Lab 10
Trying out AWS

CSCI2720 Building Web Applications

Dr. Chuck-jee Chau
chuckjee@cse.cuhk.edu.hk

Agenda

- create-react-app
- GitHub repo
- Hosting on AWS Amplify
- Building an API on AWS Lambda
- Linking everything

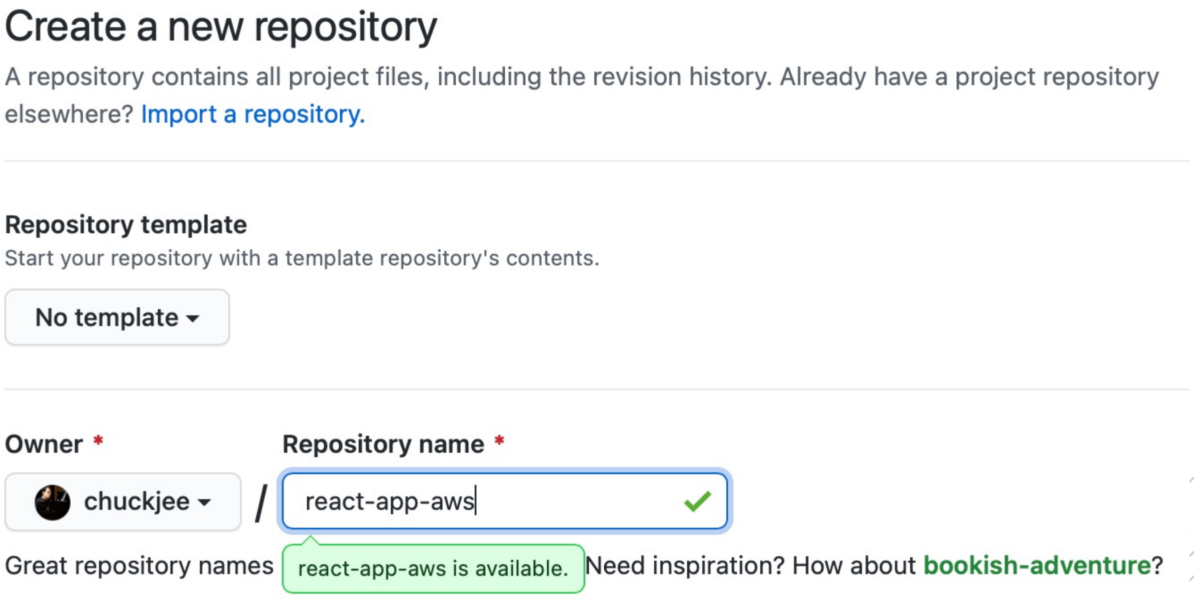
Deploying the React app in AWS

- Rather than hosting on your own premises, a growing trend is to use serverless platforms like Amazon AWS for web apps
- We will use the package ***AWS Amplify*** to host our app (while there are more possibilities!)
- The code could be hosted on an online git repo (e.g. GitHub)
- First, create a new React app on your computer or CSCI2720 VM using npx

```
npx create-react-app react-app-aws
```

Pushing the local repo to GitHub

- You need to get ready with your GitHub account
  - Open one if you don't have it yet at www.github.com
- Create a new GitHub repo for your app, e.g. *react-app-aws*
  - *https://github.com/new*



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Repository template

Start your repository with a template repository's contents.

Owner * chuckjee ▾ / **Repository name *** react-app-aws ✓

react-app-aws is available.

Great repository names are short and memorable. Need inspiration? How about **bookish-adventure**?

