

# Okuma: AWS Lambda

## Giriş

Basit bir sunucusuz uygulama oluşturalım, AWS Lambda kullanarak.

Bu uygulama, bazı metinler girebileceğiniz AWS Amplify üzerinde barındırılan bir html ön yüzüne sahip olacak. Formu gönderdiğinizde, girdiğiniz metnin büyük harfli ve tersine çevrilmiş bir yanıtını alacaksınız.

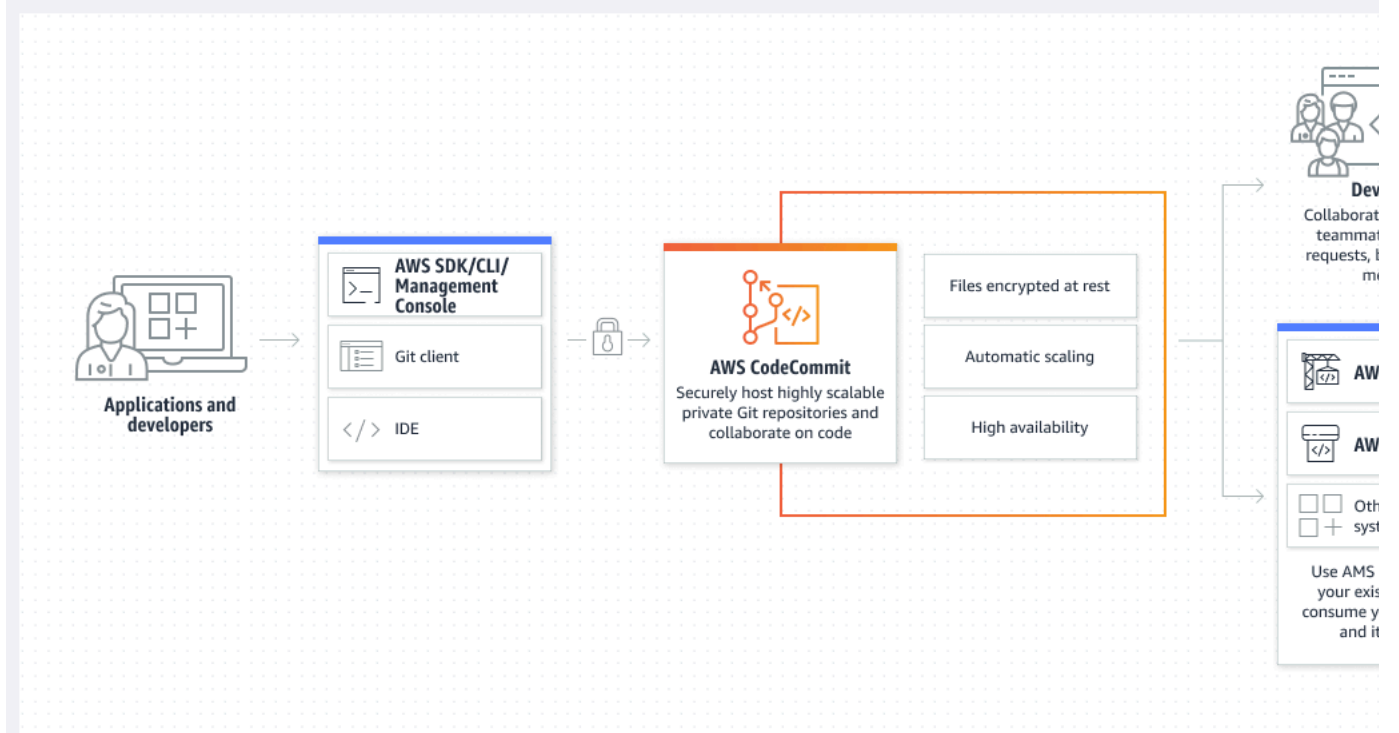
Büyük Harf ve Tersine Çevirme, zincirleme yeteneklerini göstermek için iki ayrı Lambda fonksiyonu olacak. Bu fonksiyonlara doğrudan erişmek yerine, istemci isteklerini kabul etmek ve nihai çıktıyı yanıtlamak için bir API Gateway kullanılacaktır.

Uygulama, **Büyük Harf** ve **Tersine Çevirme** adında iki ayrı Lambda fonksiyonu içerecektir. Bu fonksiyonlar, AWS Step Functions kullanılarak zincirlenecektir. Bu fonksiyonlara doğrudan erişmek yerine, istemci isteklerini kabul etmek ve nihai çıktıyı yanıtlamak için bir API Gateway kullanılacaktır.

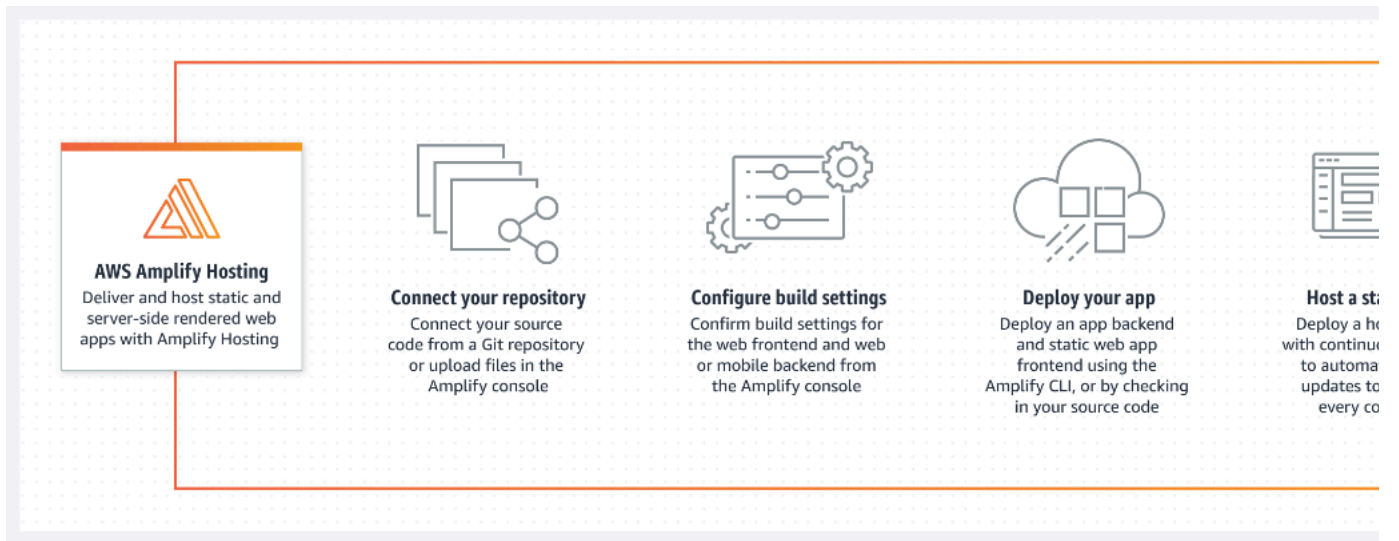
## Kullanılan Bileşenler:

- **AWS CodeCommit:** AWS CodeCommit, özel Git depolarını barındıran güvenli, yüksek ölçeklenebilir, tamamen yönetilen bir kaynak kontrol hizmetidir.

