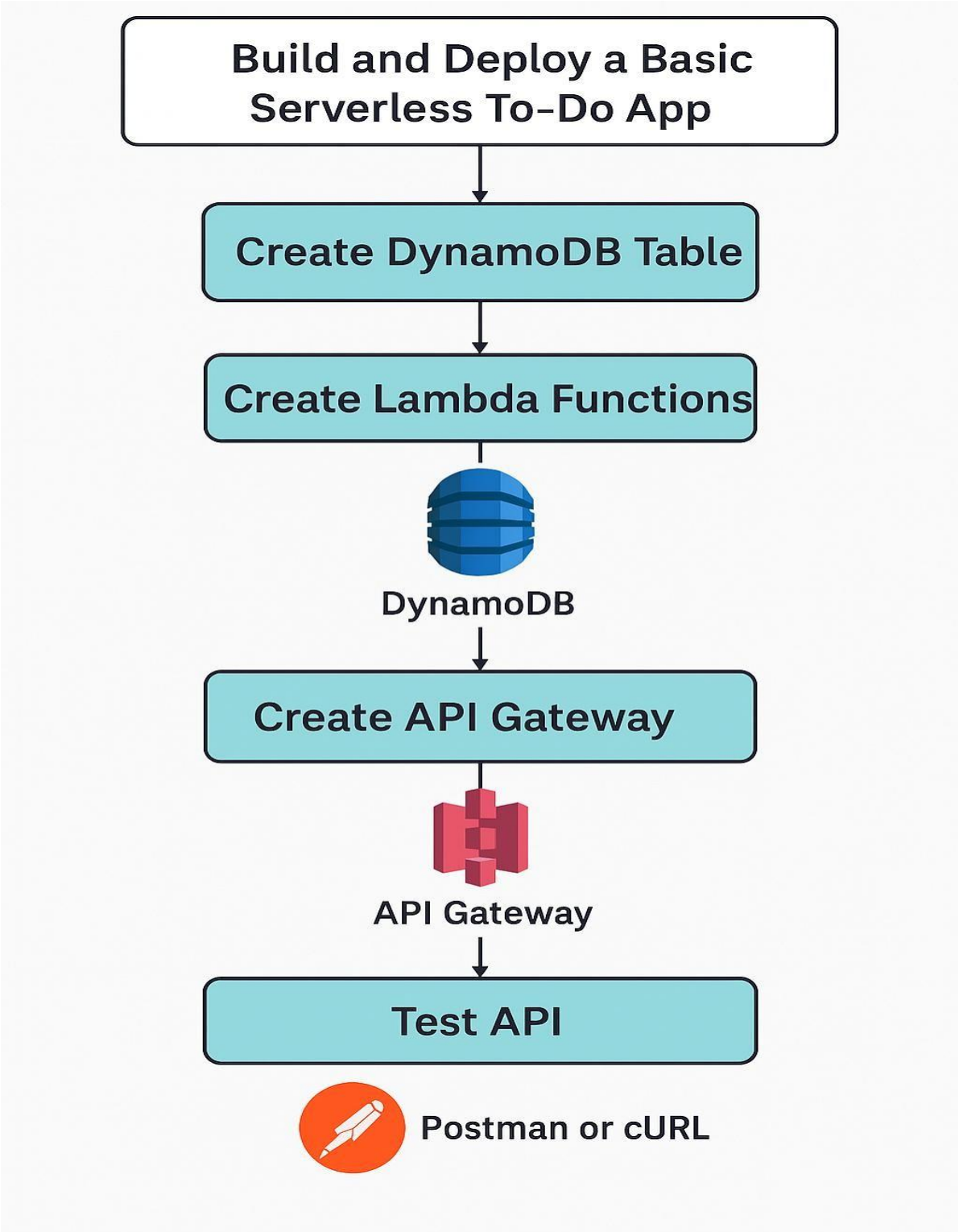
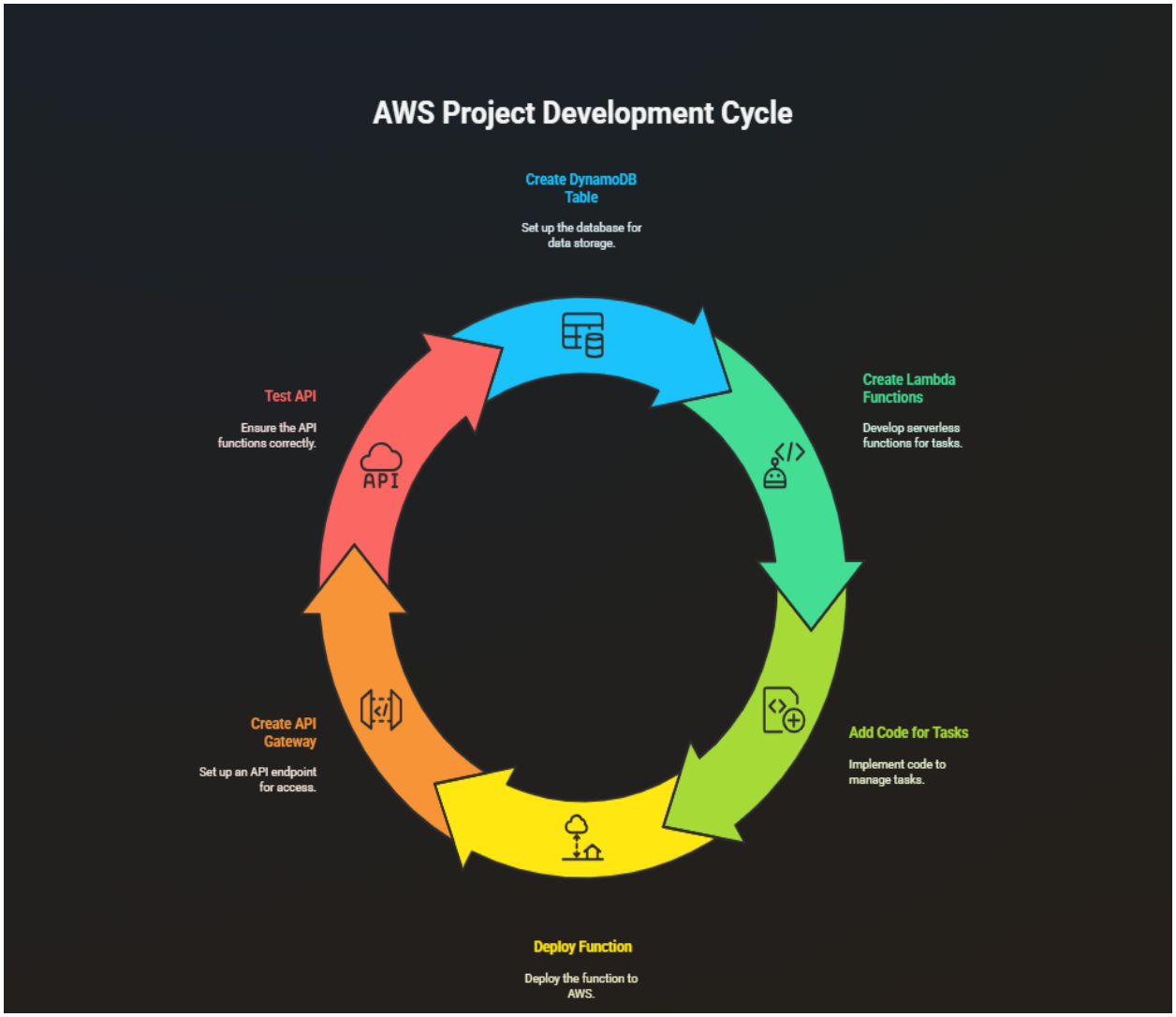
Project: Serverless To-Do App Backend (Full CRUD)





