NextWork.org

Build a Three-Tier Web App

TA tahirgroot@gmail.com

TA
tahirgroot@gmail.com
NextWork Student
NextWork.org

Introducing Today's Project!

In this project, I will demonstrate how to setup a three tier web app from scratch! I ll start with the presentation tier, then se tup the logic tier and finally set up the data tier before tying them all together.

Tools and concepts

Services I used were Amazon S3, Cloudfront, DynamoDB, Lambda, API Gateway, Key concepts I learnt include Lambda functions, CORS errors, updating the javascript filw with the API invoke URL, and testing the invoke URL within the browser

Project reflection

This project took me approximately 2 hours The most challenging part was resolving the cors erros in both APi Gateway nad Lambda. It was most rewarding to see the final outcome - data getting returned in my web app.

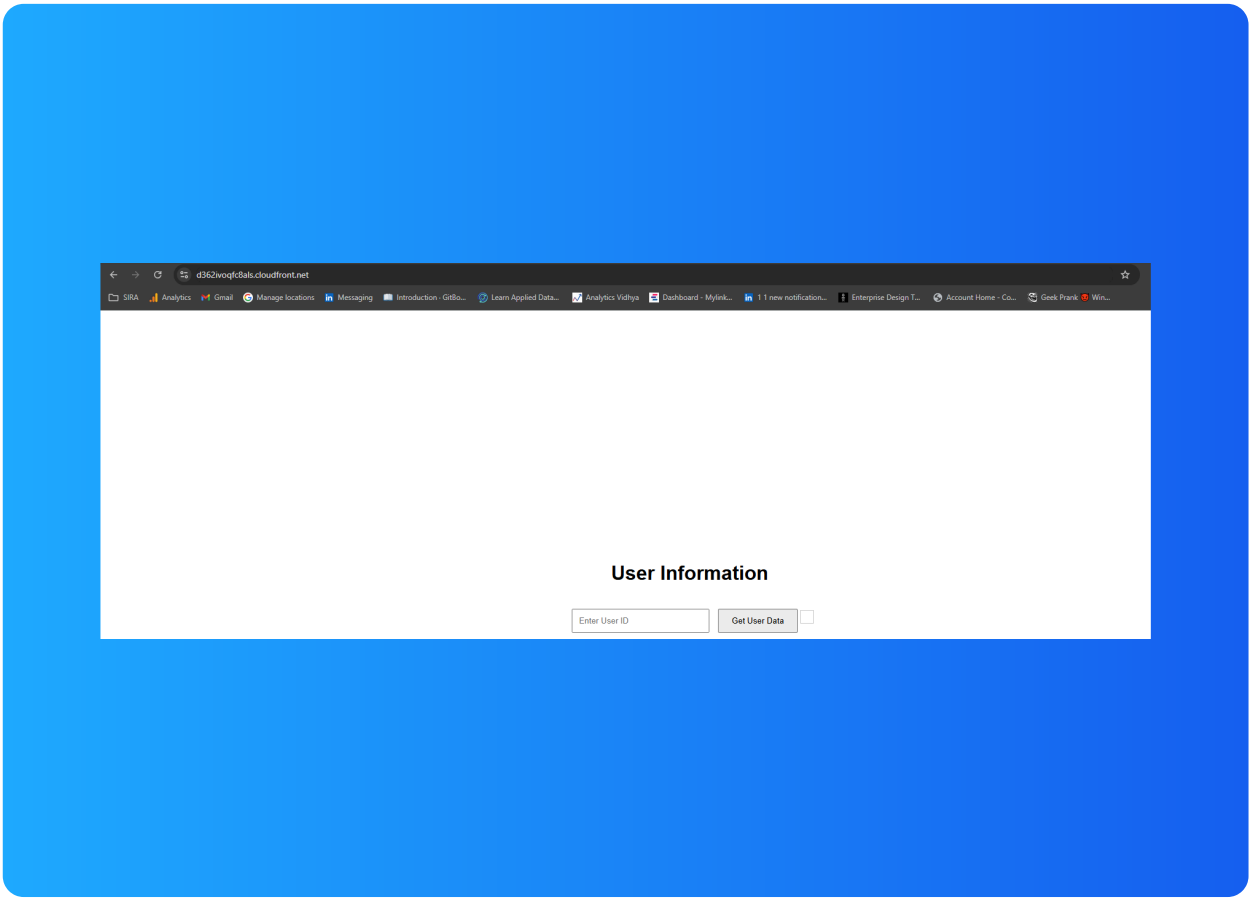
I did this project to learn about the three teir architecture and setup my own webapp. This project absolutley met my goals and i could see a fully functioning web app by the end.