Pushing the local repo

- Initialize git and push the app to your new GitHub repo (*observe the lines!*)
  1. `cd react-app-aws ⏎`
  2. `git init ⏎`
  3. `git remote add origin ssh://git@ssh.github.com:443/username/reponame.git ⏎`
     *(of course, it should be your **username** and **repo** name!)*
  4. `git add . ⏎`
  5. `git commit -m "initial commit" ⏎`
  6. `git push origin master ⏎`
- If you see the connection being refused in the last step, refer to the next slide
- You can also see the new repo in the web interface
  - Verify that and then we will access this repo from AWS

Allowing key access to GitHub

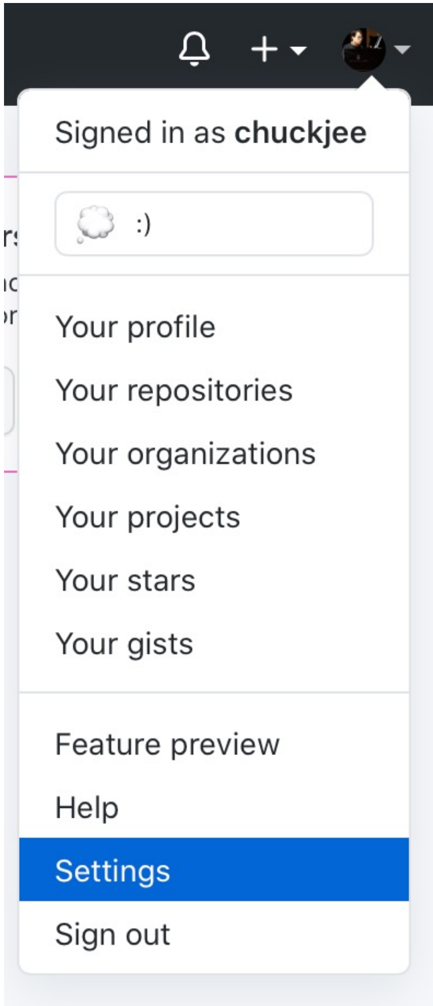
- For every user on every client machine, you can generate an SSH key with this command in Terminal:

  ```
  ssh-keygen
  ```

  - *(It is possible to use a variety of algorithms, but the default is fine too)*
  - Accept all the defaults and you will have a private key in **`~/.ssh/id_rsa`** and public key in **`~/.ssh/id_rsa.pub`**
  - *See: https://www.ssh.com/ssh/keygen/*
- Then, you can view the contents of the public key and copy it to GitHub

  ```
  cat ~/.ssh/id_rsa.pub
  ```

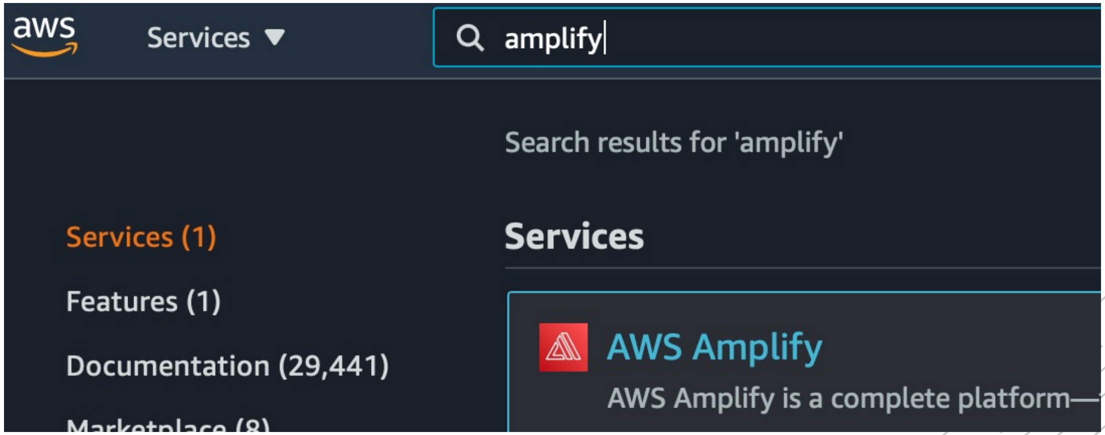
  - Go to ***Settings*** in GitHub
  - Choose ***SSH and GPG keys*** in the menu
  - Choose ***New SSH key***, and then paste the contents into Key, and set any Title you like
- ***Note:*** If it's not working for you, read this: *https://docs.github.com/en/github/authenticating-to-github/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent*