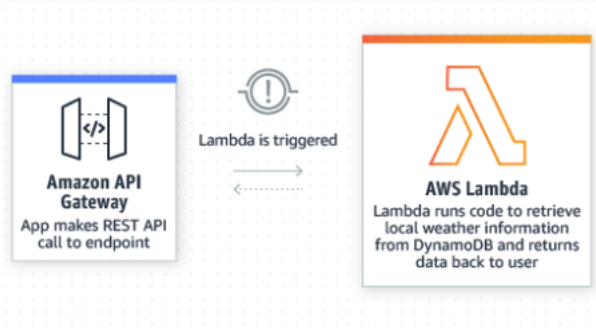
Git tabanlı bir hizmet olarak, CodeCommit çoğu sürüm kontrol ihtiyacına uygun bir şekilde tasarlanmıştır. Dosya boyutu, dosya türü ve depo boyutu üzerinde keyfi sınırlamalar yoktur.



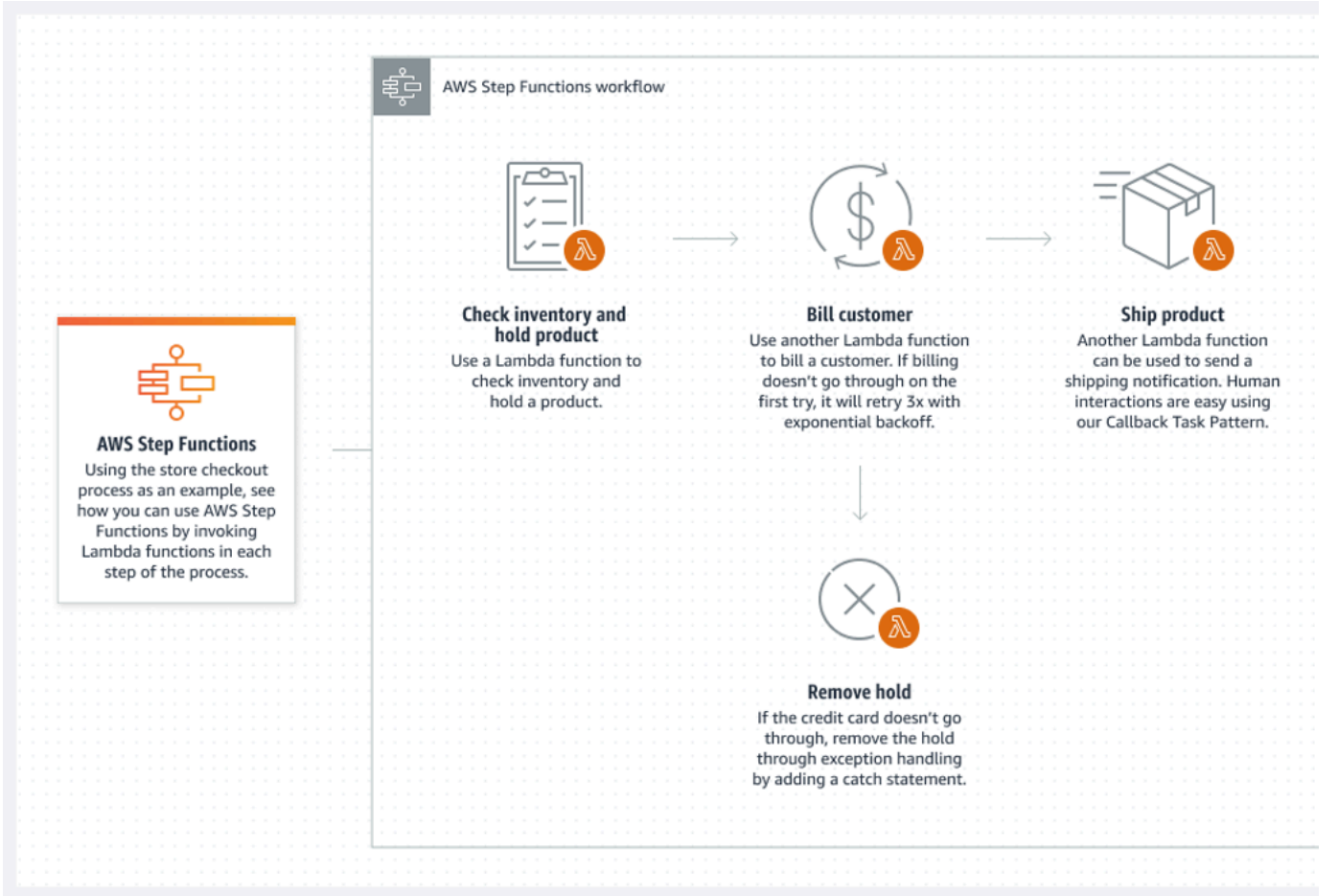
- **AWS Amplify:** AWS Amplify, ön uç web ve mobil geliştiricilerin AWS üzerinde tam yığın uygulamaları kolayca inşa etmelerini, dağıtmalarını ve barındırmalarını sağlayan eksiksiz bir çözümdür; kullanım durumları geliştikçe AWS hizmetlerinin genişliğinden yararlanma esnekliği sunar.



- **AWS Lambda:** AWS Lambda, sunucusuz, olay odaklı bir hesaplama hizmetidir; bu hizmet sayesinde sunucu tahsis etmeden veya yönetmeden hemen hemen her tür uygulama veya arka uç hizmeti için kod çalıştırabilirsiniz. Lambda'yı 200'den fazla AWS hizmetinden ve yazılım olarak hizmet (SaaS) uygulamasından tetikleyebilir ve yalnızca kullandığınız kadar ödeyebilirsiniz.



- **AWS Step Function:** AWS Step Function, geliştiricilerin AWS hizmetlerini kullanarak dağıtık uygulamalar inşa etmelerine, süreçleri otomatikleştirmelerine, mikro hizmetleri düzenlemelerine ve veri ile makine öğrenimi (ML) boru hatları oluşturmalarına yardımcı olan görsel bir iş akışı hizmetidir.

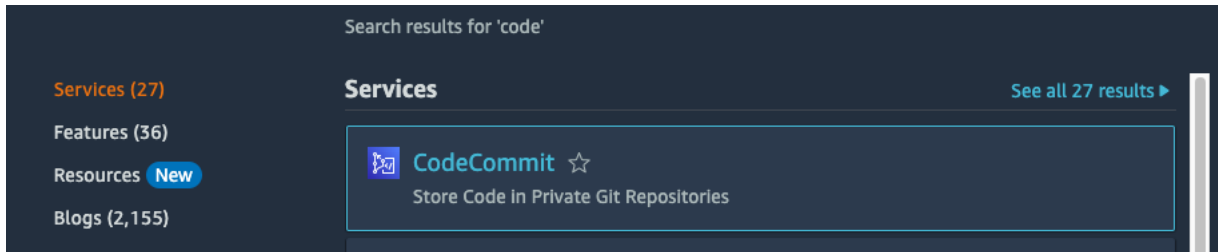


- **AWS API Gateway:** Amazon API Gateway, geliştiricilerin herhangi bir ölçekte API'ler oluşturmalarını, yayımlamalarını, sürdürmesini, izlemesini ve güvence altına almasını kolaylaştıran tamamen yönetilen bir hizmettir. API'ler, uygulamaların arka uç hizmetlerinizden veri, iş mantığı veya işlevsellığe erişmesi için "ön kapı" işlevi görür.