TA
tahirgroot@gmail.com
NextWork Student

Presentation tier

For the presentation tier, I will set up how my website will be displayed and available to my end users. This is because the presentation tier is responsible for storing my website's files(Amazon S3) + website distribution (Amazon Cloudfront).

I accessed my delivered website by visting the cloudfront distribution url. This distribution url is working becuase i also setuo an origin access control that lets my s3 bucket restrict access to only my cloudfront disribution.

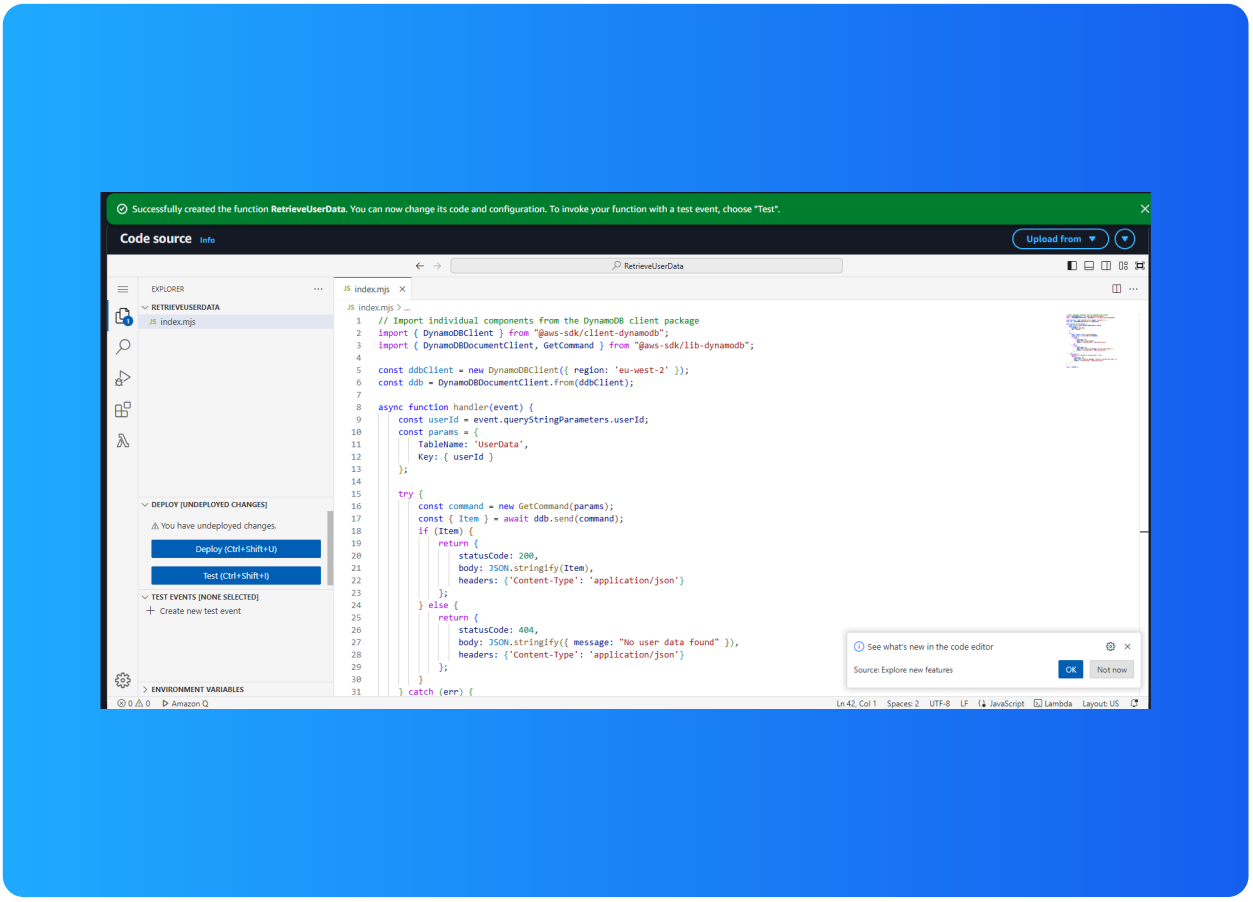


TA
tahirgroot@gmail.com
NextWork Student

Logic tier

For the logic tier, I will set up a lambda fucntion to handle requests (e.g. look up userid to retrun some user data) and also an API with API Gateway to receive requests from the user and hand it over to Lambda.