Signed in as **chuckjee**

Setting up AWS Amplify

- You should have received an email about our AWS Classroom in your @link email
  - Please accept the invitation, then you can *Go to classroom*
- Usually, you can log in to your account in AWS https://www.awseducate.com/signin/SiteLogin
  - There should be a link in the top under "My Classes"
  - You should be able to see your AWS account status
  - Find the button for ***AWS Console***
- In AWS, look for the service ***Amplify***
  - ***Note:*** For AWS Educate starter accounts, you must be using the us-east-1 region

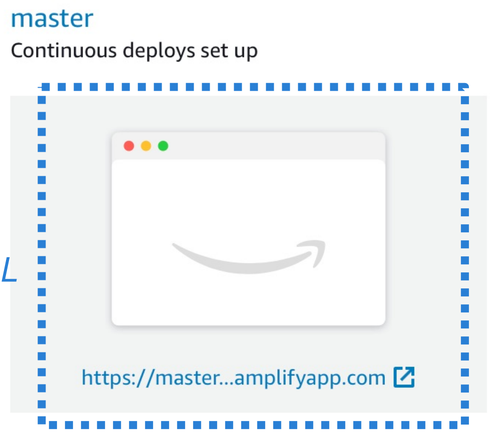


Search results for 'amplify'

AWS Amplify is a complete platform—

Setting up AWS Amplify

- Get started by choosing the button under "Deliver"
  - Or if you have used Amplify already, choose "Connect app"
- Choose GitHub, and authorize properly
- Then choose the *react-app-aws* repo in the list
- And go on with the defaults until you can choose "***Save and deploy***"
  - You should see your app being built and then a link is given to visit the app
  - *It would take a while…*

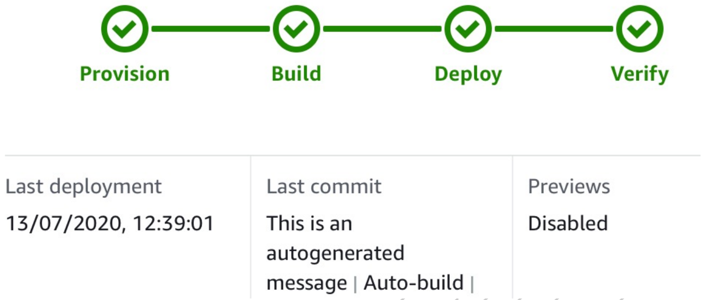


master

Continuous deploys set up

https://master...amplifyapp.com

Your Amplify app URL



Last deployment

13/07/2020, 12:39:01

Last commit

This is an autogenerated message | Auto-build |

Previews

Disabled

React on Amplify

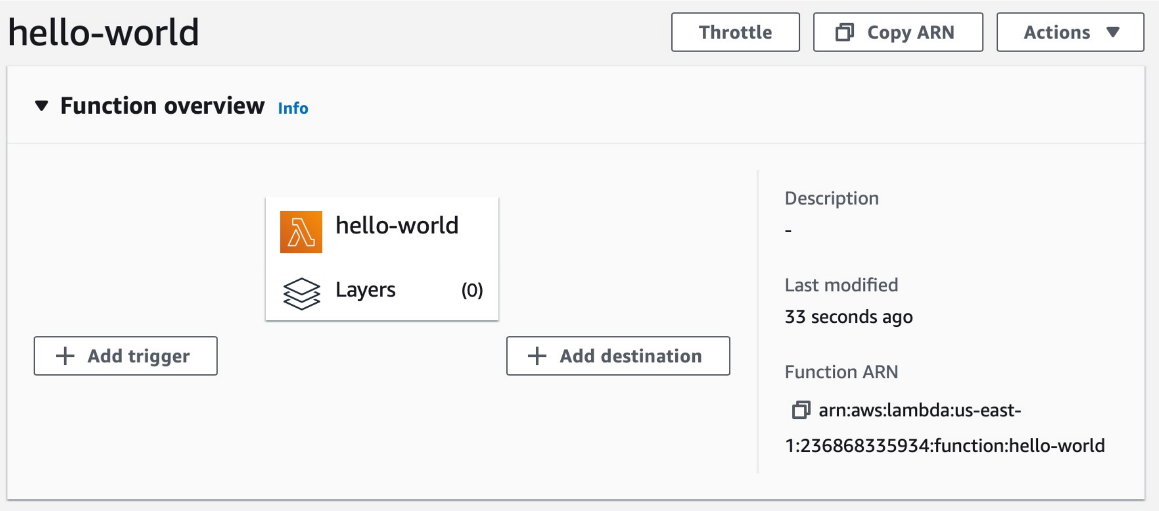
- All your code updates in the GitHub repo will automatically trigger a rebuild
- Recommended workflow:
  - Try locally with development server
  - Push the changes to GitHub

