Create and Deploy a Basic Blog Using AWS Amplify and React

AWS Amplify is a development platform that provides a set of tools and services to help front-end web and mobile developers build scalable and secure cloud-powered applications. It enables developers to quickly configure and deploy cloud resources, set up authentication and authorization, and integrate with other AWS services, all from a single unified dashboard.

With Amplify, developers can focus on building great user experiences while AWS handles the underlying infrastructure.

Goals:

Through this tutorial, you will be able to create your first Amplify project, add authentication to it, configure and use DynamoDB database, and add a custom domain to your application. At the end, you will deploy your app with AWS Amplify.

Part 1: Configure Amplify CLI

The Amplify Command Line Interface (CLI) is a tool that allows you to create, configure, and manage AWS Amplify projects and services from the command line.

1. To install the Amplify CLI, run the following command:



2. Configure Amplify by running the following command:



Amplify configure will open a browser and ask you to sign into the AWS Console.

Once you're signed in, Amplify CLI will ask you to create an IAM user. Create a user with AdministratorAccess to your account to provision AWS resources. Once the user is created, Amplify CLI will ask you to provide the accessKeyId and the secretAccessKey to connect Amplify CLI with your newly created IAM user.

After that, you will see the "Successfully set up the new user." Message.

Part 2: Create React App

Create an app for your application in the AWS Amplify Console, where you can view and manage your Amplify projects.

aws	Services	Q Search	[Alt+S]	
=	All apps > (Treate app backend		
	Get sta	rted with Amplify Studio		
	Amplify Studio full control ov	o is a visual development environment for building full-stack web and mob er your app design and behavior. Learn more 🔀	bile apps. With Studio, you can quickly l	ouild a
	App deta	nils		
	App name			
	HirasBlog			
			Confirm deployment	

Part 3: Install and consume AWS Amplify Libraries

You need to install the Amplify JavaScript library aws-amplify and Amplify UI library for React @aws-amplify/ui-react (contains the React UI components).

1. Run the following command to install them:

D:\CloudComputing\amplify-app>npm install aws-amplify @aws-amplify/ui-react
up to date, audited 2654 packages in 8s
263 packages are looking for funding run `npm fund` for details
6 high severity vulnerabilities
To address all issues (including breaking changes), run: npm audit fixforce
Run `npm audit` for details.

2. Now, to make your front end use these libraries, open the src/index.js file and replace its entire content with the following code to initialize your Amplify libraries:

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';
import "@aws-amplify/ui-react/styles.css"; // Ensure React UI
libraries are styled correctly
import { Amplify } from 'aws-amplify'
import awsconfig from './aws-exports'
Amplify.configure(awsconfig) // Configures the Amplify libraries
with the cloud backend set up via the Amplify CLI
const root =
ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
reportWebVitals();
```

Part 4: Add a GraphQL API with Amplify

You are now going to add an API to the application. Amplify uses AWS AppSync and Amazon DynamoDB to power GraphQL APIs. AppSync is a managed GraphQL service that will take care of our API, and Amazon DynamoDB is a NoSQL database that will store the data our API will use.

1. To add the API, run the following command:



If you entered Yes to edit the schema now, your default editor should open to the file needed for the next section.

Part 5: Create a GraphQL Schema

A GraphQL schema is a representation of an object and its fields. You need to define your GraphQL schema, and Amplify will create the required DynamoDB table, and configure GraphQL to handle the reads, writes, updates and deletes for you.

1. Open the amplify/backend/api/amplifyapp/schema.graphql file and replace the contents with the following:

```
type BlogPost @model @auth(rules: [{ allow: owner }]) {
  id: ID!
   title: String!
   text: String!
}
```

Go back to command line and continue the execution. Once successful, you will see this message.

√ Do you wan	nt to edit the so	chema now? (Y	//n) · yes	amplify\backend\api\amplifyapp\schema.graphql				
Edit the fi	le in your editor	: D:\CloudCo	omputing\amplify-app					
❷ Successfu	lly added resourc	ce amplifyapp	locally					
☑ Some next	B Some next steps:							
"amplify pu:	"amplify push" will build all your local backend resources and provision it in the cloud							
"amplify pu	"amplify publish" will build all your local backend and frontend resources (if you have hosting category added) and prov							
ision it in	ision it in the cloud							
√ Successfu Current	lly pulled backer Environment: dev	nd environmer /	nt dev from the cloud					
Category	Resource name	Operation	Provider plugin					
Api	amplifyapp	Create	awscloudformation					

Part 6: Deploy Application

You are now ready to deploy your Amplify web application.

1. Run the following command:



Accept all the default values to configure auth with a simple username and password combination. After confirming the auth settings, select the default values to the follow up questions when prompted.

Part 7: Update Front End to Use API

To use the new API and auth backend you just deployed, update the src/App.js file by replacing the contents with the following code:

