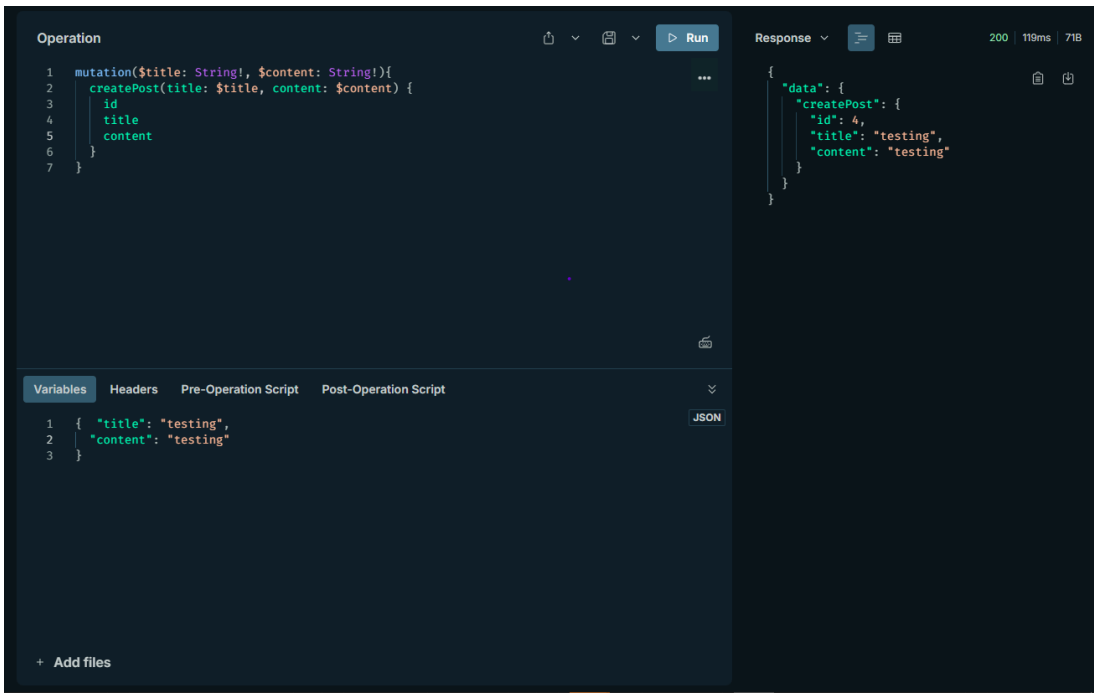
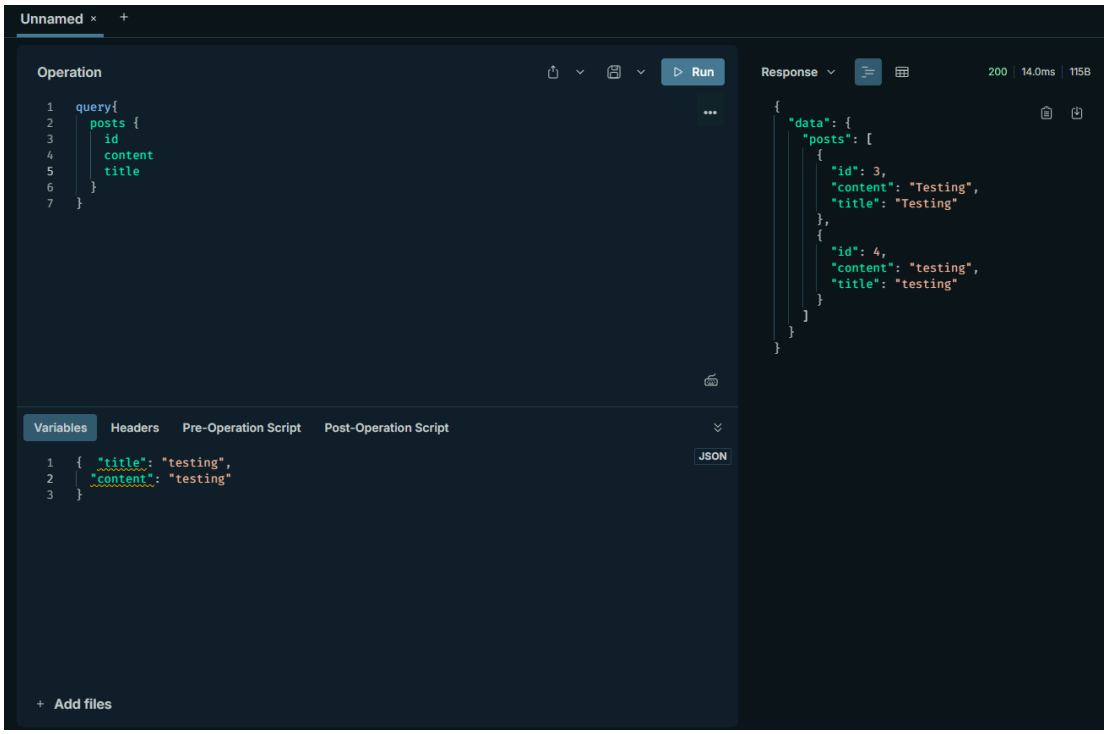
Ethan Daniel S. Saballa 03/15/2025

