Documentation: Creating a Pet-Tracker App using React Native and AWS Amplify

Introduction:

This documentation will guide you through the process of creating a Pet-Tracker app using React Native and AWS Amplify. The app will allow users to track their pets, store pet photos in Amazon S3, and store pet details in Amazon DynamoDB. We will also use AWS Lambda functions to handle serverless operations and interact with the storage and database services.

**Prerequisites:**

Before getting started, make sure you have the following:

1. Node.js installed on your machine (preferably version 12 or later)

2. NPM (Node Package Manager) installed

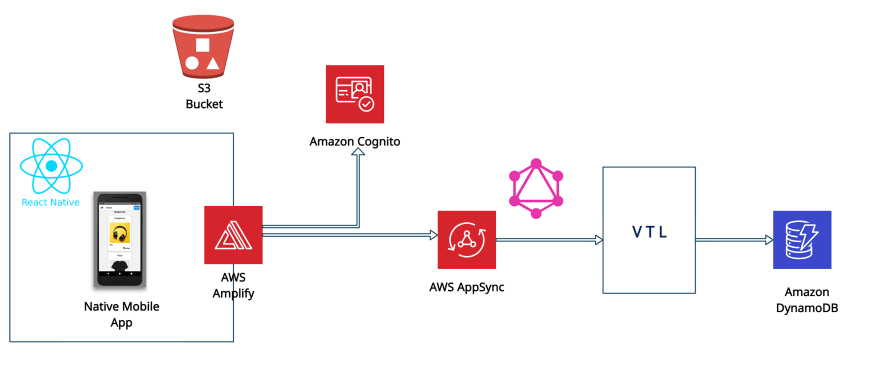
3. React Native CLI installed globally (`npm install -g react-native-cli`)

4. An AWS account with access to create and manage AWS services

**AWS Services used.**

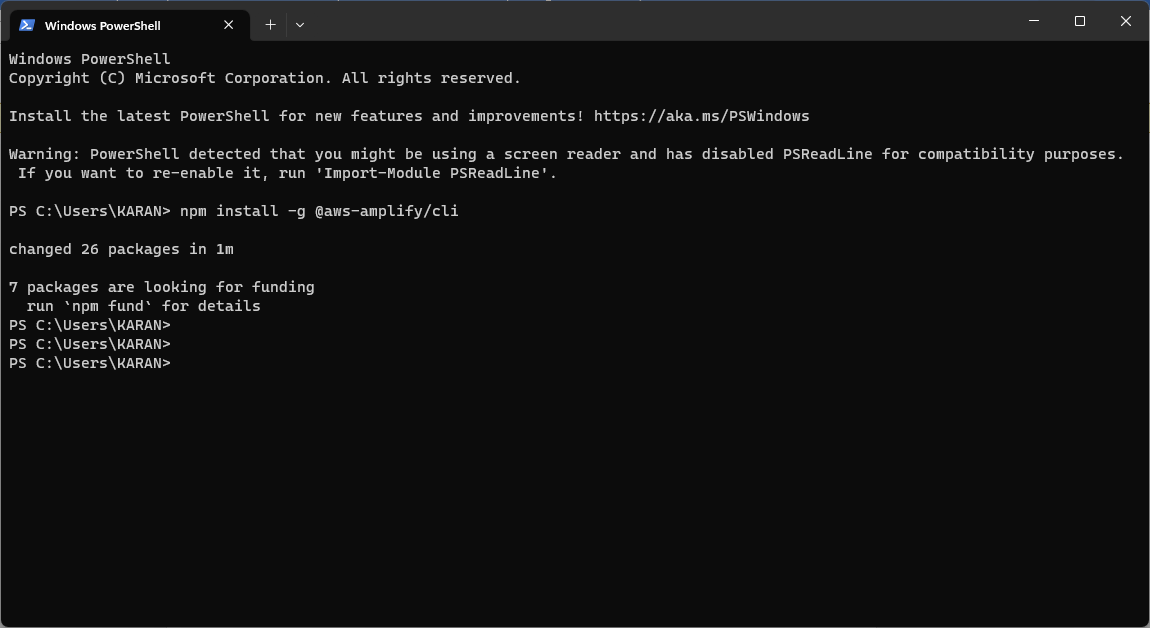
* Amazon Cognito User Pools.
* Amazon API Gateway.
* Amazon S3