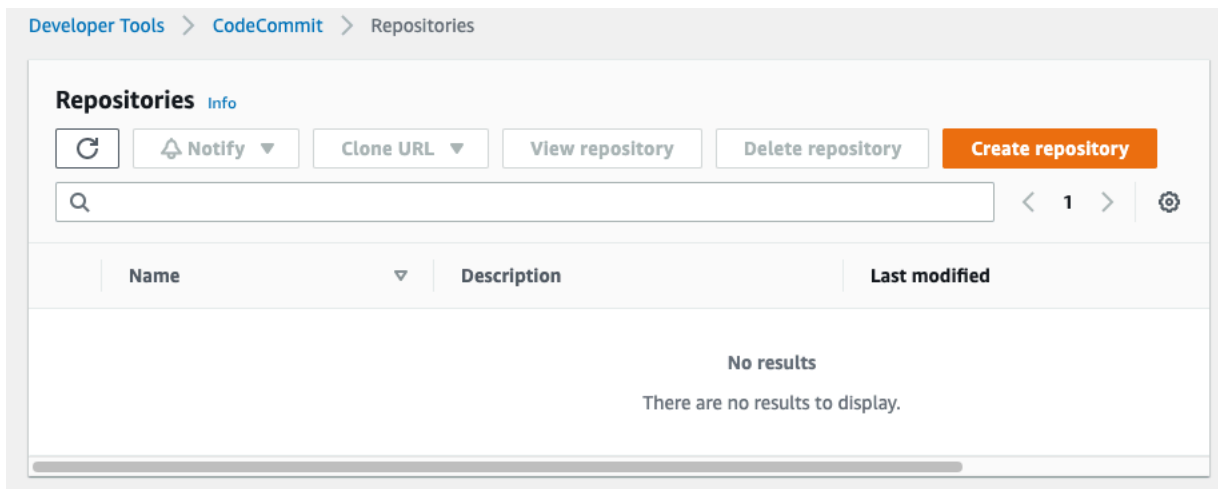


## Süreç

- Kod deposu olarak kullanabileceğiniz CodeCommit kaynağını tanımlayarak başlayalım.



- Boş bir depo ile başlayın. Depo oluştur'a tıklayın.



- Bir depo adı ve isteğe bağlı bir açıklama sağlayın.

## Create repository

Create a secure repository to store and share your code. Begin by typing a repository name and a description for your repository. Repository names are included in the URLs for that repository.

### Repository settings

Repository name

100 characters maximum. Other limits apply.

Description - *optional*

1,000 characters maximum

Tags

☐ Enable Amazon CodeGuru Reviewer for Java and Python - *optional*

Get recommendations to improve the quality of the Java and Python code for all pull requests in this repository.

A service-linked role will be created in IAM on your behalf if it does not exist.

Cancel

Create

3. Şimdi bu deponun ayrıntılarını alarak yerel ortamınıza klonlayın.

Success

Repository successfully created

Create a notification rule for this repository

Developer Tools > CodeCommit > Repositories > capitaliseAndReverseUI

capitaliseAndReverseUI

Clone URL ▲

Clone HTTPS

Clone SSH

Clone HTTPS (GRC)

▼ Connection steps

HTTPS | SSH | HTTPS (GRC)

4. Gerekli html kaynaklarını oluşturmak için depoyu bilgisayarınıza klonlayın.

```
$ git clone https://git-codecommit.eu-west-2.amazonaws.com/v1/repos/capitaliseAndReverseUI
Cloning into 'capitaliseAndReverseUI'...
Username for 'https://git-codecommit.eu-west-2.amazonaws.com': 
Password for 'https://[redacted]@git-codecommit.eu-west-2.amazonaws.com': 
warning: You appear to have cloned an empty repository.
```

5. Ardından basit bir html sayfası oluşturun (içinde gereken JavaScript ve CSS bölümleri olacak).

```
$ git add index.html
$ git status
On branch master

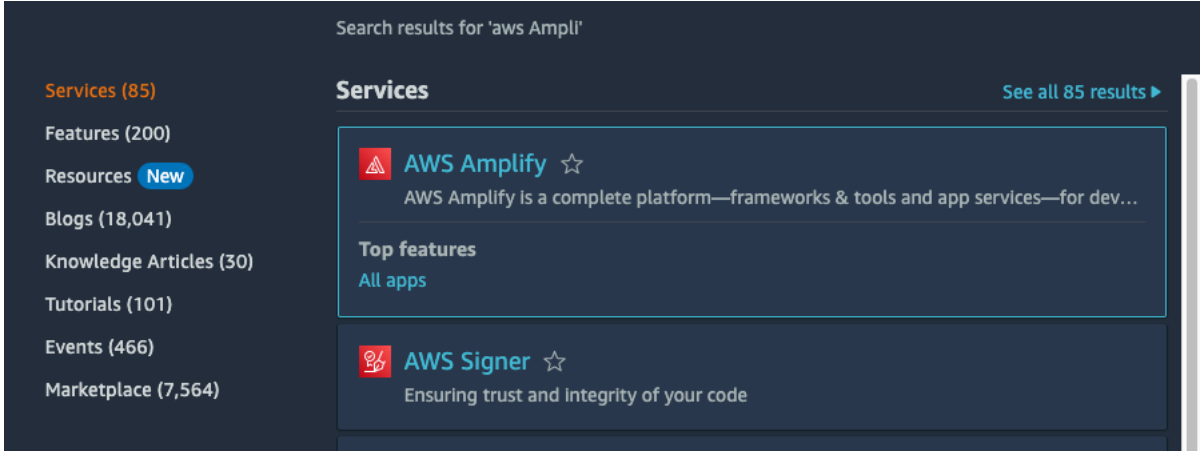
No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   index.html
```

6. Değişikliklerinizi kaydedin ve değişiklikleri AWS CodeCommit üzerindeki uzak depoya da itebilirsiniz.

```
$ git commit -m "index.html created which accepts a text value"
[master (root-commit) 14aa3ab] index.html created which accepts a text value
1 file changed, 49 insertions(+)
create mode 100644 index.html
```

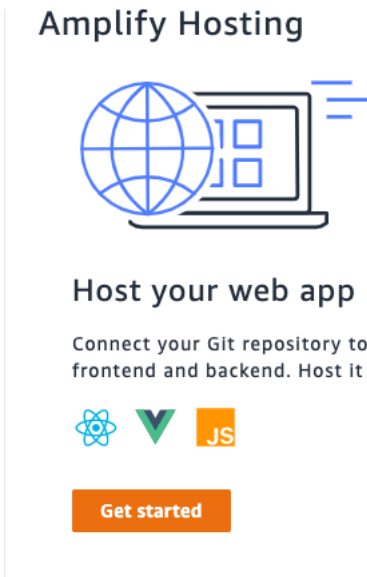
1. Şimdi statik içeriğinizi (HTML) barındırmak için AWS Amplify kaynağı oluşturun.