Post-service

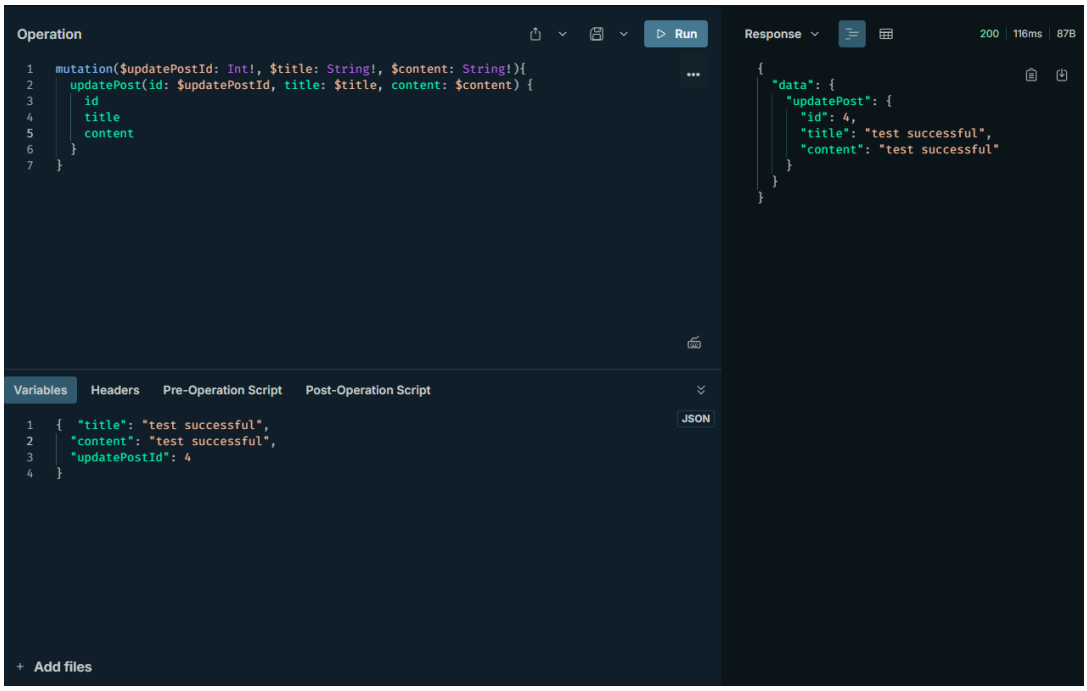
Create:

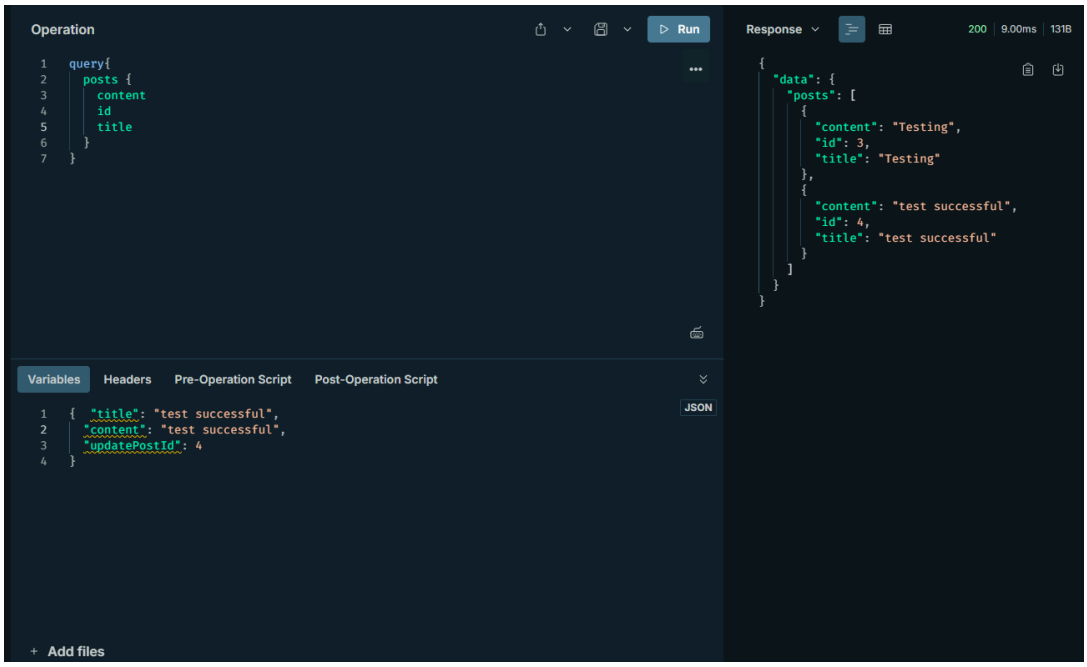


Read:

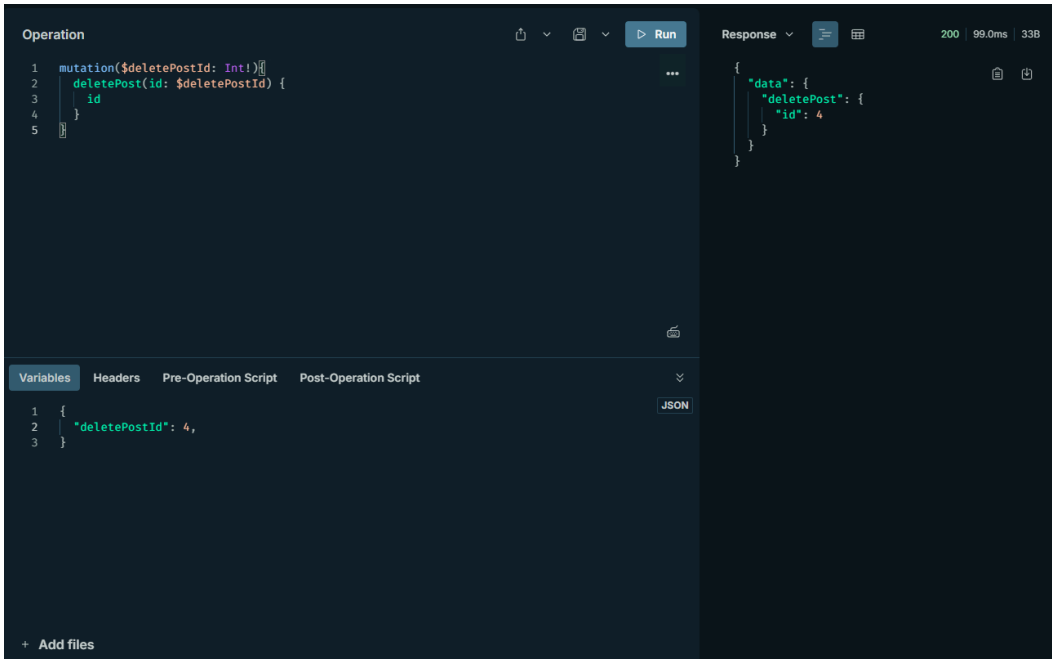


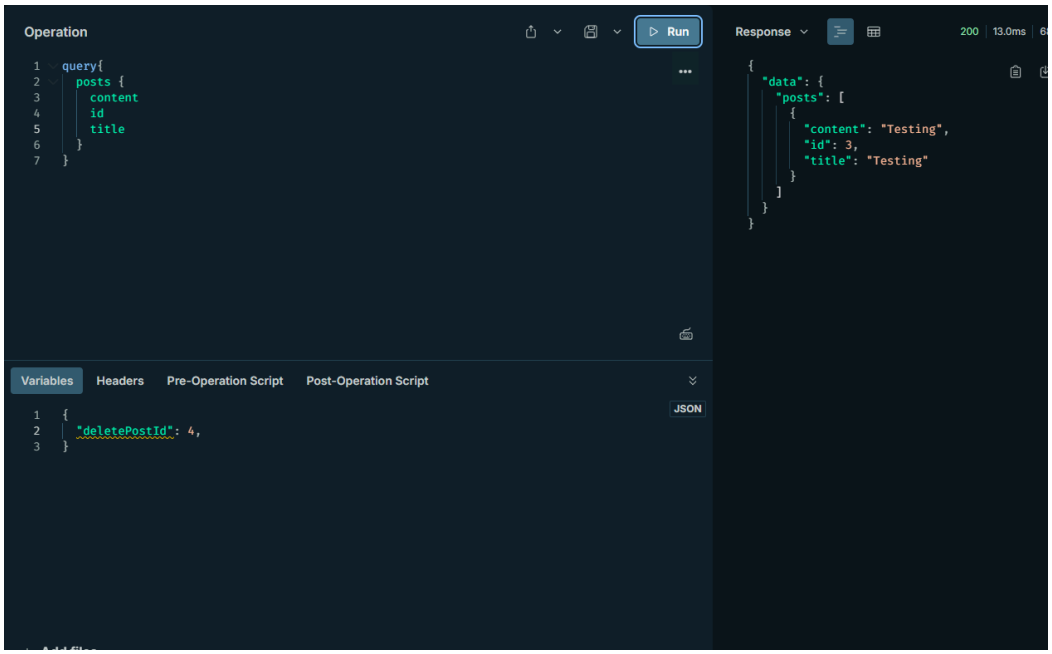
Update:


Operation
Run
mutation($updatePostId: Int!, $title: String!, $content: String!){
updatePost(id: $updatePostId, title: $title, content: $content) {
id
title
content
}
}
Response
200 | 116ms | 87B
{
"data": {
"updatePost": {
"id": 4,
"title": "test successful",
"content": "test successful"
}
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{ "title": "test successful",
"content": "test successful",
"updatePostId": 4
}
Add files


Operation
Run
query{
posts {
content
id
title
}
}
Response
200 | 9.00ms | 131B
{
"data": {
"posts": [
{
"content": "Testing",
"id": 3,
"title": "Testing"
},
{
"content": "test successful",
"id": 4,
"title": "test successful"
}
]
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{ "title": "test successful",
"content": "test successful",
"updatePostId": 4
}
Add files

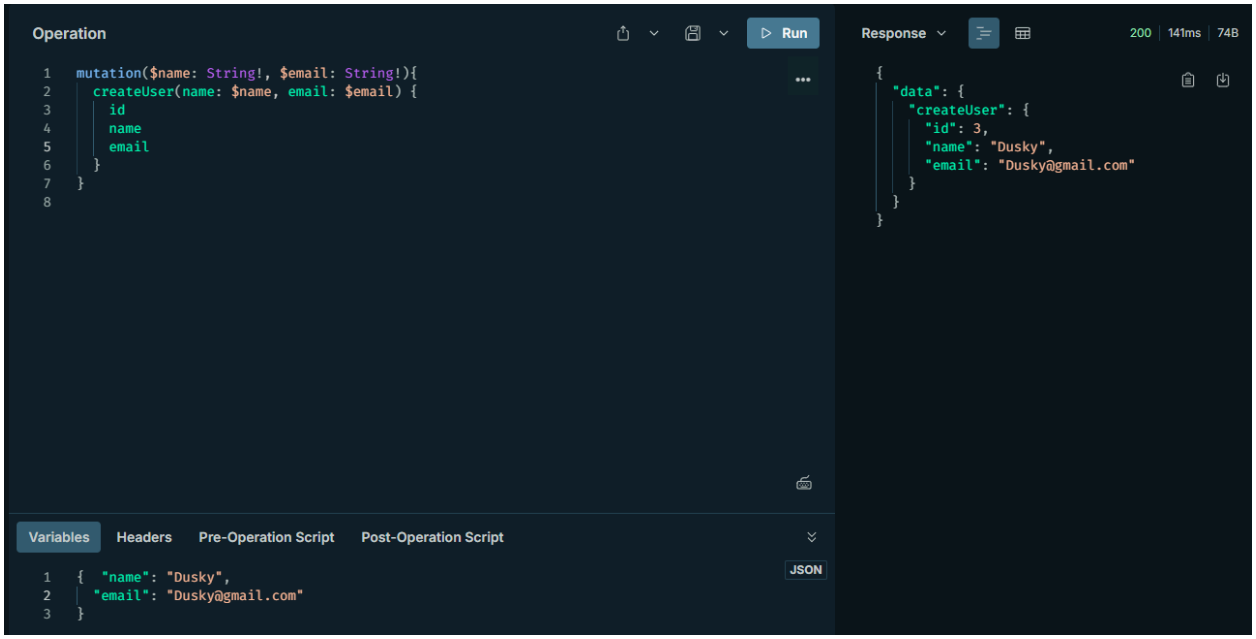
Delete:


Operation
Run
mutation($deletePostId: Int!){
deletePost(id: $deletePostId) {
id
}
}
Response
200 | 99.0ms | 33B
{
"data": {
"deletePost": {
"id": 4
}
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"deletePostId": 4,
}
Add files

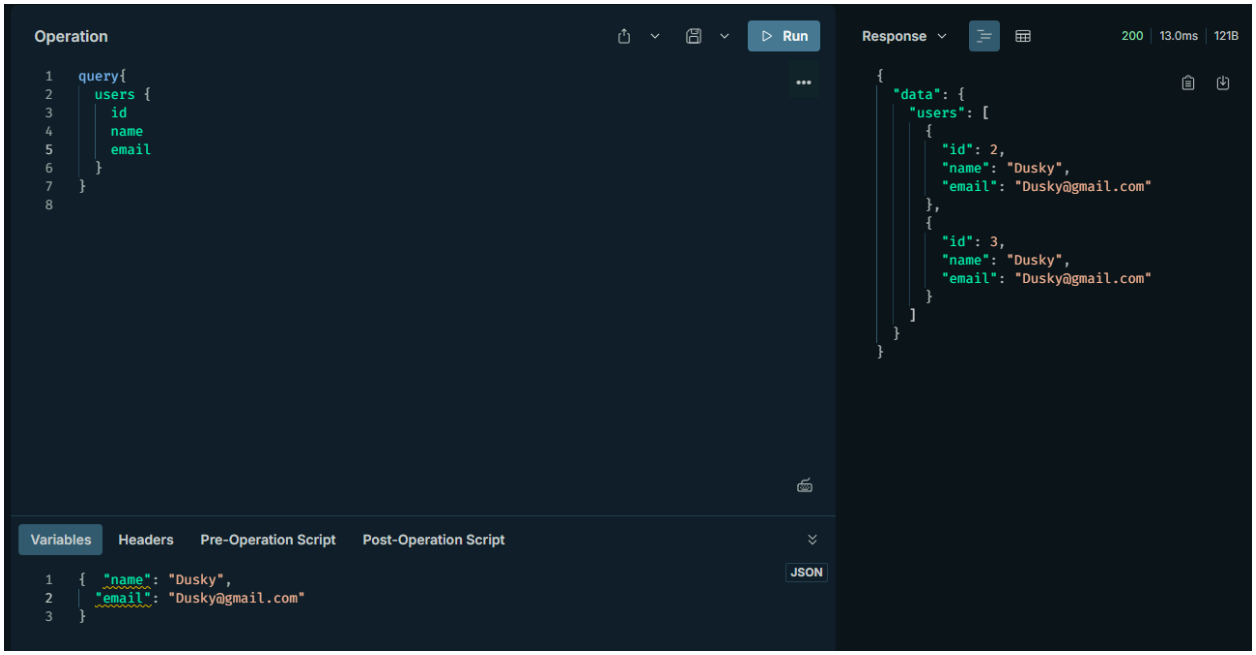

Operation
Run
query{
posts {
content
id
title
}
}
Response
200 | 13.0ms
{
"data": {
"posts": [
{
"content": "Testing",
"id": 3,
"title": "Testing"
}
]
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"deletePostId": 4,
}