**Architecture:**

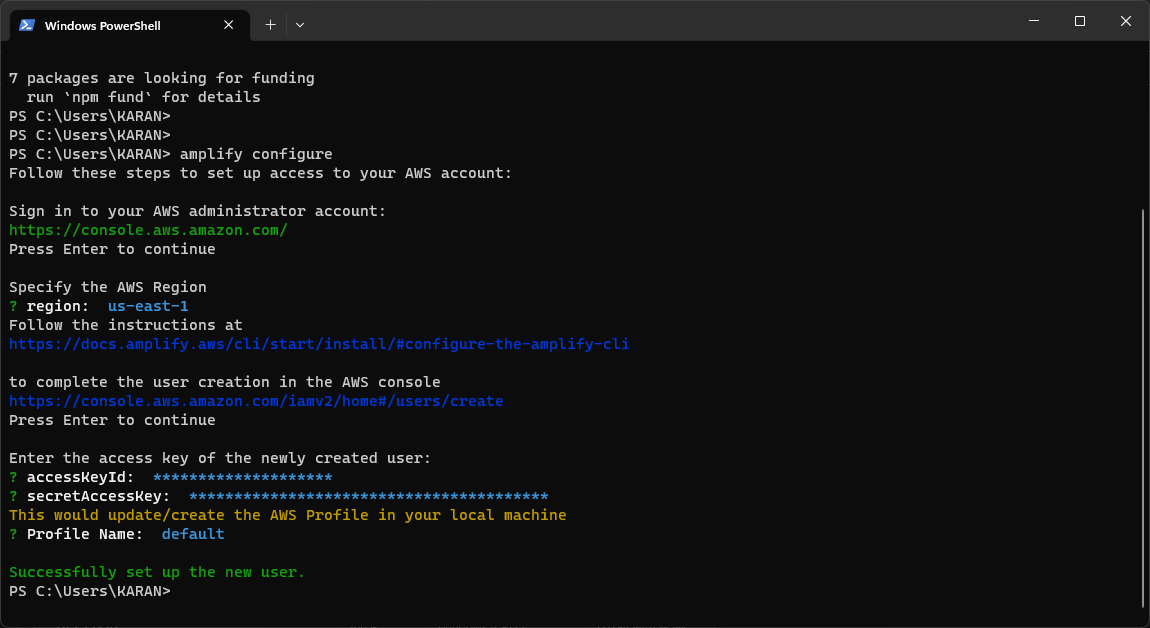
[](https://res.cloudinary.com/practicaldev/image/fetch/s--ACT7nOMI--/c_limit%2Cf_auto%2Cfl_progressive%2Cq_auto%2Cw_880/https:/dev-to-uploads.s3.amazonaws.com/i/yz3l2dv9wki6uahjakv2.jpg)