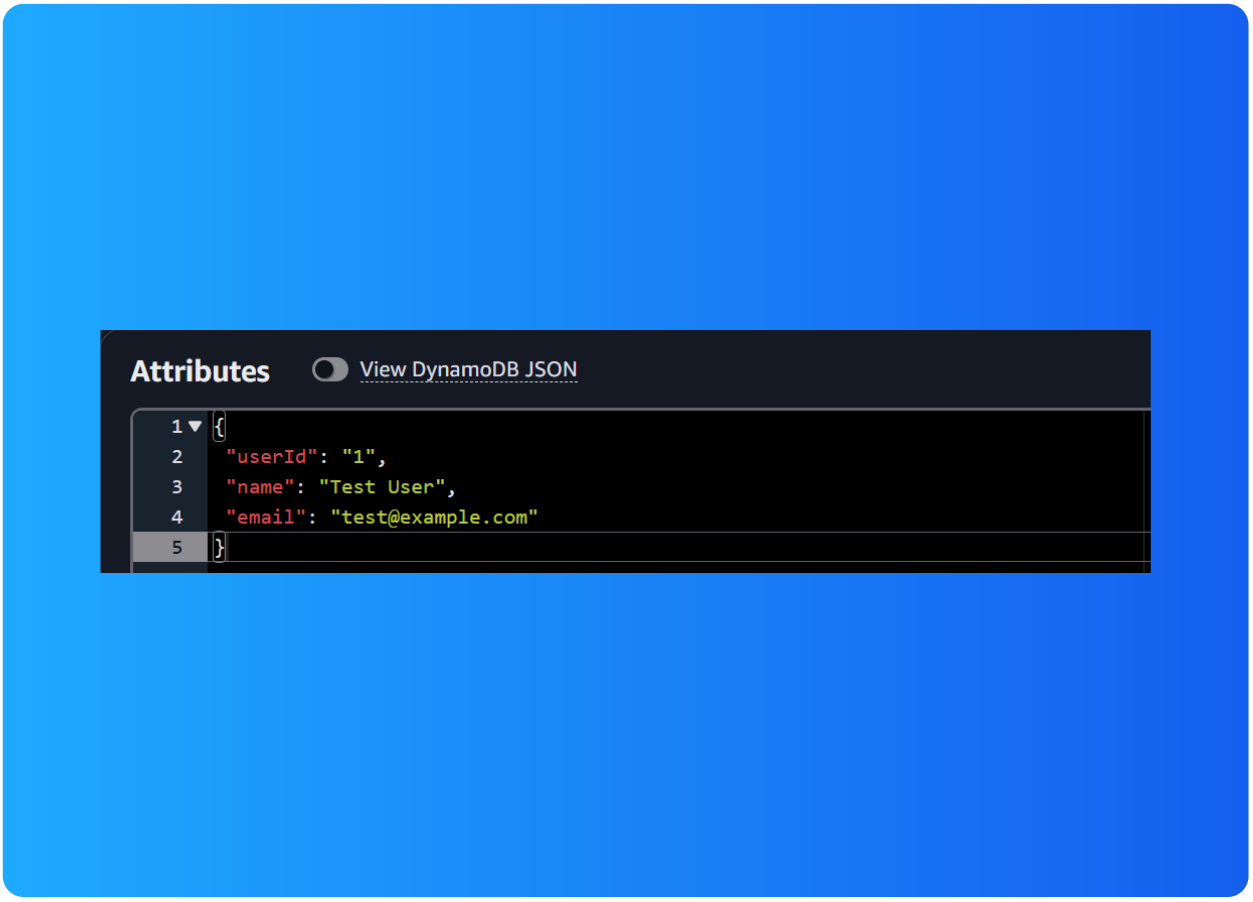
The Lambda function retrieves data by looking up a userid (that the user enters over the web app) in DynamoDB. The aws SDK is used in the function code so we can usetemplates and libraries that lets us find the correct DynamoDB table + request data.