    ```
    git add .
    git commit –m "some updates"
    git push origin master
    ```

  - Then you can see Amplify automatically rebuilds (*which takes some time*)
- This is only the beginning of a longer tutorial on AWS, which you may try yourself later: [*https://aws.amazon.com/getting-started/hands-on/build-react-app-amplify-graphql/module-one/*](https://aws.amazon.com/getting-started/hands-on/build-react-app-amplify-graphql/module-one/)

Create an API on Lambda

- Our next task is to build a serverless API
- Return to the AWS Console
- Find ***Lambda***
- Create a new function
  - Choose "***Author from scratch***"
  - Set a function name you like, e.g. "hello-world"
  - Choose runtime as Node.js 14.x
- Now you can create the function with other default options



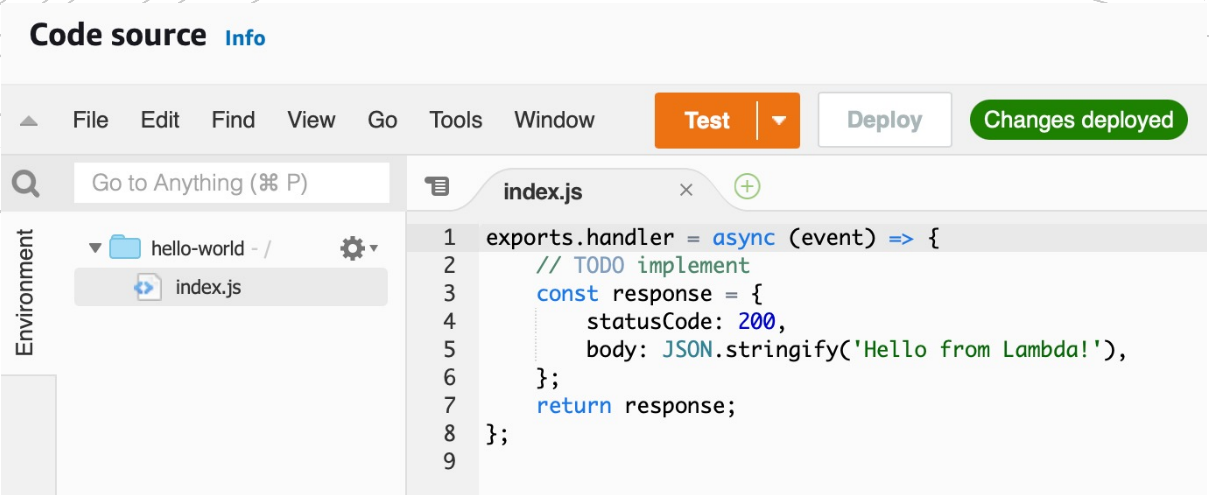
hello-world