Step 1: Setting Up AWS Amplify

1. Install the AWS Amplify CLI globally by running the following command in your terminal:

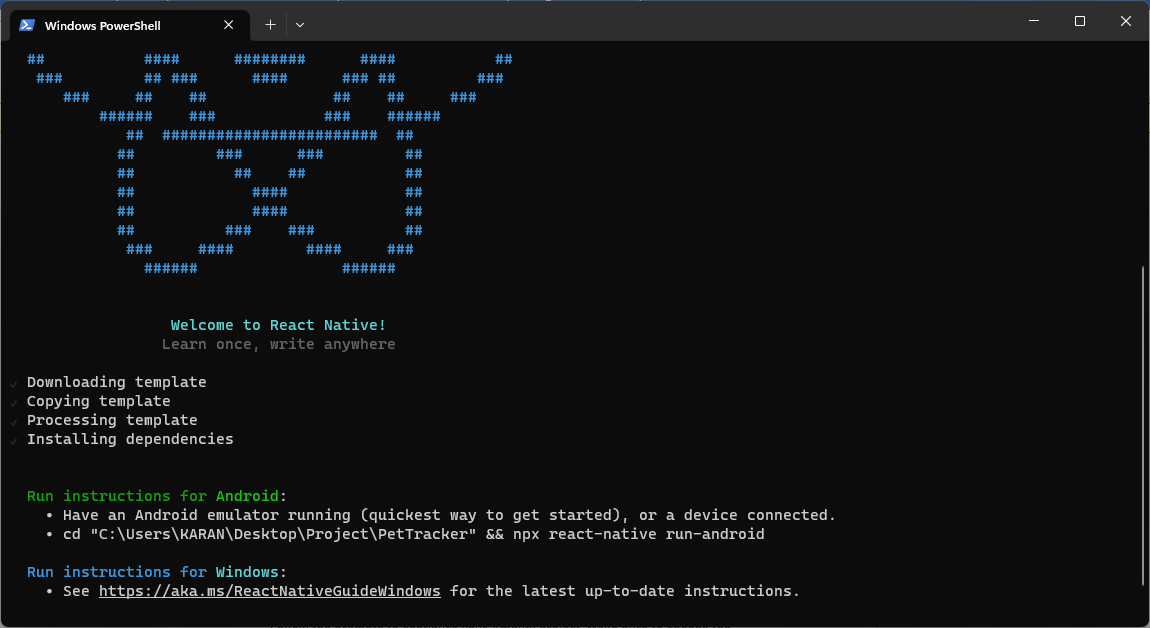
npm install -g @aws-amplify/cli 

2. Configure the AWS Amplify CLI by running the following command and following the prompts:

amplify configure 

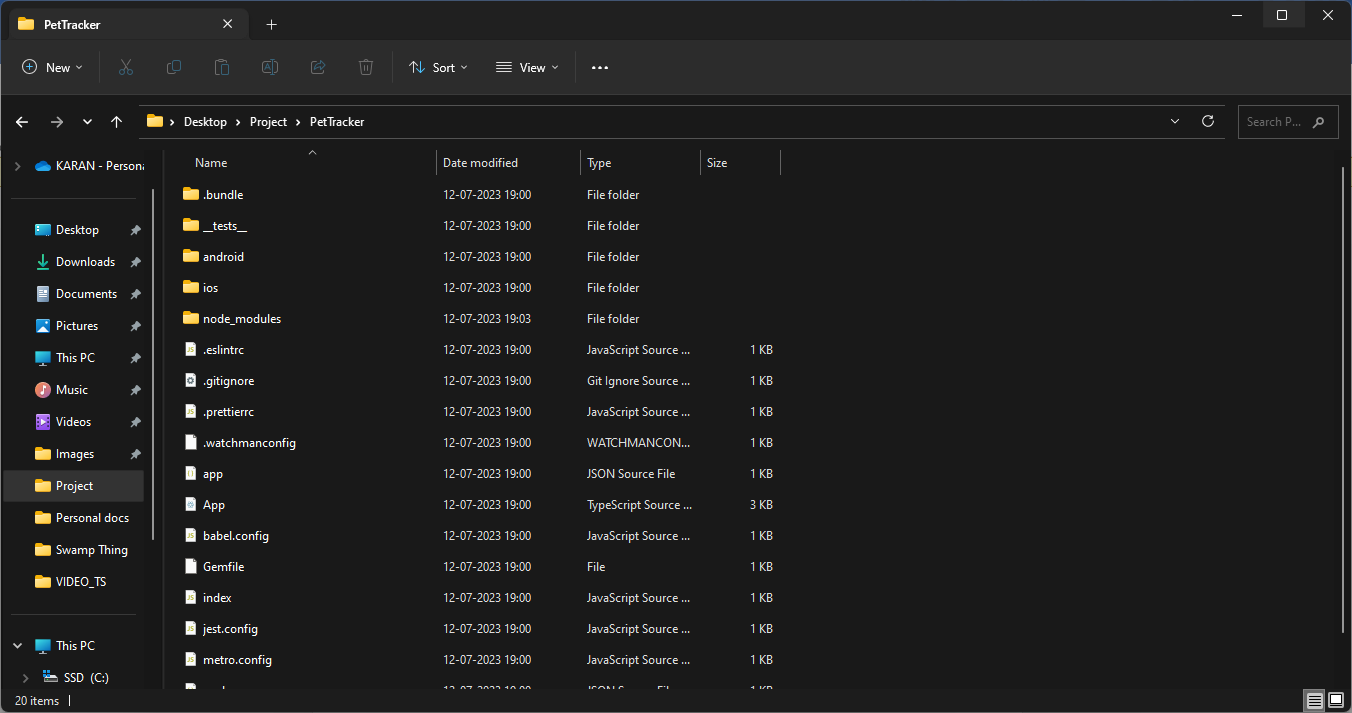
This will prompt you to enter your AWS access keys, secret access keys, and other required information. Make sure to configure the CLI with the AWS account you want to use for this project.

3. Once the configuration is complete, create a new React Native project by running the following command:

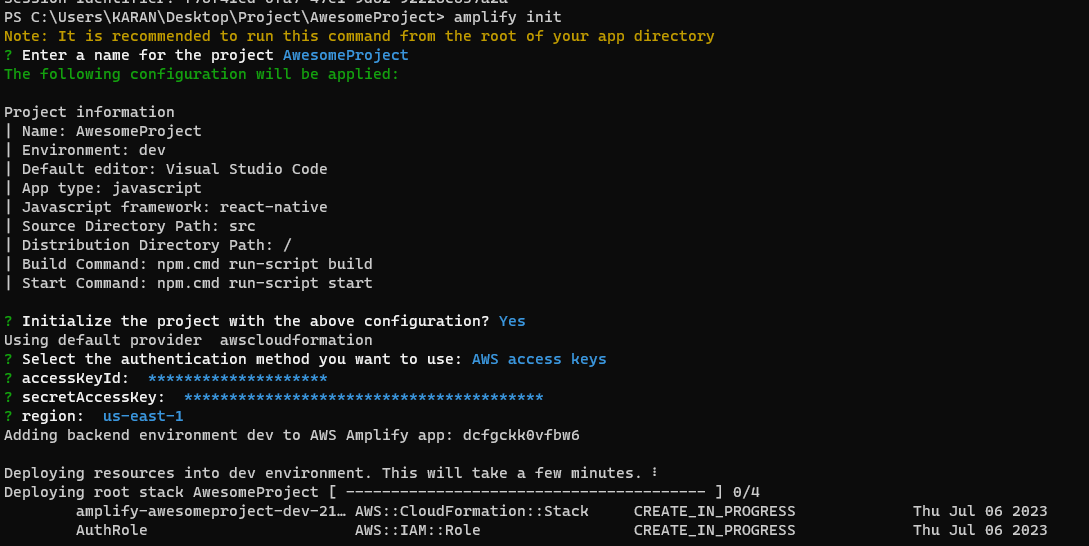
npx react-native init Pet-Tracker 

It will create all files and dependencies files in your Project Directory.