2. Kaynağı oluşturarak başlayın.



3. Web uygulamanızı barındırmayı seçin.



4. AWS CodeCommit'i seçin; burası yerel ortamınızdan depoya değişiklikler gönderdiğiniz yerdir.

# Get started with Amplify Hosting

Amplify Hosting is a fully managed hosting service for web apps. Connect your repository to build, deploy, and host your web app.

## From your existing code

Connect your source code from a Git repository or upload files to host a web app in minutes.

☐ GitHub



☐ Bitbucket



☐ GitLab



☒ AWS CodeCommit



☐ Deploy without Git provider



Amplify Hosting requires read-only access to your repository.

Cor

5. Şimdi ana dalı AWS Amplify ile bağlayacaksınız. Bu, ana dalda değişiklik yaptığınızda sürekli teslimat sağlayacaktır.

## Add repository branch

### AWS CodeCommit

✓ AWS CodeCommit authorization was successful.

#### Repository service provider



AWS CodeCommit

#### Recently updated repositories

If you don't see your repository below, please push a commit and then click the refresh button.

capitaliseAndReverseUI



#### Branch

Select a branch from your repository.

master



☐ Connecting a monorepo? Pick a folder.

Cancel

Previous

Next

6. Varsayılan yapı ayarlarını kabul edin.

# Build settings

## App build and test settings

### App name

Pick a name for your app.

Name cannot contain periods

### Build and test settings

We've auto-detected your app's build settings. Please ensure your build command and output folder (baseDirectory) are correctly detected.

```
1 | version: 1
2 | frontend:
3 |   phases:
4 |     # IMPORTANT - Please verify your build commands
5 |     build:
6 |       commands: []
7 |   artifacts:
8 |     # IMPORTANT - Please verify your build output directory
9 |     baseDirectory: /
10 |     files:
11 |       - '**/*'
12 |   cache:
13 |     paths: []
14 |
```

Build and test settings

Download

Edit

☒ Allow AWS Amplify to automatically deploy all files hosted in your project root directory

► Advanced settings

## IAM Role

### IAM service role

Amplify requires read-only access to your CodeCommit repository. To create custom roles go to the [IAM console](#).

- ☒ Create and use a new service role
- ☐ Use an existing service role

Cancel

Previous

Next

7. Süreci gözden geçirin ve tamamlayın.

## Review

### Repository details

Repository service

AWS CodeCommit

Repository

capitaliseAndReverseUI

Branch

master

Branch environment

Application root

### App settings

Edit

App name

capitaliseAndReverseUI

Build image

Using default image

Environment variables

None

Framework

Web

Build settings

Auto-detected settings will be used

Cancel

Previous

Save and deploy

8. Süreç tamamlanması biraz zaman alır (sağlama, oluşturma ve değişikliklerinizi dağıtma).

## capitaliseAndReverseUI

The app homepage lists all deployed frontend and backend environments.

► Learn how to get the most out of Amplify Hosting

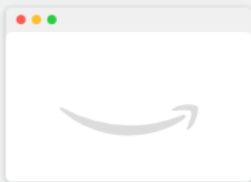
Hosting environments

Backend environments

This tab lists all connected branches, select a branch to view build details.

master

Continuous deploys set up ([Edit](#))



<https://master...amplifyapp.com>



Last deployment

23/11/2022, 22:05:05

Last commit

This is an autogenerated message | Auto-build | [AWS CodeCommit - master](#)

Pr

Di

9. Tamamlandığında, web uygulamanızı görmek için URL'yi ziyaret edebilirsiniz.

Submit

## Uygulamanızı GitHub Kullanarak AWS Amplify Üzerinde Dağıtma

1. **GitHub**'ı seçin ve **İleri**'ye tıklayın. Kod deponuz olarak kullanmak istediğiniz GitHub deposunu tanımlamanız gerekecek.

aws

Services

Search

[Alt+S]

AWS Amplify

All apps / Create new app

1 Choose source code provider

2 Add repository and branch

3 App settings

4 Review

Start building with Amplify

Amplify provides a fully-managed web hosting experience and a backend building service to build fullstack apps. If you need a starter project, please visit the [docs](#)

Deploy your app

To deploy an app from a Git provider, select one of the options below:

GitHub

BitBucket

CodeCommit

Amplify requires read-only access to your repository.

To deploy an app manually, select "Deploy without Git"

Deploy without Git

Looking to build an app with our Gen 1 tools (Amplify Studio/Amplify CLI)? [Create an app with Gen 1](#)

*Not: GitHub'ı seçtikten sonra, erişim izni vermeniz istenecektir.*

2. Dağıtmak istediğiniz projeyi seçin, ardından çalışma dalını (ana/ust) GitHub ile bağlayın. Bu yapılandırma, çalışma dalına değişiklik gönderdiğinizde güncellemeleri otomatik olarak dağıtacak şekilde sürekli teslimat sağlayacaktır.



Choose source code provider



Add repository and branch



App settings




Review

## Add repository and branch

If you don't see your repository in the dropdown above, ensure the Amplify GitHub App has permission and click the refresh button.

☐ My app is a monorepo

Not: Projenize göre Ön uç oluşturma komutunu ve oluşturma dizinini belirtin.

 Services  [Alt+S]

All apps / Create new app

✓ Choose source code provider

✓ Add repository and branch

**App settings**

4 Review

### App settings

App name

### Build settings

Your build settings have been detected automatically, please verify your "Frontend build command" and "Backend build command" in your build file.

Auto-detected frameworks

Frontend build command

Edit YML file

☐ Password protect my site

### Advanced settings

3. Süreci gözden geçirin ve tamamlayın.

aws Services Search [Alt+S]

All apps / Create new app

- ✓ Choose source code provider
- ✓ Add repository and branch
- ✓ App settings
- Review

github

Branch

master

### App settings

App name

vftvk-Simple-Interest-Calculator

Framework

None

### Advanced settings

Build image

Using default image

Live package updates

Server-Side Rendering (SSR) deployment

Disabled

**First-time account setup required**  
Amplify needs to run a one-time setup for this account and region before it can deploy resources in the account.

Süreç tamamlanması biraz zaman alır (sağlama, oluşturma ve değişikliklerinizi dağıtma).

aws Services Search [Alt+S]