Data tier

For the data tier, I will set up a dynamoDB database that stores user data. At the moment there is no user data for us to return to our web app's users. The data in my database will get returned once i setup Dynamo DB and connect it with Lambda.

The partition key for my DynamoDB table is userid. which means that when my table looks up data, it will look it up based on userid. Then, it can return all data(values) related to the item with that ID.



TA
tahirgroot@gmail.com
NextWork Student

Logic and Data tier

Once all three layers of my three-tier architecture are set up, the next step is to connect the presentation and logic tier. This is because there is no way for my API to catch requests that users make through my distributed site!

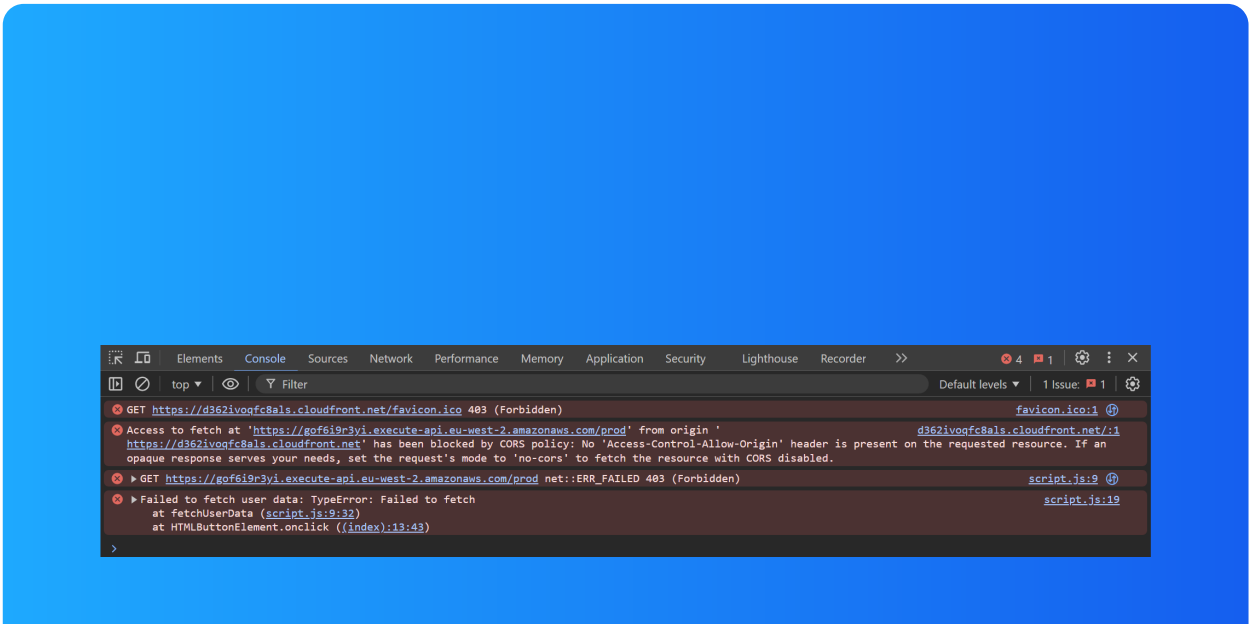
To test my API, I visisted the invoke URL of the prod stage API. This lets us tets whwether i can use the API and retrive user data. The results were some user data in JSON when we looked up userid=1. This proved a logic + data tier connection.

Console Errors

The error in my distributed site was because there was an error within script.js (one of the website files i had uploaded into s3). The script.js file is referencing a prod stage API URL placeholder and not my API's actual URL.

To resolve the error, I updated script.js by replacing some placeholder text with the API's prod stage invoke URL. I then reuploaded script.js into it into S3 because the S3 bucket is still storing the last uploaded version(i.e. with the error).

I ran into a second error after updating script.js. This was an error on the browser side with CORS because API Gateway, by default, is only configured to allow requests from users directly running its invoke URL in the browser(now with cloudfront).