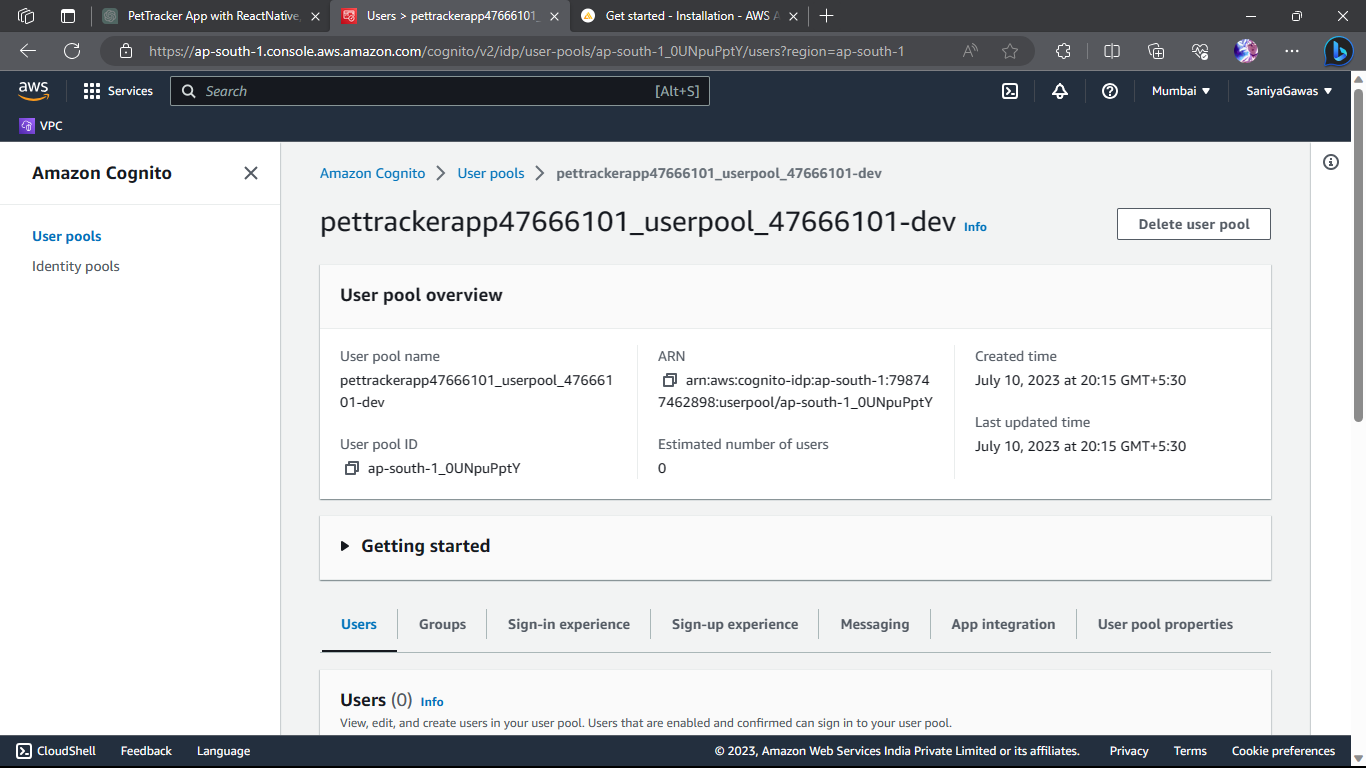
4. Change into the project directory:

cd Pet-Tracker 

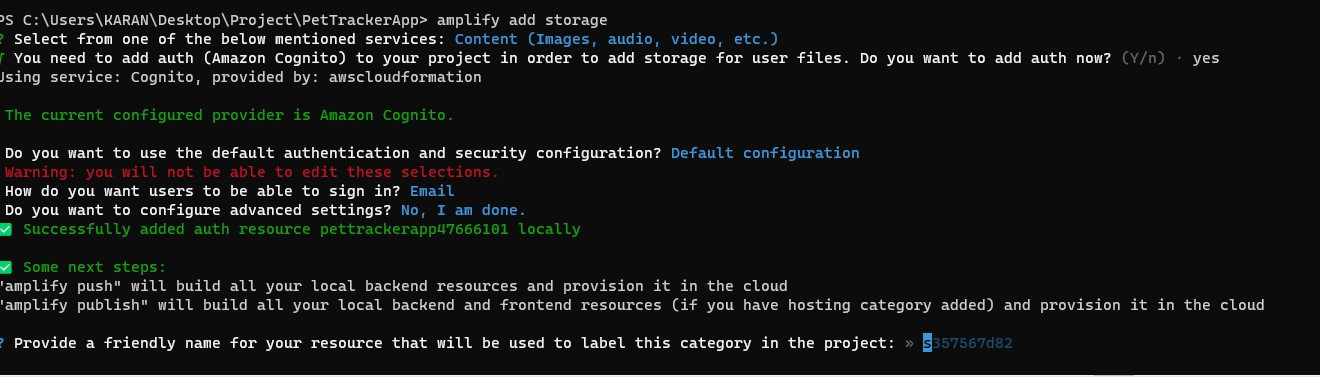
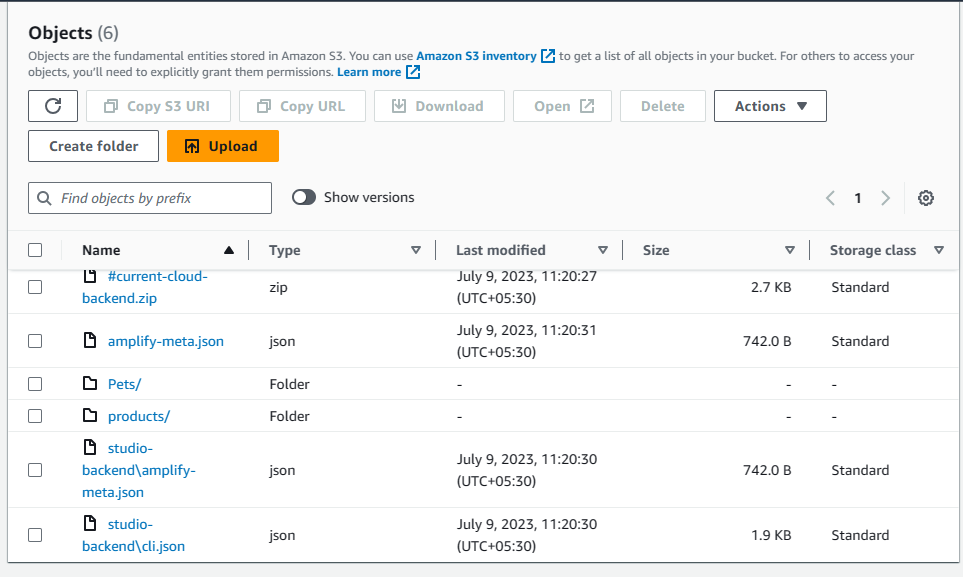
5. Initialize Amplify in your project by running the following command:

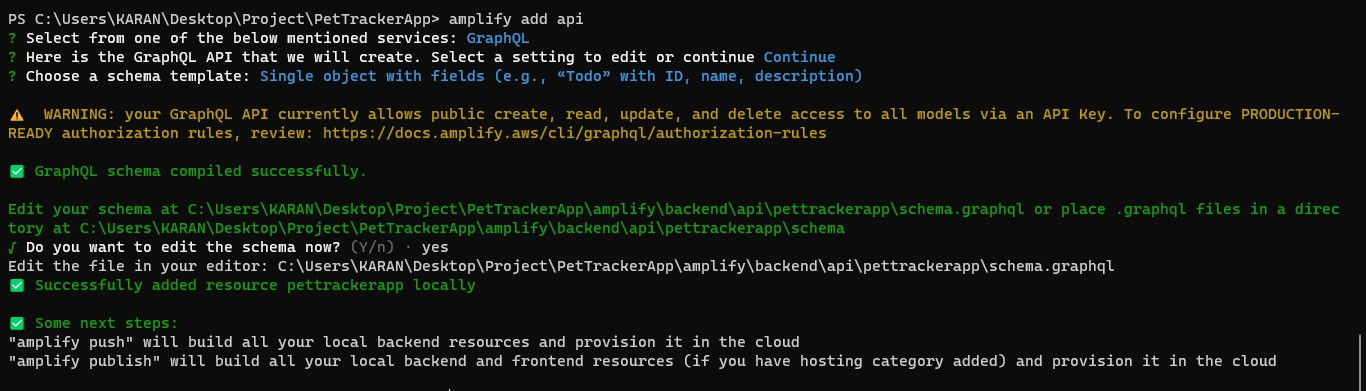
amplify init

6. Adding Cognito Userpool:

Amplify add auth 

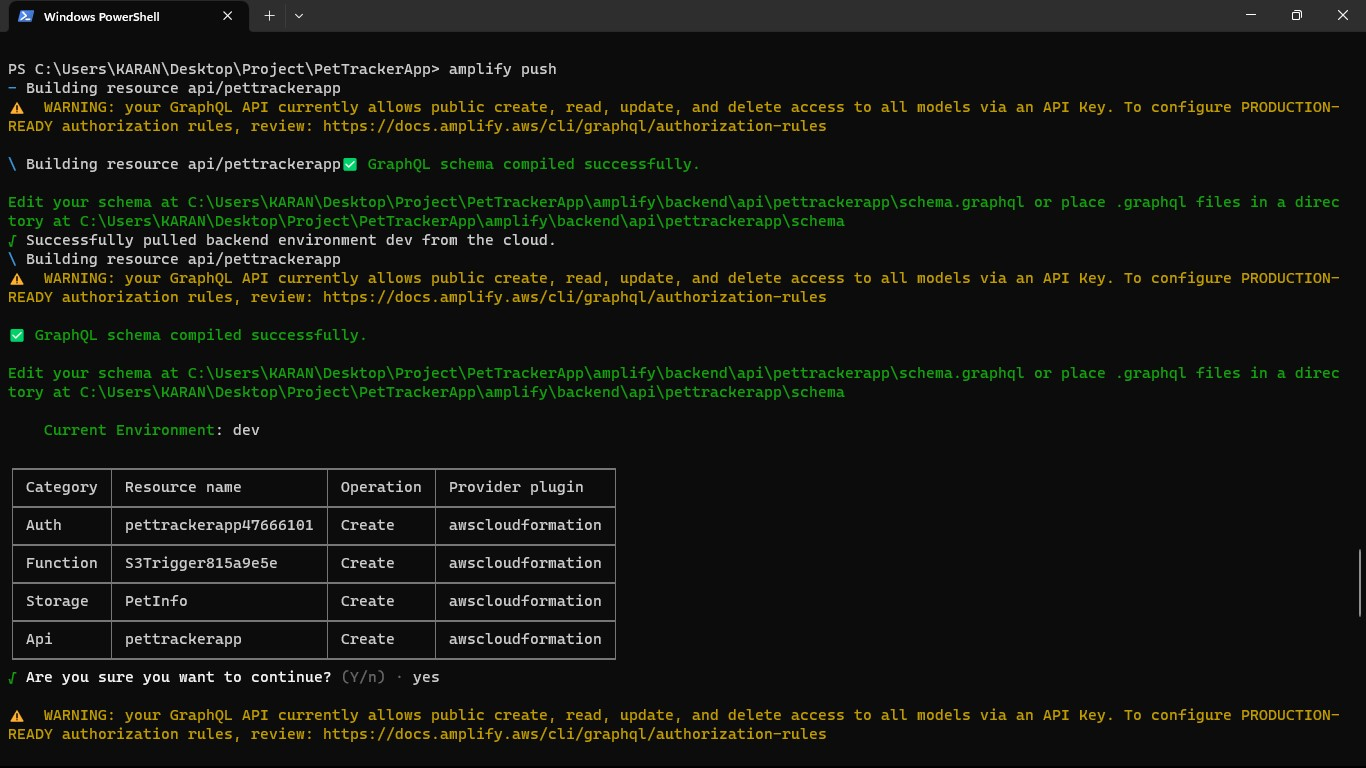
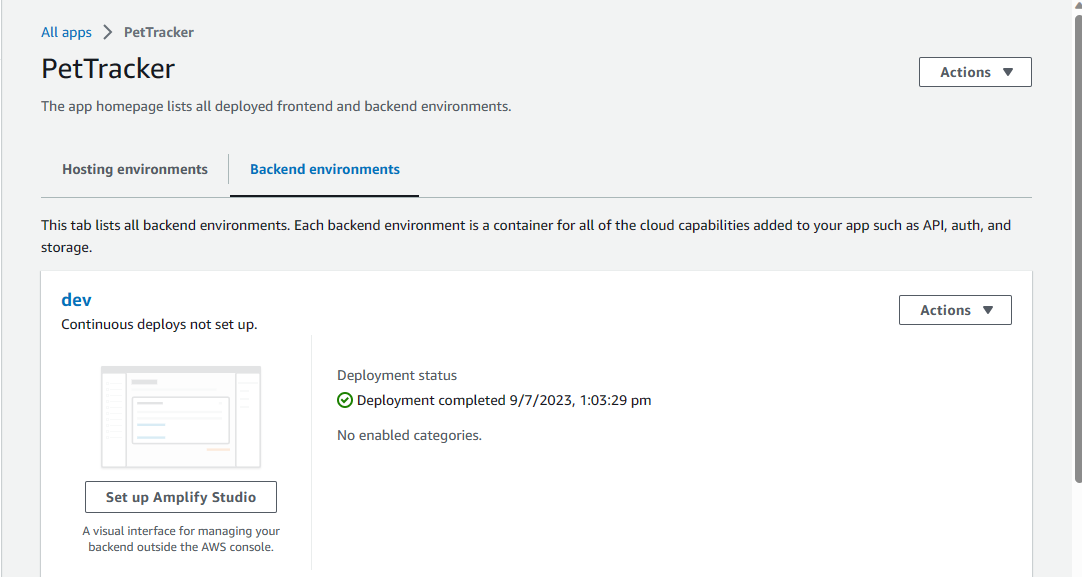
Follow the prompts to configure your Amplify project. Make sure to choose the appropriate options for your project, such as the region and the name of your environment.

7. Add the necessary Amplify categories to your project:amplify add storage******

amplify add api

Follow the prompts and choose the appropriate options for each category.

8. Push the changes to your Amplify environment:

amplify push******

This will create the necessary resources in your AWS account based on the Amplify configuration.

Step 2: Integrating React Native with AWS Amplify

1. Install the AWS Amplify libraries by running the following command in your project directory:

npm install aws-amplify aws-amplify-react-native

2. Import the necessary Amplify libraries in your React Native app. Open the `App.js` file in your project and add the following code at the top:

import Amplify from 'aws-amplify';

import config from './aws-exports';

Amplify.configure(config);

3. Add code to interact with AWS services in your app components. For example, you can create a form to add pet details and upload a photo. Use the Amplify libraries and API methods to interact with the storage and database services.

