**Project Report: Build and deploy Node.js app to Azure Web App – GraphQL-Team3**

**Author:** Alberto Gomez

**Organization:** CIT 262 – Team 3

**Date:** 03/28/2024

**Abstract:** With the use of GraphQL and Prisma as part of the API layer, this project builds and deploys Node.js application to the Azure Web App service. Using defined data structures for building specific data to be queried. Our project uses the benefits of GraphQL for efficient data querying and used Prisma for the database management.

This approach to data querying allows fetching of data with precision and not overproducing data or duplicating search results. It also allows for more efficient data organization and makes the storage of the data more efficient and easier to access. In our particular project we use two types of tasks, Query and Mutation. Query allows for the creation of data, and the Mutation allows for the modification of our created data.

CREATING DATA / QUERY MODIFY DATA / MUTATIONS

# define a type for Task # define the root mutation type

type Task { type Mutation {

id: ID! # create a new task

title: String! createTask(

description: String description: String

title: String! status: Status

status: Status! dueDate: Date

dueDate: Date! ): Task!

}

# update an existing task by id

# define the root query type updateTask(

type Query { id: ID!

# get all tasks title: String!

tasks: [Task!]! description: String!

status: Status!

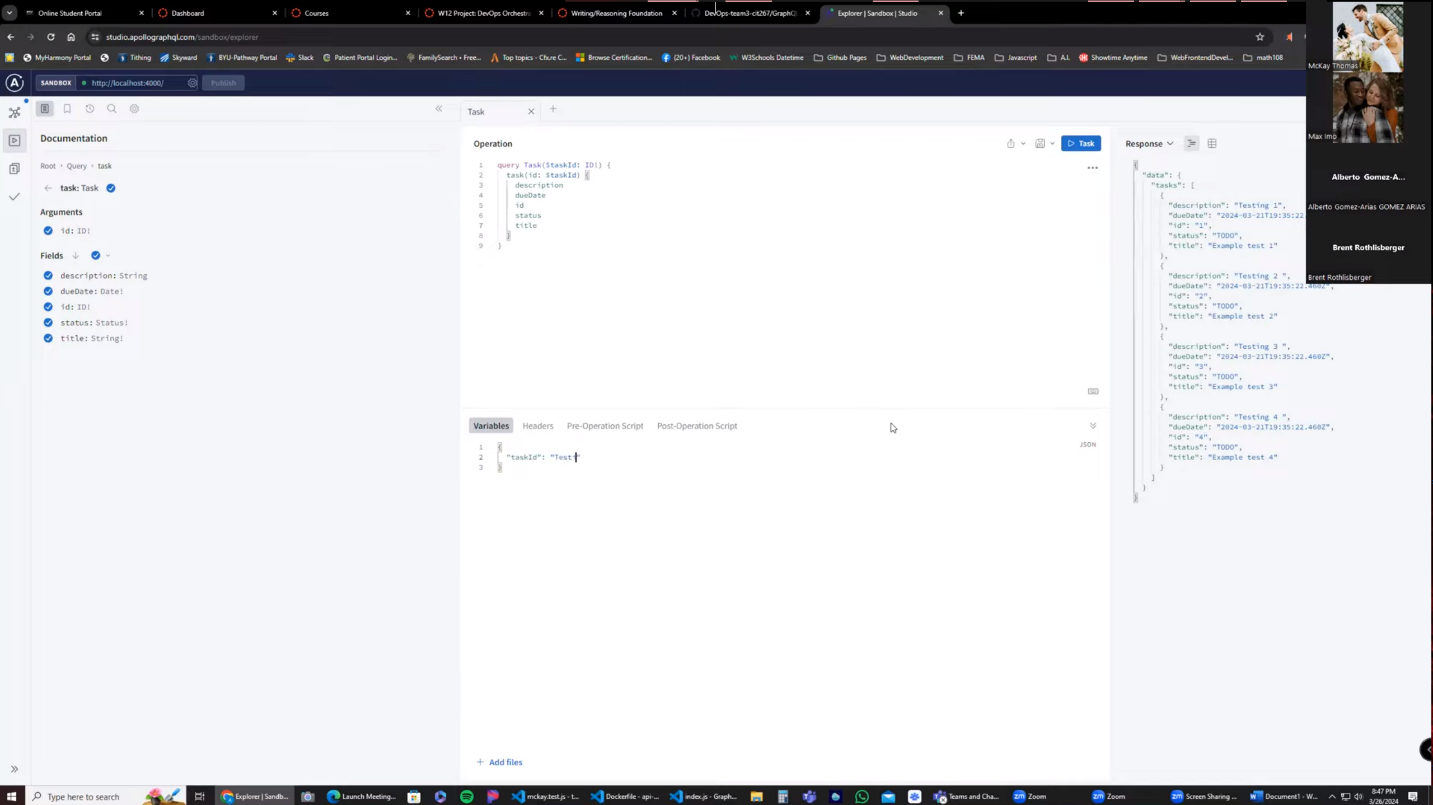
# get a single task by id dueDate: Date!

task(id: ID!): Task ): Task!