Description
-

Last modified
33 seconds ago

Function ARN
arn:aws:lambda:us-east-1:236868335934:function:hello-world

Create an API on Lambda

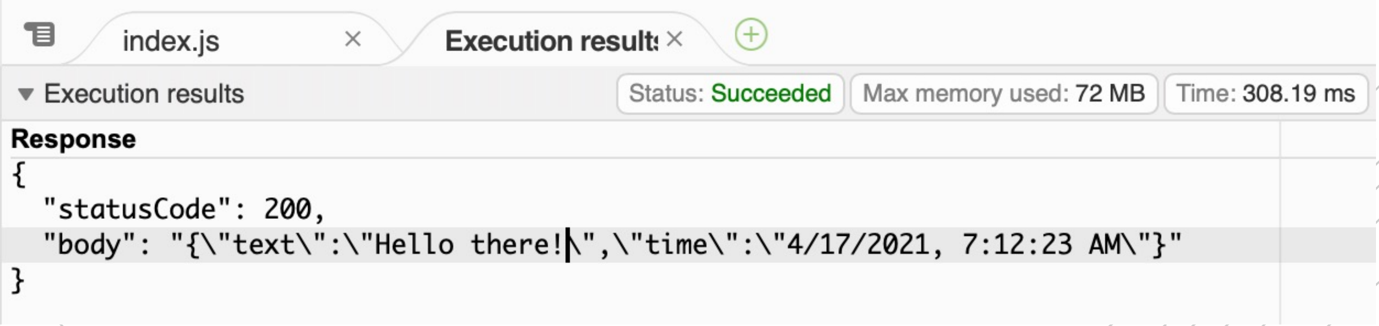


```
exports.handler = async (event) => {
    // TODO implement
    const response = {
        statusCode: 200,
        body: JSON.stringify('Hello from Lambda!'),
    };
    return response;
};
```

- Under Code source, you can click and edit index.js
- Let's change the *response* ***body*** to this

```
body: JSON.stringify({text: 'Hello there!', time: new Date().toLocaleString()}),
```

- You can "Deploy" and then "Test"
  - The testing event name and contents do not matter
  - You should see some testing results in JSON format



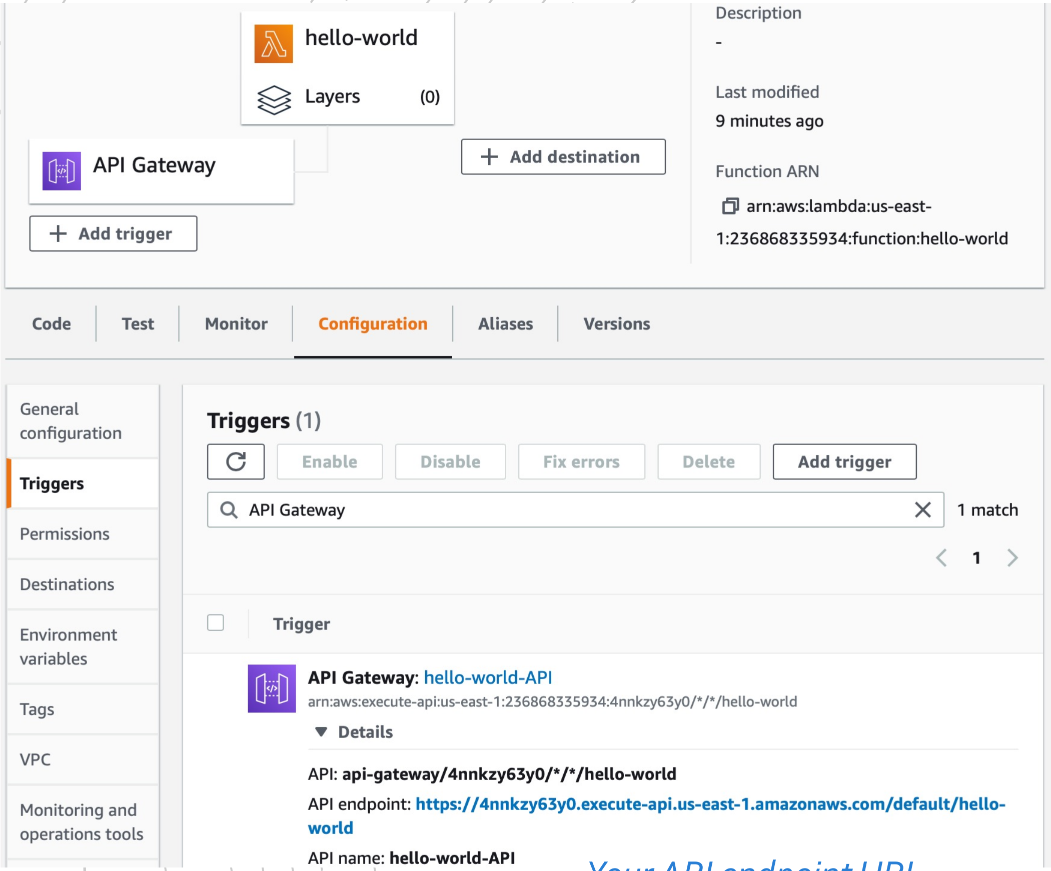
Status: Succeeded | Max memory used: 72 MB | Time: 308.19 ms