Here's an example of a component that allows users to add pet details and upload a photo:

import React, { useState } from 'react';

import { View, TextInput, Button, Image } from 'react-native';

import { Storage, API } from 'aws-amplify';

const AddPetScreen = () => {

const [name, setName] = useState('');

const [photoUrl, setPhotoUrl] = useState('');

const handlePhotoUpload = async () => {

// Implement photo upload logic using Amplify Storage

};

const handleAddPet = async () => {

// Implement pet details submission logic using Amplify API

};

return (

<View>

<TextInput

placeholder="Pet Name"

value={name}

onChangeText={setName}

/>

<Button title="Upload Photo" onPress={handlePhotoUpload} />

<Image source={{ uri: photoUrl }} style={{ width: 200, height: 200 }} />

<Button title="Add Pet" onPress={handleAddPet} />

</View>

);

};

export default AddPetScreen;

```

Note: You will need to implement the logic for photo upload and pet details submission using the Amplify Storage and API methods. Refer to the Amplify documentation for more details on how to interact with these services.

Step 3: Storing Photos in Amazon S3

1. In your project directory, open the `amplify/backend/storage/storage-cloudformation-template.json` file and modify the S3 bucket configuration to enable public read access for uploaded photos. Add the following configuration to the `Properties` section of the `Bucket` resource:

```json

"CorsConfiguration": {

"CorsRules": [

{

"AllowedOrigins": ["\*"],

"AllowedHeaders": ["\*"],

"AllowedMethods": ["GET", "PUT", "POST", "HEAD"],

"MaxAge": 3000

}

]

}

```

This configuration allows any origin to read the photos.

This will update the S3 bucket configuration to allow public read access

2. Update your React Native component to handle photo uploads:

const handlePhotoUpload = async () => {

try {

const imagePickerResponse = await ImagePicker.launchImageLibraryAsync({

mediaTypes: ImagePicker.MediaTypeOptions.Images,

allowsEditing: true,

aspect: [1, 1],

quality: 0.7,

});

if (!imagePickerResponse.cancelled) {

const response = await fetch(imagePickerResponse.uri);

const blob = await response.blob();

const fileName = imagePickerResponse.uri.split('/').pop();

await Storage.put(fileName, blob, {

contentType: 'image/jpeg',

});

const photoUrl = await Storage.get(fileName);

setPhotoUrl(photoUrl);

}

} catch (error) {

console.log('Error uploading photo:', error);

}

};