All apps / vftvk-Simple-Interest-Calculator / Overview

vftvk-Simple-Interest-Calculator <<

- Overview
- Hosting
- App settings

### vftvk-Simple-Interest-Calculator

App ID: d39fzp1u1jfw1

Production branch

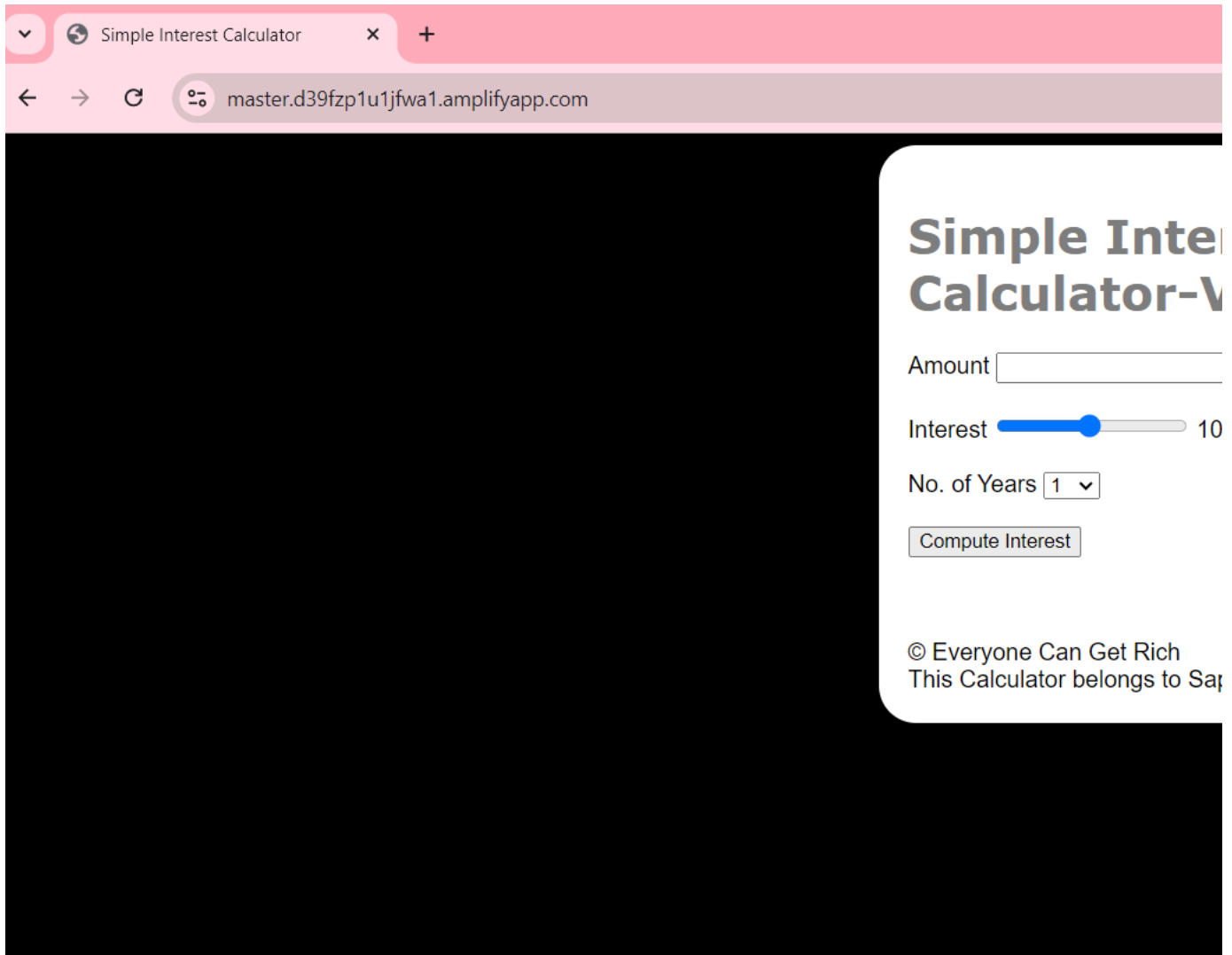
**master** >  
Deployed ✓

Domain	Updated	Last commit	Repository
<a href="https://master.d39fzp1u1jfw1.amplifyapp.com">https://master.d39fzp1u1jfw1.amplifyapp.com</a>	8/21/2024, 3:17 PM	Auto-build	vftvk-Simp

Other branches 0 Search...

No other

4. Tamamlandığında, sağlanan URL'yi ziyaret ederek web uygulamanızı görebilirsiniz.



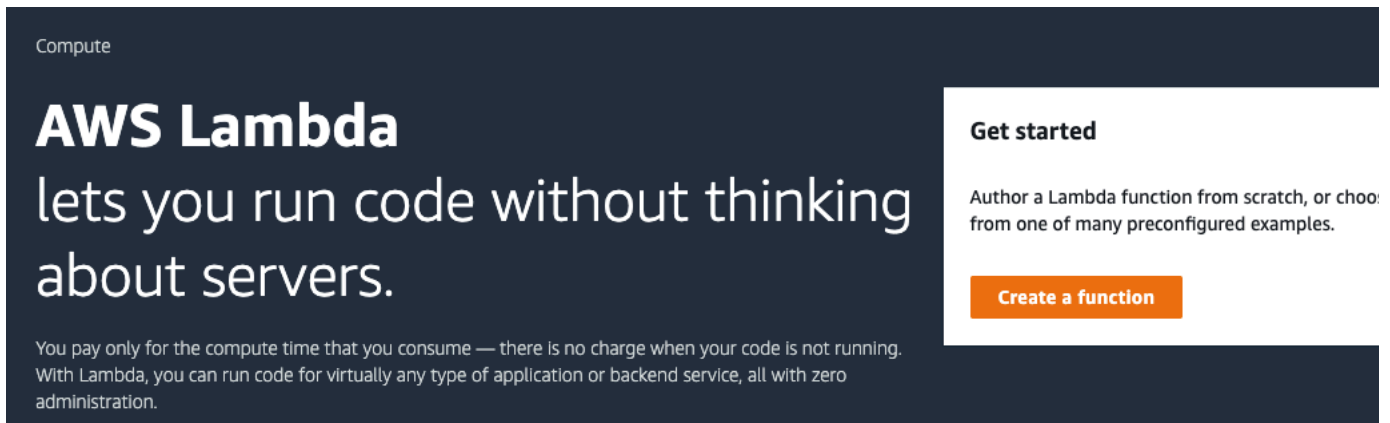
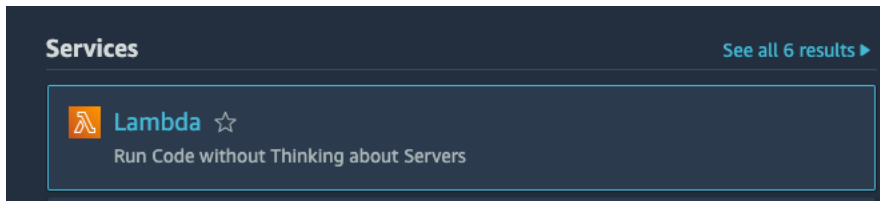
*Not: Dağıtım başarısız olursa, sol panelden Barındırma'ya tıklayın ve oluşturma ayarlarını seçin ve YML dosyasını güncelleyin.*

Ancak bu uygulama tamamlanmış değil, giriş dizesinin büyük harf yapma ve ters çevirme işlemi için henüz arka ucu oluşturmanız gerekiyor.

## AWS Lambda Fonksiyonları Oluşturma

### Büyük Harf Fonksiyonu:

1. İlk olarak, girdi metnini büyük harfe çeviren AWS Lambda fonksiyonunu tanımlayarak başlarsınız.



2. Fonksiyon adını ve çalışma zamanını belirtin. Bunun için Python 3.9'u seçin.

# Create function [Info](#)

AWS Serverless Application Repository applications have moved to [Create application](#).

## Author from scratch



Start with a simple Hello World example.

## Use a blueprint



Build a Lambda application from sample code and configuration presets for common use cases.

### Basic information

#### Function name

Enter a name that describes the purpose of your function.