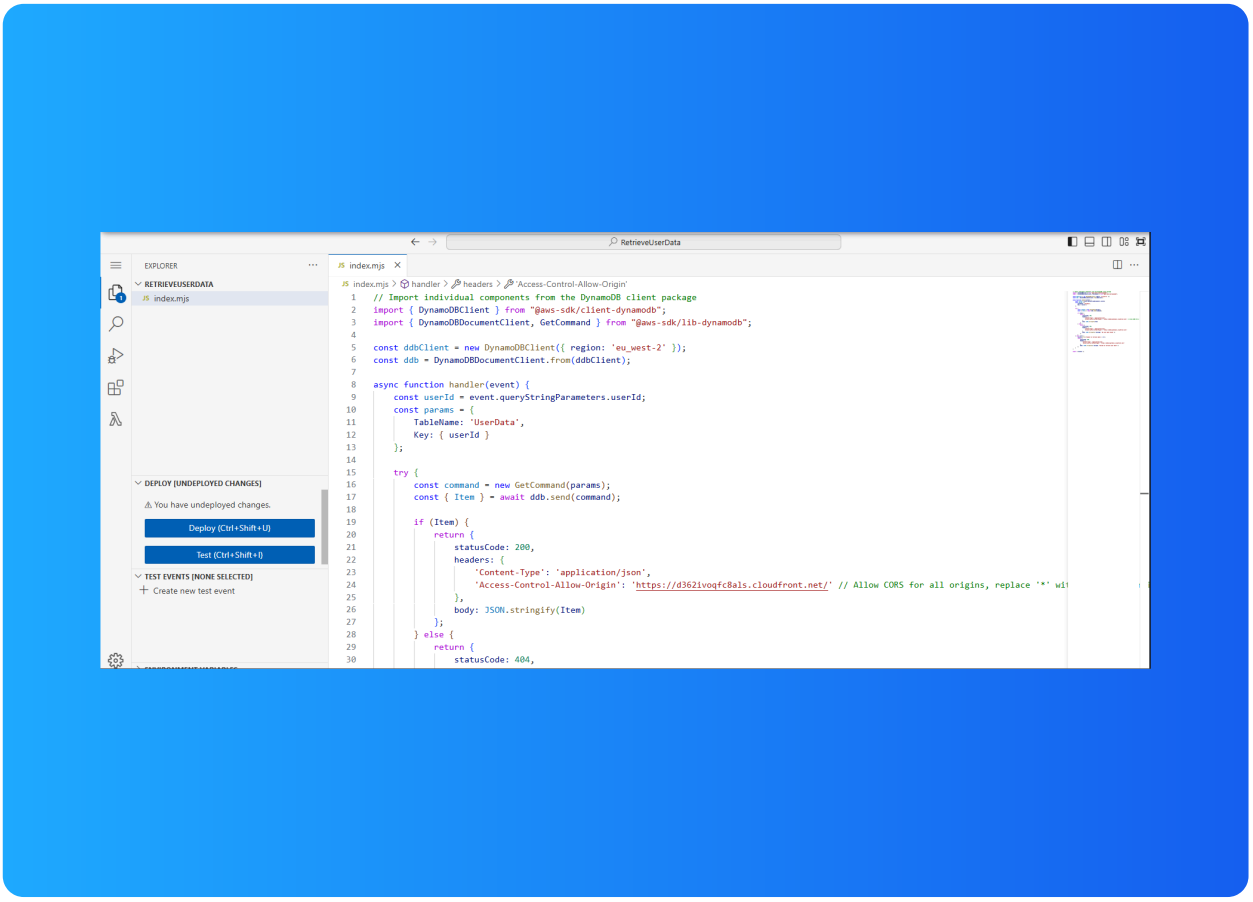


TA
tahirgroot@gmail.com
NextWork Student

Resolving CORS Errors

To resolve the CORS error, I first went into my API(API GATEWAY) and enabled CORS on the /users resource. I then made sure GET requests are enabled, and referenced my CloudFront domain as the domain getting access.

I also updated my Lambda function because it needs to be able to return CORS headers to show that it has the permission to invoke the API's invoke URL and return a response. This means adding 'Access-Control-Allow-Origin' as a header in the response.



Fixed Solution

I verified the fixed connection between API Gateway and CloudFront by looking up user data in the distributed site again. In my final test user data could be returned - so a user request in the presentation tier gets data from the data tier.

Everyone should be in a job they love.

Check out nextwork.org for more projects