} # delete an existing task by id

deleteTask(id: ID!): Boolean!

  }

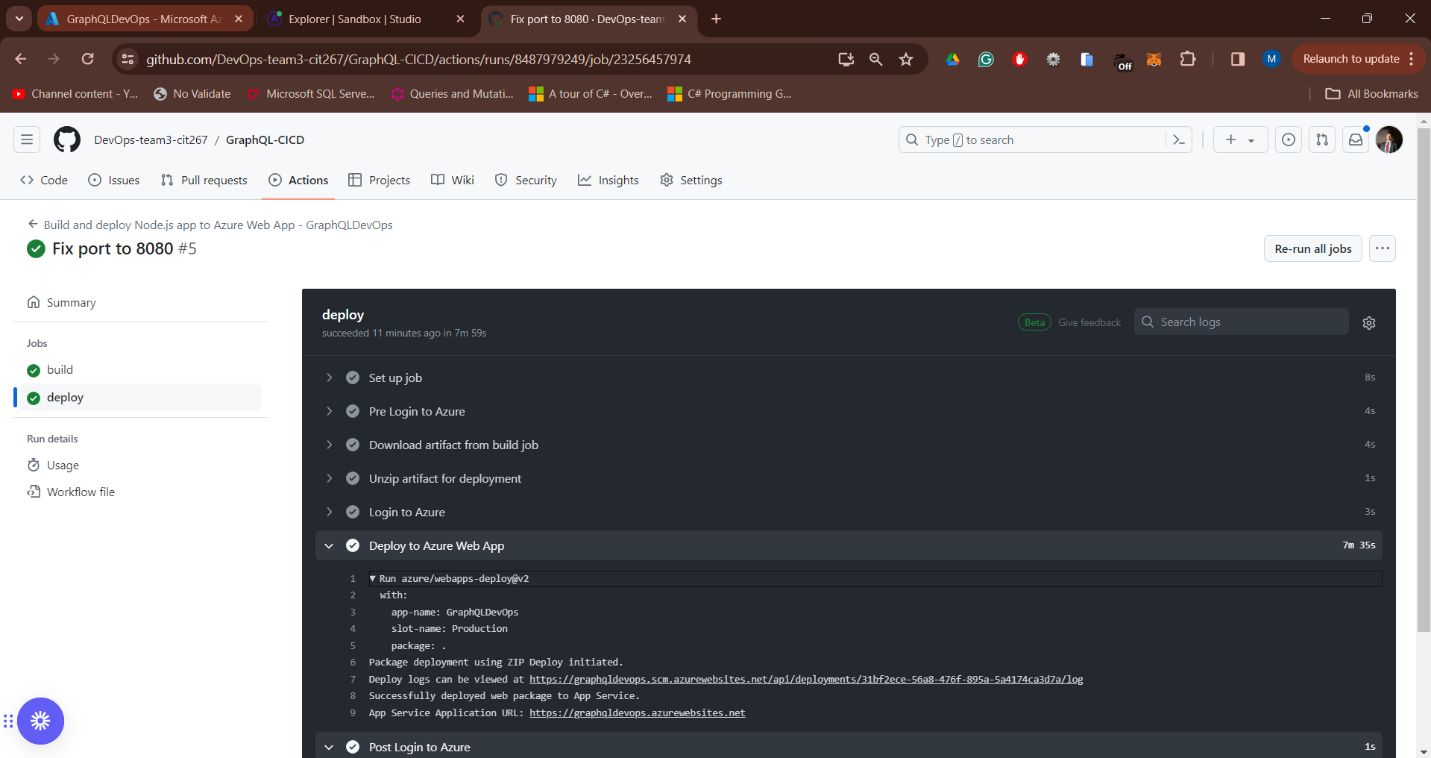
This is an example of our data query.