Use only letters, numbers, hyphens, or underscores with no spaces.

#### Runtime [Info](#)

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

#### 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](#)


► [Advanced settings](#)


3. Fonksiyonu tanımlamak bu şekilde görünecektir:

# capitaliseFunc

▼ Function overview [Info](#)

 **capitaliseFunc**

 Layers (0)

 API Gateway

[+ Add trigger](#)

[Code](#) | [Test](#) | [Monitor](#) | [Configuration](#) | [Aliases](#) | [Versions](#)

## Code source [Info](#)

File Edit Find View Go Tools Window [Test](#) [Deploy](#)

Go to Anything (% P)

Environment

- capitaliseFunc - /
  - lambda\_function.py

```
1 import json
2
3 def lambda_handler(event, context):
4     input_text = str(event['inputText'])
5     capitalised_input_text = input_text.upper()
6     return {"inputText": capitalised_input_text}
```

4. Yazdığınız kod oldukça basittir, çünkü girdi metnini gövde parçası olarak alır (bu bir HTTP POST fonksiyonudur). Ve nesneyi tekrar büyük harfli değeriyle girdi metni olarak döndürür (bunu ters fonksiyona bağlayabilirsiniz).

```
import json
def lambda_handler(event, context):
    input_text = str(event['inputText'])
    capitalised_input_text = input_text.upper()
    return {"inputText": capitalised_input_text}
```

### Configure test event ×

A test event is a JSON object that mocks the structure of requests emitted by AWS services to invoke a Lambda function. Use it to see the function's invocation result.

To invoke your function without saving an event, modify the event, then choose Test. Lambda uses the modified event to invoke your function, but does not overwrite the original event until you choose Save changes.

Test event action

☐ Create new event

☒ Edit saved event

Event name

capitaliseText ▼ [↺](#) [Delete](#)

#### Event JSON [Format JSON](#)

```
1 {
2   "inputText": "value1"
3 }
```

5. Fonksiyonunuzu dağıttıktan sonra, onu test edebilir ve aşağıdaki sonucu görebilirsiniz.

The screenshot shows the AWS Lambda console interface. At the top, there are tabs for 'Tools', 'Window', 'Test', and 'Deploy'. Below these, there are tabs for 'lambda\_function' and 'Execution result:'. The 'Execution results' tab is active, showing a status of 'Success'. The 'Test Event Name' is 'capitaliseText'. The 'Response' is a JSON object: `{ "inputText": "VALUE1" }`. The 'Function Logs' section shows the following log entries:   
START RequestId: b56e0c53-6fd2-4080-9800-ac42efdb43b4 Version: \$LATEST   
END RequestId: b56e0c53-6fd2-4080-9800-ac42efdb43b4   
REPORT RequestId: b56e0c53-6fd2-4080-9800-ac42efdb43b4 Duration: 1.29 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 36 MB   
The 'Request ID' is 'b56e0c53-6fd2-4080-9800-ac42efdb43b4'.

6. Benzer şekilde, ters fonksiyonu oluşturunuz.

The screenshot shows the AWS Lambda console interface. At the top, there are tabs for 'lambda\_function' and 'Execution results:'. The 'Execution results' tab is active, showing a status of 'Success'. The code for the function is as follows:   
1 import json   
2   
3 def lambda\_handler(event, context):   
4 input\_text = str(event['inputText'])   
5 reversed\_input\_text = input\_text[::-1]   
6 return {"inputText": reversed\_input\_text}   
7   
The function is named 'reverseText'.

7. Fonksiyonu dağıtın ve test edin.

The screenshot shows the AWS Lambda console interface. At the top, there are tabs for 'lambda\_function' and 'Execution result:'. The 'Execution results' tab is active, showing a status of 'Success'. The 'Test Event Name' is 'reverseText'. The 'Response' is a JSON object: `{ "inputText": "1eulav" }`.

8. Artık iki fonksiyon tanımlayıp oluşturduğunuza göre, bunları StepFunctions kullanarak birbirine bağlayabilirsiniz.

The screenshot shows the AWS Step Functions console. The header 'Step Functions' is displayed with a star icon and the text 'Coordinate Distributed Applications'. Below this, there is a section titled 'Top features' with links to 'State Machines', 'Activities', 'Getting started', 'Sample projects', and 'Feature spotlight'.

9. Bir durum makinesi oluşturarak başlayın.

The screenshot shows the AWS Step Functions console. The header 'State machines (1)' is displayed. Below this, there is a search bar with the text 'Search for state machines' and a dropdown menu with the text 'Any type'. The table below shows the following columns: 'Name', 'Type', 'Creation date', 'Status', 'Logs', 'Total', 'Running', and 'Succeeded'.

10. Kolaylık sağlamak için iş akışını görsel olarak tasarlamayı seçebilir ve fonksiyonlarınızı senkronize çalışması için Express kullanabilirsiniz.

## Choose authoring method

### Design your workflow visually

Drag and drop your workflow together with Step Functions Workflow Studio. **New**

### Write your workflow in code

Author your workflow using Amazon States Language. You can generate code snippets to easily build out your workflow steps.

### Run a sample pro

Deploy and run a fully minutes using CloudF

## Type

### ☐ Standard

Durable, checkpointed workflows for machine learning, order fulfillment, IT/DevOps automation, ETL jobs, and other long-duration workloads.

### ☒ Express

Event-driven workflows for streaming data processing, microservice ingestion, mobile backends, and other short duration, high-event

► **Help me decide**

## Invoke Capitalise

### Configuration

### Input

### Output

### Error handling

#### State name

Invoke Capitalise

#### API

Lambda: Invoke

#### Integration type [Info](#)

The type of service integration to use. [Learn more](#)

Optimized

#### API Parameters

☒ Edit

##### Function name

The Lambda function to invoke

Enter function name

arn:aws:lambda:eu-west-2

:function:capitaliseFun

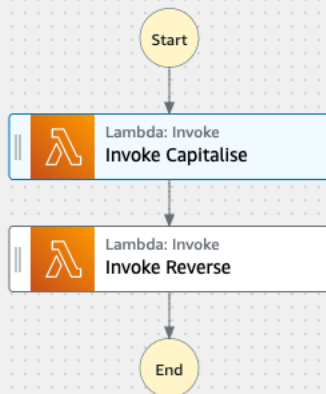
Must be a valid function name.