```

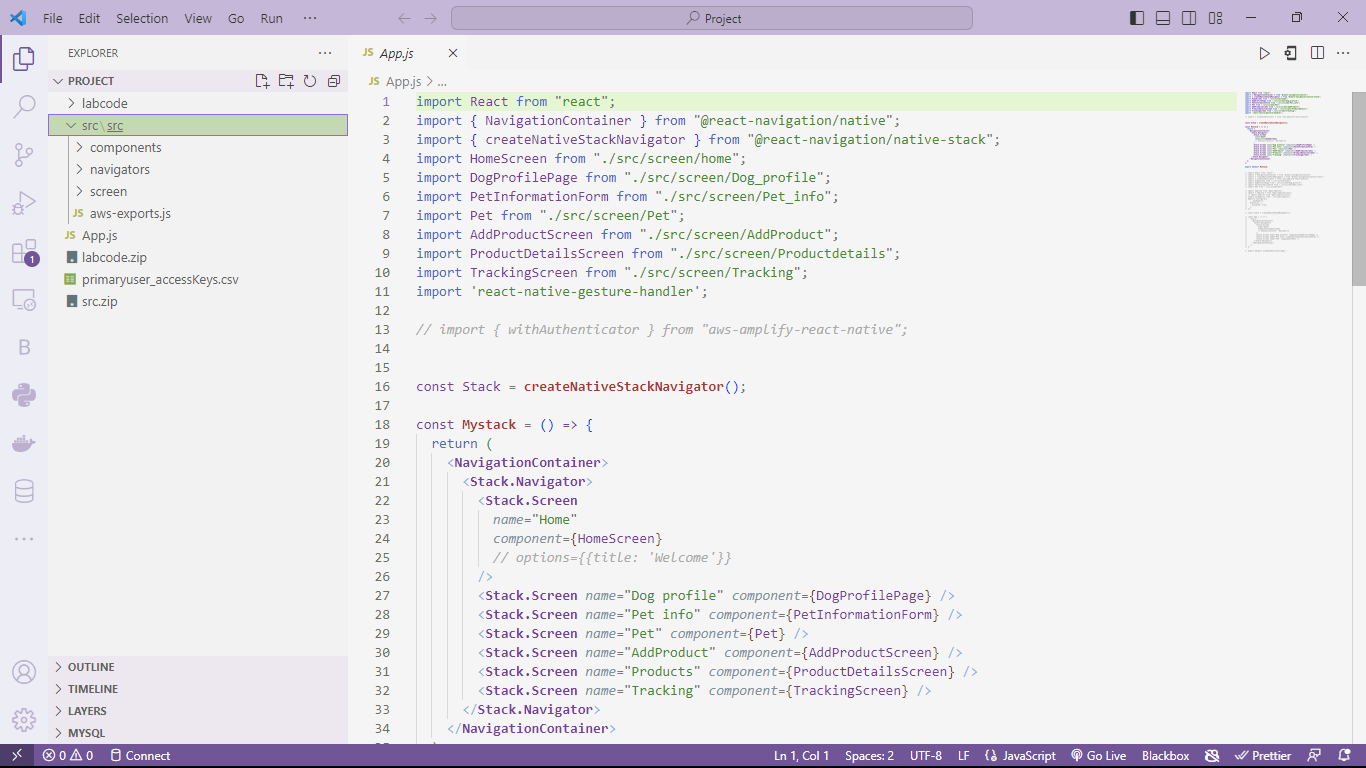
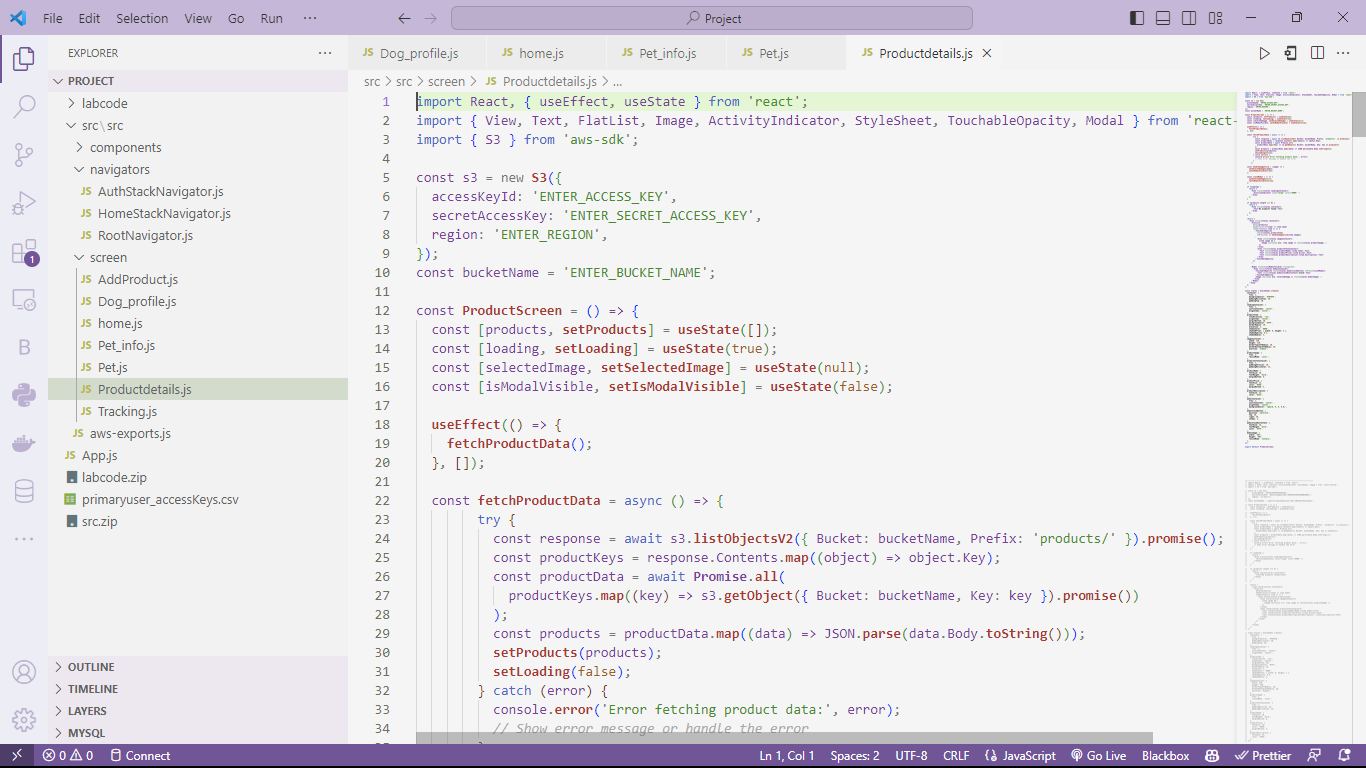
This code uses the `ImagePicker` library to select an image from the device's gallery and then uploads it to the S3 bucket using the Amplify Storage `put` method. After successful upload, the `photoUrl` is updated with the URL of the uploaded photo, which is then displayed in the `Image` component.

Step 4: Adding Different Screens for the app

Our App will contain three screens. One Screen to display the list of Pets and the other to add new Pet and a Tracking Page which Requires GPS enabled.

Create a new file in the src directory. In that directory, create a folder called ‘screens’. In that folder src/screens, create three new javascript files named add-pet-screen.js and home-screen.js.

Edit the Pages with the following code:

1. App.js 
2. Different src file

This src directory file and App.js files can be re-written as per need to modify the app or make changes to the app as per requirement.

Conclusion

This Pet-Tracker app using React Native and AWS Amplify allows users to track their pets, store pet photos & pet details in S3 , You can further enhance the app by adding more features or integrating additional AWS services as needed. This can be used by Vets and Pet-Shop Owners to maintain the records of their animals or Pets.