In real world scenarios, the combined use of GraphQL and Prisma through the Azure Web App can be inplemented through many commercial applications as well as home applications for anyone requiring organization and data management. In stores for example, it can be used for storing product inventory with things like names, item number, re-order numbers, etc. In a home setting, someone who wishes to be more organized can catalog purchases by date and time and or bills, etc. All of these items can be accessed quickly by any type of name or id that is set up, even by category or anything imaginable or anything that helps said users. All of this can be done locally, but by using the Azure Web App, It makes such an application easily scallable and deployable to any type of user.

In conclusion, this type of technology stack allows for a scalable and easy solution for building applications. The ease of use and reliability of such data control allows for easy integration with any type of application that requires organization and data management. The following are links to the GitHub Repository, the delployment app to azzure along with screenshots.

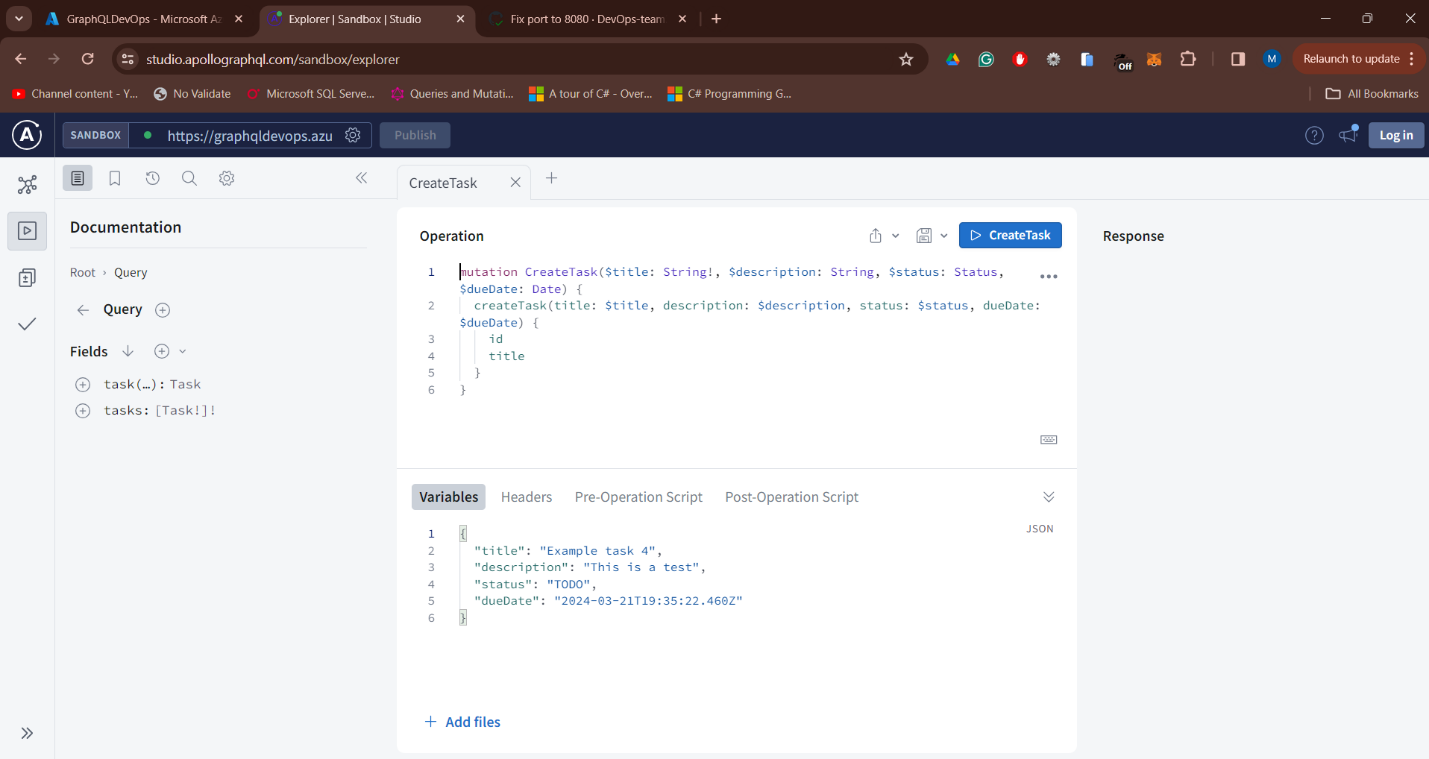
Github Repository

<https://github.com/DevOps-team3-cit267/GraphQL-CICD>



Deployment to Azure

<https://studio.apollographql.com/sandbox/explorer>



Successful Deployment on Microsoft Azure