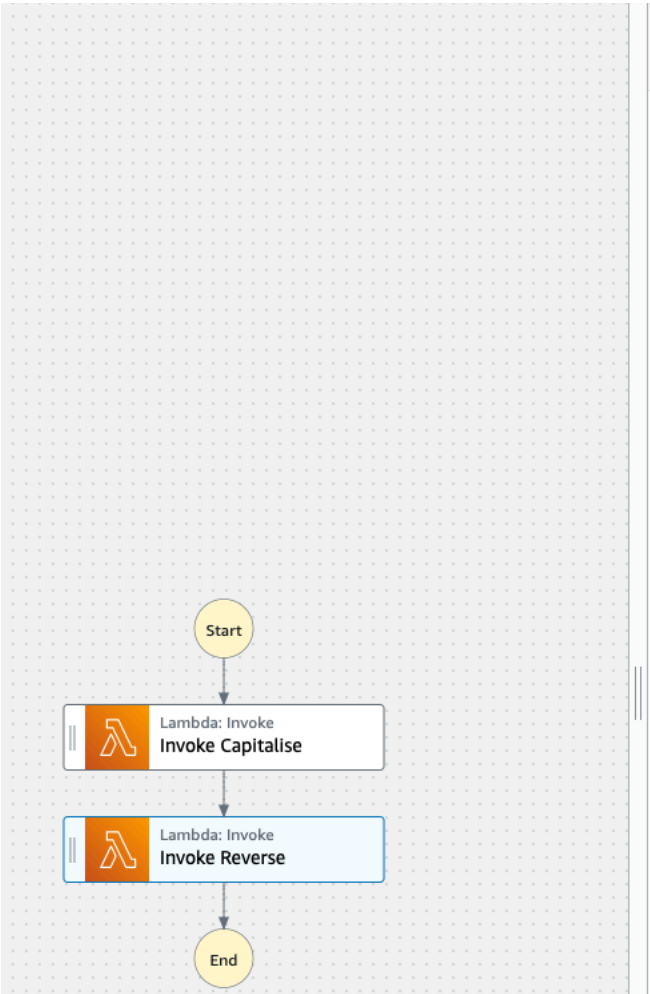
[View function](#)

#### Payload

The JSON that you want to provide to your Lambda function.

Use state input as payload





# Invoke Reverse

Configuration | Input | Output | Error handling

## State name

Invoke Reverse

## API

Lambda: Invoke

## Integration type

The type of service integration to use. [Learn more](#)

Optimized

## API Parameters

☐ Edit as JSON

### Function name

The Lambda function to invoke

Enter function name

arn:aws:lambda:eu-west-2 :function:reverseFunc:\$LATEST

Must be a valid function name.

[View function](#)

## Payload

The JSON that you want to provide to your Lambda function.

Use state input as payload

# Edit CapitaliseAndReverseStateMachine

## Definition

Define your workflow using [Amazon States Language](#). Test your data flow with the new [Data Flow Simulator](#).

Generate code snippet ▼

Format JSON

```
1 {
2   "Comment": "A description of my state machine",
3   "StartAt": "Invoke Capitalise",
4   "States": {
5     "Invoke Capitalise": {
6       "Type": "Task",
7       "Resource": "arn:aws:states:::lambda:invoke",
8       "OutputPath": "$$.Payload",
9       "Parameters": {
10        "Payload.$": "$",
11        "FunctionName": "arn:aws:lambda:eu-west-2: :function:capitaliseFunc:$LATEST"
12      },
13       "Retry": [
14         {
15           "ErrorEquals": [
16             "Lambda.ServiceException",
17             "Lambda.AWSLambdaException",
18             "Lambda.SdkClientException",
19             "Lambda.TooManyRequestsException"
20           ],
21           "IntervalSeconds": 2,
22           "MaxAttempts": 6,
23           "BackoffRate": 2
24         }
25       ],
26       "Next": "Invoke Reverse"
27     },
28     "Invoke Reverse": {
29       "Type": "Task",
30       "Resource": "arn:aws:states:::lambda:invoke",
31       "OutputPath": "$$.Payload",
32       "Parameters": {
33        "Payload.$": "$",
34        "FunctionName": "arn:aws:lambda:eu-west-2: :function:reverseFunc:$LATEST"
35      },
36       "Retry": [
37         {
```



11. Durum makinenizi test etmek için Yeni yürütme seçeneğine tıklayın..

Edit state machine

Export

New execution

## Start execution

Start an execution using the latest definition of the state machine. [Learn more](#)

Name - *optional*

testint\_state\_machine

Input - *optional*

Enter input values for this execution in JSON format

Format JSON

Export

Import

```
1 {"inputText": "this is an example of anagram radar"}
```

Execution: testing\_state\_machine:d0f5e86c-20c6-45b9-985b-eadd2dc30b03

Details

Execution input and output

Definition

Input

```
1 {
2   "inputText": "this is an example of anagram radar"
3 }
```

Output

```
1 {
2   "inputText": "RADAR MARC
3 }
```

## API Geçidi Oluşturun

Search results for 'API Gate'

### Services

[See all 37 results](#)



API Gateway ☆

Build, Deploy and Manage APIs

## 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



## Choose the protocol

Select whether you would like to create a REST API or a WebSocket API.

☒ REST ☐ WebSocket

## Create new API

In Amazon API Gateway, a REST API refers to a collection of resources and methods that can be invoked through HTTPS endpoints.

☒ New API ☐ Import from Swagger or Open API 3 ☐ Example API

## Settings

Choose a friendly name and description for your API.

API name\*

Description

Endpoint Type  ⓘ

\* Required

Create

APIs

Custom Domain Names

VPC Links

API: capitaliseAndR...

Resources

Stages

Authorizers

Resources

Actions

/ Methods

### RESOURCE ACTIONS

Create Method  
Create Resource  
Enable CORS  
Edit Resource Documentation

### API ACTIONS

Deploy API  
Import API  
Edit API Documentation  
Delete API

Resources

Actions

## New Child Resource

Use this page to create a new child resource for your resource. ⓘ

Configure as ☒ proxy resource ⓘ

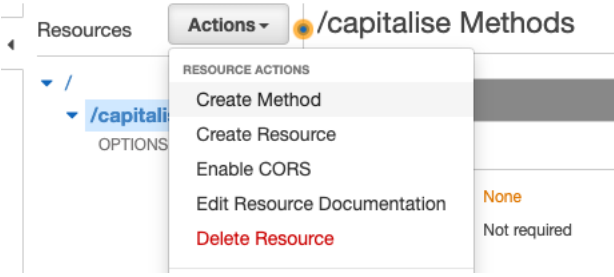
Resource Name\*

Resource Path\*

You can add path parameters using brackets. For example, the resource path {username} called 'username'. Configuring /{proxy+} as a proxy resource catches all requests to its : works for a GET request to /foo. To handle requests to /, add a new ANY method on the

Enable API Gateway CORS ☐ ⓘ

\* Required



Provide information about the target backend that this method will call and whether the incoming request data should be modified.

- Integration type**
- ☐ Lambda Function ⓘ
  - ☐ HTTP ⓘ
  - ☐ Mock ⓘ
  - ☒ AWS Service ⓘ
  - ☐ VPC Link ⓘ

**AWS Region** eu-west-2 ⓘ

**AWS Service** Step Functions ⓘ

**AWS Subdomain** ⓘ

**HTTP method** POST ⓘ

**Action** StartSyncExecution ⓘ

**Execution role** arn:aws:iam:: /APIGatewayToStepFunctions ⓘ

**Credentials cache** Do not add caller credentials to cache key ⓘ

**Content Handling** Passthrough ⓘ ⓘ

**Use Default Timeout** ☒ ⓘ

## ▼ Mapping Templates ⓘ

- Request body passthrough**
- ☐ When no template matches the request Content-Type header ⓘ
  - ☐ When there are no templates defined (recommended) ⓘ
  - ☒ Never ⓘ

Content-Type	
application/json	⊖

+ Add mapping template

application/json

Generate template:

```
1 #set($input = $input.json('$'))
2 {
3   "input": "$util.escapeJavaScript($input)",
4   "stateMachineArn": "arn:aws:states:eu-west-2::stateMachine
   :CapitaliseAndReverseStateMachine"
5 }
```

Sonra Aşama'yı tanımlarsınız. Aşama, API'nin bir anlık görüntüsü olan bir dağıtıma adlandırılmış bir referanstır. Belirli bir dağıtımı yönetmek ve optimize etmek için Aşama kullanırsınız. Örneğin, önbellekleme etkinleştirmek, istek sınırlamasını özelleştirmek, günlüğü yapılandırmak, aşama değişkenlerini tanımlamak veya test için bir kanarya sürümü eklemek için Aşama ayarlarını yapılandırabilirsiniz.