Tech Stack

- **DynamoDB** – NoSQL database to store tasks
- **AWS Lambda** – Serverless functions for logic
- **API Gateway** – Exposes API endpoints
- **CloudWatch** – For logging/debugging

TASK 1: Setup DynamoDB and Lambda Functions

Step 1: Create DynamoDB Table

1. Go to **DynamoDB** from AWS Console.
2. Click **"Create Table"**.
3. Table name: ToDoTable
   Partition key: taskId (String)
4. Leave the rest as default and **Create**.

   DynamoDB stores each task as an item. We use taskId to uniquely identify tasks.

Step 2: Create Lambda Function – ========== AddTask

1. Go to **Lambda → Create Function**
   - Choose:
   - Author from scratch
   - Function name: AddTask
   - Runtime: Python 3.x
3. Code given for reference Paste this code:

```
import boto3
import json
import uuid

def lambda_handler(event, context):
    dynamodb = boto3.resource('dynamodb')
    table = dynamodb.Table('ToDoTable')

    task = json.loads(event['body'])
    task['taskId'] = str(uuid.uuid4())

    table.put_item(Item=task)

    return {
        'statusCode': 200,
        'body': json.dumps({'message': 'Task added successfully', 'task': task})
    }
```

4. Click **Deploy**
5. Create a test event with:

```
{
  "body": "{\"taskName\":\"Learn AWS\",\"status\":\"Pending\"}"
}
```

6. Click **Test** → □ Confirm task is added.

Step 3: Create Lambda Function –==========GetTasks

1. Create new function: GetTasks
2. Use this code:

```
import boto3
import json

def lambda_handler(event, context):
    dynamodb = boto3.resource('dynamodb')
    table = dynamodb.Table('ToDoTable')

    response = table.scan()

    return {
        'statusCode': 200,
        'body': json.dumps(response['Items'])
    }
```

3. Deploy and test.

Step 4: Create Lambda Function – UpdateTask

```
import boto3
import json

def lambda_handler(event, context):
    dynamodb = boto3.resource('dynamodb')
    table = dynamodb.Table('ToDoTable')

    data = json.loads(event['body'])
    taskId = data['taskId']
    taskName = data['taskName']
    status = data['status']

    table.update_item(
        Key={'taskId': taskId},
        UpdateExpression="set taskName = :t, #s = :s",
        ExpressionAttributeValues={
            ':t': taskName,
            ':s': status
        },
```

```
        ExpressionAttributeNames={
            "#s": "status"
        },
        ReturnValues="UPDATED_NEW"
    )

    return {
        'statusCode': 200,
        'body': json.dumps({'message': 'Task updated successfully'})
    }
```

Deploy and test with:

```
{
  "body": "{\"taskId\":\"<paste-task-id>\", \"taskName\":\"Learn Lambda\", \"status\":\"Completed\"}"
}
```

Step 5: Create Lambda Function – DeleteTask

```
import boto3
import json

def lambda_handler(event, context):
    dynamodb = boto3.resource('dynamodb')
    table = dynamodb.Table('ToDoTable')

    taskId = event['queryStringParameters']['taskId']

    table.delete_item(Key={'taskId': taskId})

    return {
        'statusCode': 200,
        'body': json.dumps({'message': 'Task deleted successfully'})
    }
```

Deploy and test with query parameter: ?taskId=<your-task-id>

TASK 2: Setup API Gateway (CRUD)

Step 1: Create REST API

1. Go to **API Gateway** → Create **REST API**
2. Create **Resource**:
    - Resource Name: tasks
    - Resource Path: /tasks
3. Under /tasks:
    - **POST** → connect to AddTask
    - **GET** → connect to GetTasks
    - **PUT** → connect to UpdateTask
    - **DELETE** → connect to DeleteTask
4. **Deploy API** → Stage: prod
   Copy the Invoke URL (e.g., https://xyz123.execute-api.aws-region.amazonaws.com/prod/tasks)

TASK 3: Test the API with cURL or Postman

POST – Add Task

```
curl -X POST "https://<API-ID>.execute-api.<region>.amazonaws.com/prod/tasks" \
-H "Content-Type: application/json" \
-d '{"taskName": "Learn AWS", "status": "Pending"}'
```

GET – View All Tasks

```
curl -X GET "https://<API-ID>.execute-api.<region>.amazonaws.com/prod/tasks"
```

PUT – Update Task

```
curl -X PUT "https://<API-ID>.execute-api.<region>.amazonaws.com/prod/tasks" \
-H "Content-Type: application/json" \
-d '{"taskId": "abc123", "taskName": "Learn Lambda", "status": "Completed"}'
```

DELETE – Remove Task

```
curl -X DELETE "https://<API-ID>.execute-api.<region>.amazonaws.com/prod/tasks?taskId=abc123"
```

Troubleshooting Tips

Issue	Solution
403 Forbidden	Check IAM roles and API Gateway resource policies
500 Error	Use CloudWatch Logs, check for syntax/DynamoDB table name
Empty Response	Ensure task was added before trying GET or UPDATE
Update not working?	Make sure the correct taskId is used

Simple summary Table is given just take a look here

Feature	Lambda Function	API Method	HTTP Verb
Add Task	AddTask	/tasks	POST
View Tasks	GetTasks	/tasks	GET
Update Task	UpdateTask	/tasks	PUT
Delete Task	DeleteTask	/tasks	DELETE