```
import './App.css';
import { createBlogPost, deleteBlogPost } from './graphql/mutations'
import { listBlogPosts } from './graphql/queries'
import { withAuthenticator, Button, Text, Flex, Heading } from
"@aws-amplify/ui-react";
```

```
import { useCallback, useEffect, useState } from 'react';
import { API } from 'aws-amplify';
function App({ signOut }) {
  const [blogPosts, setblogPosts] = useState([])
  const fetchBlogPosts = useCallback(async () => {
    const result = await API.graphql({
      query: listBlogPosts,
      authMode: 'AMAZON COGNITO USER POOLS'
    })
    setblogPosts(result.data.listBlogPosts.items)
  }, [setblogPosts])
 const handleCreateBlogPost = useCallback(async () => {
   await API.graphql({
      query: createBlogPost,
      variables: { input: { title: window.prompt("New Post Title"),
text: window.prompt("New Post Text") } },
      authMode: 'AMAZON COGNITO USER POOLS'
    })
   fetchBlogPosts()
  }, [fetchBlogPosts])
  const handleDeleteBlogPosts = useCallback(async (id) => {
   await API.graphgl({
      query: deleteBlogPost,
      variables: { input: { id: id } },
      authMode: 'AMAZON COGNITO USER POOLS'
    })
   fetchBlogPosts()
  }, [fetchBlogPosts])
 useEffect(() => {
   fetchBlogPosts()
  }, [fetchBlogPosts])
  return (
    <Flex direction={"column"}>
      <Flex justifyContent={'space-between'}>
        <Heading level={1}>My Posts</Heading>
        <Button onClick={signOut}>Sign Out</Button>
      </Flex>
      {blogPosts.map(post => <Flex alignItems={'center'}>
        <Text>{post.title}</Text>
        <Text>{post.text}</Text>
        <Button onClick={() =>
handleDeleteBlogPosts(post.id)}>Remove</Button>
      </Flex>) }
      <Button onClick={handleCreateBlogPost}>Add a Post</Button>
    </Flex>
 );
export default withAuthenticator(App);
```

Part 8: Test your application

Now you can run your app locally to test it. Run the following command.

<u>D:\CloudComputing\amplify-app</u> >npm s	tart
> amplify-app@0.1.0 start > react-scripts start	

You should be able to see the app running on localhost.

	👹 Read	ct App	×	+																		-	٥
	C	i localhost:3000														A٩	τô		G	£≡	Ē	3	
💮 Da	ashboard	Change request form		Implementing smar	Course	e Offering	basic	0	Getting Starte	d Pl	💿 ps	O JESS	O munityapps/keysaa	s 😕 A	Australasia –	Overvi.	0	Building A	pplicatio.				>
My Posts)ut															
Add a Post																							

Part 9: Store project on GitHub

Create a private repo, and push the code you have created so far to that repo.

Part 10: Setup Continuous Deployment

To configure Amplify to deploy your code, you need to connect it with your GitHub account. This is done via the AWS Console as it needs to generate a GitHub token to access your private repo, and store it in your AWS account.

1. To do this, from the amplify-app directory, run amplify add hosting. Amplify will present a list of questions about the hosting, please select the options shown below:

```
amplify add hosting
?
? Select the plugin module to execute (Use arrow keys)
4 > Hosting with Amplify Console (Managed hosting with custom domains, Continuous deployment)
5 Amazon CloudFront and S3
6 ? Choose a type
7 > Continuous deployment (Git-based deployments)
8 Manual deployment
9 Learn more
```

2. This will launch launch a new window in your browser, and open the Amplify Console for you to configure hosting on your project. From the app page on the Amplify Console, click on the Frontend environments tab, select GitHub and click on the Connect branch button.

amplifyapp		Actions v
The app homepage lists all deployed frontend a	nd backend environments.	
Frontend environments Backend env	ironments	
Host a web app Connect your source code from a Git reposite	ory or upload files to host a web app in minutes.	
• GitHub	O Bitbucket	○ GitLab
○ AWS CodeCommit	O Deploy without Git provider	
		Connect branch

3. This will redirect you to GitHub. You now will give the Amplify Console access to your GitHub account so it can deploy the source code you are hosting there. To do so, click on the green Authorize aws-amplify-console button.

AWS AI	mplify (us-east-1) by AWS Amplify Console would like permission to:
	 Verify your GitHub identity (stampcrab) Know which resources you can access Act on your behalf (2) Learn more
	Cancel Authorize AWS Amplify (us- east-1) Authorizing will redirect to https://us-east-1.console.aws.amazon.com

4. You will now connect your GitHub repo to your Amplify application. From the Add repository branch page, select the amplify-web-app repo and the main branch. Then, click on Next.

Services Q Search		[Alt+5]
AWS Amplify $\qquad imes$	All apps > amplifyapp > Co	onnect branch
All apps	Step 1 Add repository branch	Add repository branch
amplifyapp	Step 2 Build settings	GitHub
App settings General	Step 3 Review	GitHub authorization was successful.
Amplify Studio settings		Repository service provider
Documentation 🖸 Support 🗹		Recently updated repositories
		stampcrab/aws-amplify-app C
		If you don't see your repository in the dropdown above, ensure the Amplify GitHub App has permissions to the repository. If your repository still doesn't appear, push a commit and click the refresh button.
		Branch
		Select a branch from your repository.
		Connecting a monorepo? Pick a folder.

5. The next step is to configure the build settings for your application. On the Configure build settings page, click on the Create new role button inside the blue box.

Create a new service role. In the window that create a new service role.	opens, accept the pre-selected defaults on each scree	en to	Create no	ew role
6. click on Create Role:				
Create role			1 2	3 4
Review				
Provide the required information below and review	this role before you create it.			
Role name*	amplifyconsole-backend-role			
	Use alphanumeric and '+=,.@' characters. Maximum 64 cha	aracters.		
Role description	Allows Amplify Backend Deployment to access AW	/S resources of	n your behalf.	
	Maximum 1000 characters. Use alphanumeric and '+=,.@' c	characters.		
Trusted entities	AWS service: amplify.amazonaws.com			
Policies	AdministratorAccess-Amplify 🖉			
Permissions boundary	Permissions boundary is not set			
No tags were added.				
* Required		Cancel	Previous	Create role

7. Once created, you can close this tab in your browser, and return to the Configure build settings page, click on Refresh existing roles and then select the role you just created from the dropdown menu. Then, select dev from the Environment dropdown menu - this is the environment you created when configuring Amplify on your project after running amplify init. Leave the remaining settings with their default values, and click on Next.

Step 1 Add repository branch	Build settings
Step 2 Build settings	App build and test settings
Step 3 Review	App name amplifyapp
	Auto-detected frameworks Frontend framework React Backend framework Amplify
	Select a backend environment to use with this branch App name Environment amplifyapp (this app) Full-stack CI/CD allows you to continously deploy frontend and backend changes on every code commit Enable full-stack continuous deployments (CI/CD)
	Select an existing service role or create a new one so Amplify Hosting may access your resources. amplifyconsole-backend-role
	A Filter service roles amplifyconsole-backend-role New role

8. Review the values you configured, and click on Save and deploy. Amplify will now start to automatically deploy your React application on source repository changes.

All apps > amplifyapp amplifyapp The app homepage lists all deployed frontend and backend environments.	Actions v
• Learn how to get the most out of Amplify Hosting	1 of 5 steps complete X
Hosting environments Backend environments	
This tab lists all connected branches, select a branch to view build details.	Connect branch
master Continuous deploys set up with dev backend (Edit)	
	Provision Build Deploy
Last deployment 3/21/2023, 11:48:06 J	Last commit Previews Changed ts version 5d8c52b GitHub - master 🖸 Disabled