Invoke URL: <https://execute-api.eu-west-2.amazonaws.com/prod>

Settings

Logs/Tracing

Stage Variables

SDK Generation

Export

Deployment History

Documentation History

Canary

Cache Settings

Enable API cache ☐

Default Method Throttling

Choose the default throttling level for the methods in this stage. Each method in this stage will respect these rate and burst settings. Your current account level throttles requests per second with a burst of 5000 requests. [Read more about API Gateway throttling](#)

Enable throttling ☒ ⓘ

Rate  requests per second

Burst  requests

Web Application Firewall (WAF) [Learn more.](#)

Select the Web ACL to be applied to this stage.

Web ACL  [Create Web ACL](#)

Client Certificate

Select the client certificate that API Gateway will use to call your integration endpoints in this stage.

Certificate

SDK'yı oluşturun, böylece üretilen kodu web uygulamanızda kullanabilir ve bu API Geçidi'ni çağırabilirsiniz.

Settings

Logs/Tracing

Stage Variables

SDK Generation

Export

Deployment History

Documentation History

Canary

Choose a platform and provide the settings for the SDK you will generate.

Platform\*

\* Required

Sonra üretilen JavaScript kodunu aşağıdaki gibi çıkartırsınız:

```
capitaliseAndReverseUI
├── lib
│   ├── apiGatewayCore
│   │   ├── apiGatewayClient.js
│   │   ├── sigV4Client.js
│   │   ├── simpleHttpClient.js
│   │   └── utils.js
│   ├── axios / dist
│   │   └── axios.standalone.js
│   ├── CryptoJS
│   │   └── components
│   │       ├── enc-base64.js
│   │       └── hmac.js
│   ├── rollups
│   │   ├── hmac-sha256.js
│   │   └── sha256.js
│   ├── url-template
│   │   └── url-template.js
│   ├── apigClient.js
│   └── index.html
```

Ve nihayet API'yi dağıtırsınız (AWS API Geçidi bölümünde geri dönün).

Deploy API

Choose a stage where your API will be deployed. For example, a test version of your API could be deployed to a stage named beta.

Deployment stage	[New Stage] ▼
Stage name*	prod
Stage description	
Deployment description	

Cancel

Deploy

## Ön Uç Tamamla

Son HTML'iniz aşağıdaki gibi görünecektir; çıktınızı görüntülemek için bir alan eklediğinizi unutmayın.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Reverse and Capitalise with AWS Lambda</title>
    <style>
      body { font-family: Verdana; text-align: center; }
      form { max-width: 500px; margin: 50px auto; padding: 30px 20px; box-shadow: 2px 5px 10px rgba(0, 0, 0, 0.5); }
      .form-control { text-align: left; margin-bottom: 25px; }
      .form-control input { padding: 10px; display: block; width: 95%; }
    </style>
  </head>
  <body>
    <form id="form" onsubmit="callLambdaFunction(); return false;">
      <div class="form-control">
        <input type="text" id="inputText" placeholder="Enter some text here" />
      </div>
      <div class="form-control">
        <button type="submit" value="submit">Submit</button>
      </div>
      <div class="form-control">
        <input type="text" readonly id="outputText" placeholder="Output will appear here" />
      </div>
    </form>
    <script type="text/javascript" src="lib/axios/dist/axios.standalone.js"></script>
    <script type="text/javascript" src="lib/CryptoJS/rollups/hmac-sha256.js"></script>
    <script type="text/javascript" src="lib/CryptoJS/rollups/sha256.js"></script>
    <script type="text/javascript" src="lib/CryptoJS/components/hmac.js"></script>
    <script type="text/javascript" src="lib/CryptoJS/components/enc-base64.js"></script>
    <script type="text/javascript" src="lib/url-template/url-template.js"></script>
    <script type="text/javascript" src="lib/apiGatewayCore/sigV4Client.js"></script>
    <script type="text/javascript" src="lib/apiGatewayCore/apiGatewayClient.js"></script>
    <script type="text/javascript" src="lib/apiGatewayCore/simpleHttpClient.js"></script>
    <script type="text/javascript" src="lib/apiGatewayCore/utills.js"></script>
    <script type="text/javascript" src="apigClient.js"></script>
    <script type="text/javascript">
      function callLambdaFunction() {
        try {
          var inputTextValue = document.getElementById("inputText").value;
          var apigClient = apigClientFactory.newClient();
          var params = {};
          var body = { inputText: inputTextValue };
          apigClient.capitaliseandreversePost(params, body)
            .then(function (result) {
              document.getElementById("outputText").value = JSON.parse(result.data.output).inputText;
            })
            .catch(function (result) {
              console.log(result);
            });
        } catch (error) {
          console.log(error);
        }
        return false;
      }
    </script>
  </body>
</html>
```

Değişiklikleri AWS CodeCommit deposuna kaydedip gönderdikten sonra, dağıtılmasını bekleyin.

## capitaliseAndReverseUI

The app homepage lists all deployed frontend and backend environments.

► Learn how to get the most out of Amplify Hosting

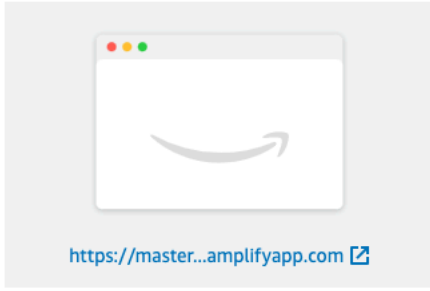
Hosting environments

Backend environments

This tab lists all connected branches, select a branch to view build details.

### master

Continuous deploys set up ([Edit](#))



Last deployment  
24/11/2022, 00:39:02

Last commit  
Please visit AWS CodeCommit Co... |  
74aedc0 | [AWS CodeCommit - master](#)

Previews  
Disabled

Artık AWS Amplify tarafından sağlanan URL'yi ziyaret ederek web uygulamanızı test edebilirsiniz.

## Sonuç

AWS tarafından sağlanan hizmetler, özellikle Lambda etrafında, hem ön hem de arka uç sağlayan sofistike uygulamalar oluşturmak için kullanılabilir. Ve uygulamanız için kod deposundan sunucusuz bir uygulama olarak dağıtılana kadar tüm ekosistemi inşa edebilirsiniz.

## Değişiklik Günlüğü

Tarih	Sürüm	Değiştiren	Değişiklik Açıklaması
21-08-2024	0.1	Manvi Gupta	Talimatlar güncellendi



**Skills** Network