Users-Table

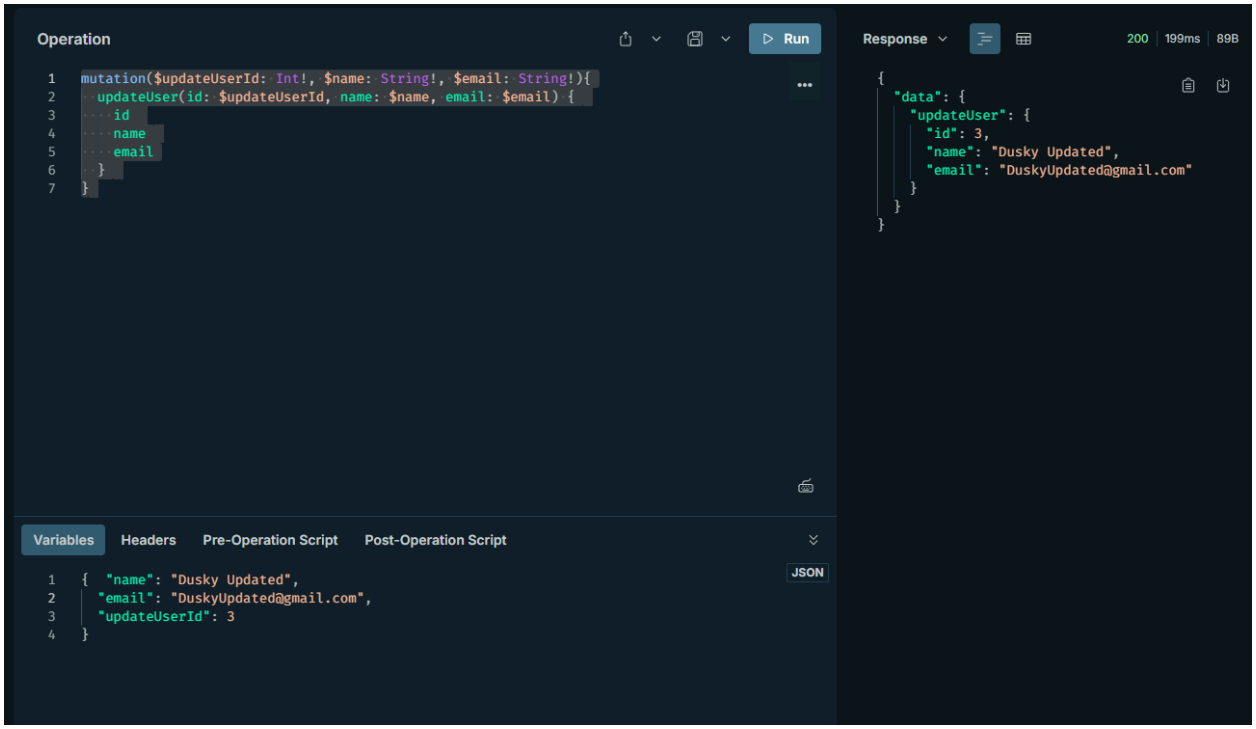
Create:

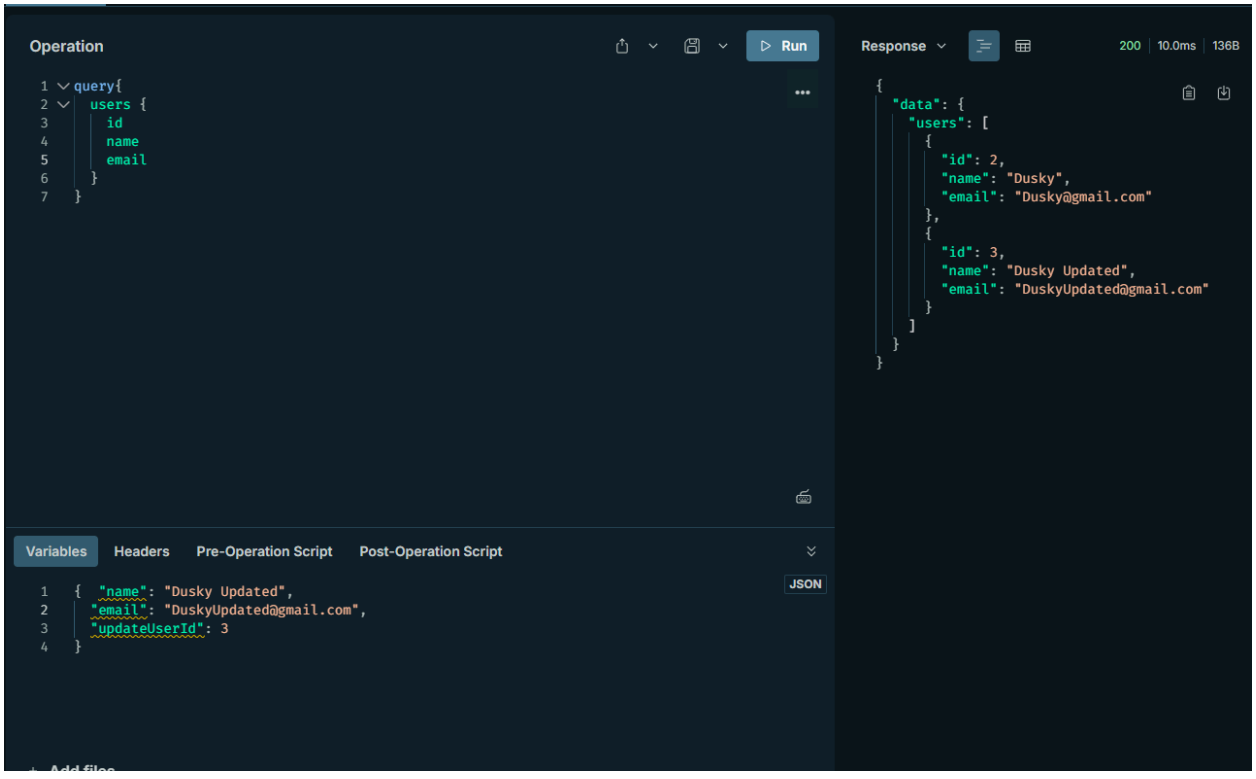


Read:

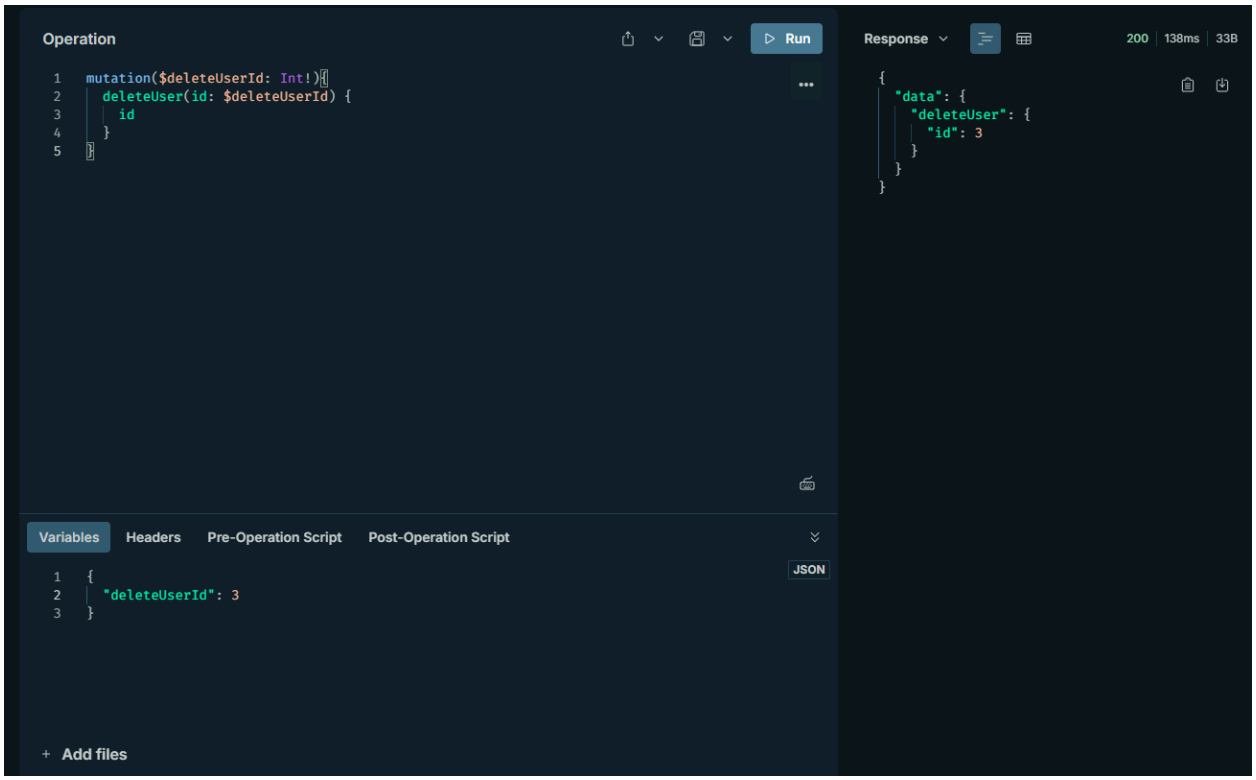


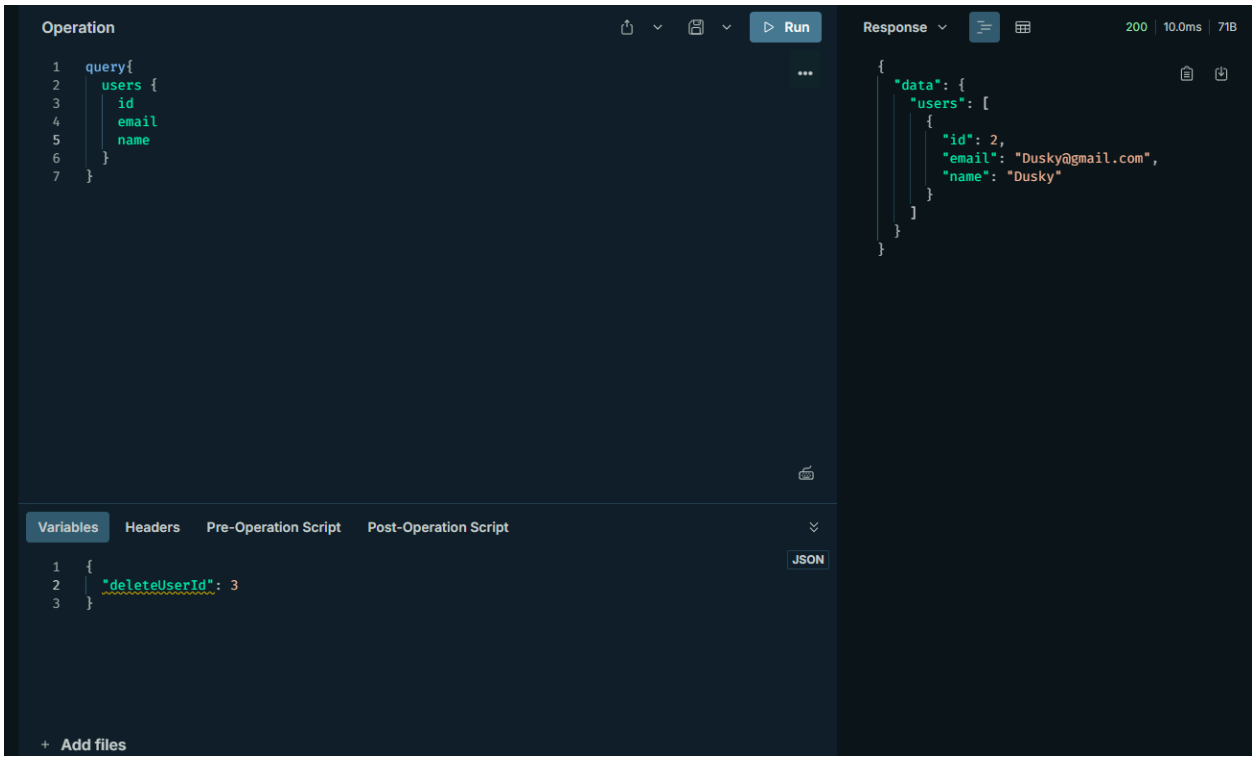
Update:


Operation
Run
mutation($updateUserId: Int!, $name: String!, $email: String!){
updateUser(id: $updateUserId, name: $name, email: $email) {
id
name
email
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{ "name": "Dusky Updated",
"email": "DuskyUpdated@gmail.com",
"updateUserId": 3
}
Response
200 | 199ms | 89B
{
"data": {
"updateUser": {
"id": 3,
"name": "Dusky Updated",
"email": "DuskyUpdated@gmail.com"
}
}
}


Operation
Run
query{
users {
id
name
email
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{ "name": "Dusky Updated",
"email": "DuskyUpdated@gmail.com",
"updateUserId": 3
}
Response
200 | 10.0ms | 136B
{
"data": {
"users": [
{
"id": 2,
"name": "Dusky",
"email": "Dusky@gmail.com"
},
{
"id": 3,
"name": "Dusky Updated",
"email": "DuskyUpdated@gmail.com"
}
]
}
}

Delete:


Operation
Run
mutation($deleteUserId: Int!){
deleteUser(id: $deleteUserId) {
id
}
}
Response
200 | 138ms | 33B
{
"data": {
"deleteUser": {
"id": 3
}
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"deleteUserId": 3
}
Add files


Operation
Run
query{
users {
id
email
name
}
}
Response
200 | 10.0ms | 71B
{
"data": {
"users": [
{
"id": 2,
"email": "Dusky@gmail.com",
"name": "Dusky"
}
]
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"deleteUserId": 3
}
Add files

- What do database migrations do and why are they useful?

  Based from my understanding, database migrations is basically described as a “snapshot” in which it captures the state of what your database is in that time. Each migration then builds on the previous whenever there should be any changes or additions to the database. Migrations are useful because it keeps track of the historical versions of each database. This also helps in case you want to go back to a previous version of the database.

- How does GraphQL differ from REST for CRUD operations?

  GraphQL offers a single endpoint that allows clients to request exactly the data they need, reducing over-fetching or under-fetching. In contrast, REST typically involves multiple endpoints with fixed data responses, which can lead to inefficiencies. GraphQL’s type system also improves client-server communication by ensuring that both parties agree on the data structure.