Response

```
{
  "statusCode": 200,
  "body": "{\"text\":\"Hello there!\",\"time\":\"4/17/2021, 7:12:23 AM\"}"
}
```

Create an API on Lambda

- This function in Lambda needs to be triggered by an event
  - E.g. HTTP request, image upload, a queue, a notification, etc.
- In the function overview, choose ***Add Trigger***
  - Choose ***API Gateway***
  - Create a new API, and pick ***REST API***
  - Use ***Open*** as the security mechanism (which is not a good idea!)



Description
-

Last modified
9 minutes ago

Function ARN
arn:aws:lambda:us-east-1:236868335934:function:hello-world

Triggers (1)

API Gateway ✕ 1 match

□ Trigger

API Gateway: hello-world-API
arn:aws:execute-api:us-east-1:236868335934:4nnkzy63y0/*/*/hello-world

API: api-gateway/4nnkzy63y0/*/*/hello-world

API endpoint: https://4nnkzy63y0.execute-api.us-east-1.amazonaws.com/default/hello-world

API name: hello-world-API

Your API endpoint URL

- The API Gateway now accepts HTTP requests and show the response in JSON
- You can try this endpoint link in your browser
- We will then access this response text from the React app we built earlier

Back to the React app

- Going back to your *react-app-aws*, now convert your App from a functional component into a class component by changing the syntax carefully
  - You may need to add `import React from 'react';`
- Add the following before `render()`

```
constructor(props) {
  super(props);
  this.state = { apitext: '' };
}
componentDidMount() {
  fetch("    Your API endpoint URL    ")
  .then(res => res.json())
  .then(
    (result) => {
      this.setState({
        apitext: result.text
                 + result.time
      });
    });
}
```

  - In the `<p>` in `render()`, show `{this.state.apitext}` instead of Hello World
  - Try deployment and you may see a CORS error in the console

Back to the React app

- Add, commit and push your edit
- You also see the CORS error from the Amplify app URL
- We need to adjust the CORS headers…
  - In the ***hello-world*** Lambda function, in the code *index.js*, add this to the response before ***body***, then deploy

```
headers: {
  "Access-Control-Allow-Headers" : "Content-Type",
  "Access-Control-Allow-Origin": " Your Amplify app URL ",
  "Vary": "Origin",
  "Access-Control-Allow-Methods": "OPTIONS,POST,GET"
},
```

- Then, commit and push again in GitHub
- Hopefully, your API text should show up
  - *Otherwise just check the console for further errors…*

Just a primer…

- This lab serves only a tiny taste of serverless environment for you
- There are tons of interesting tutorials where you can learn more, basing on what you have already knew
- Here is a list provided by AWS officially
  - *https://aws.amazon.com/getting-started/hands-on/*

Submission

- No submission is needed for labs
- Hope that you have learnt something useful!
- There is no assignment for AWS