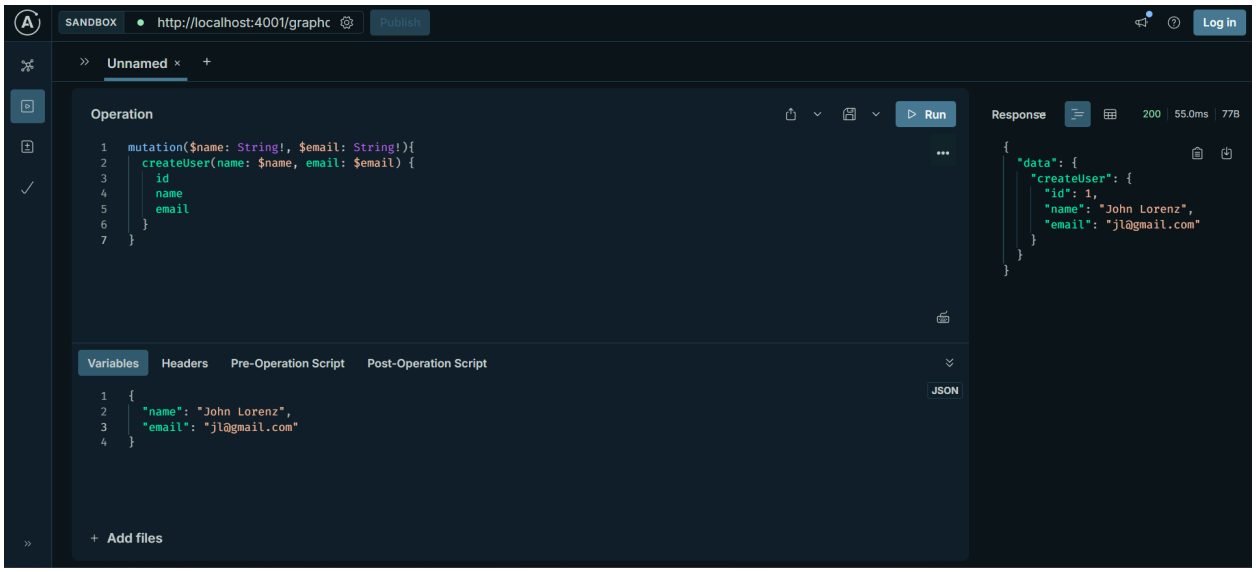
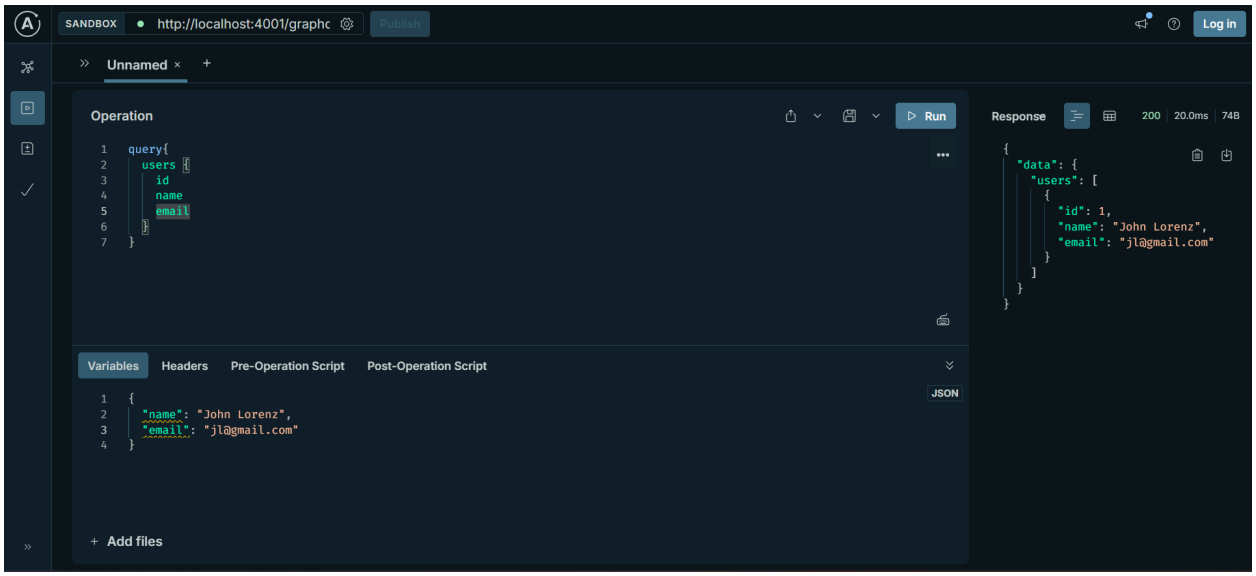
Codilla, John Lorenz 03/15/25

User Service

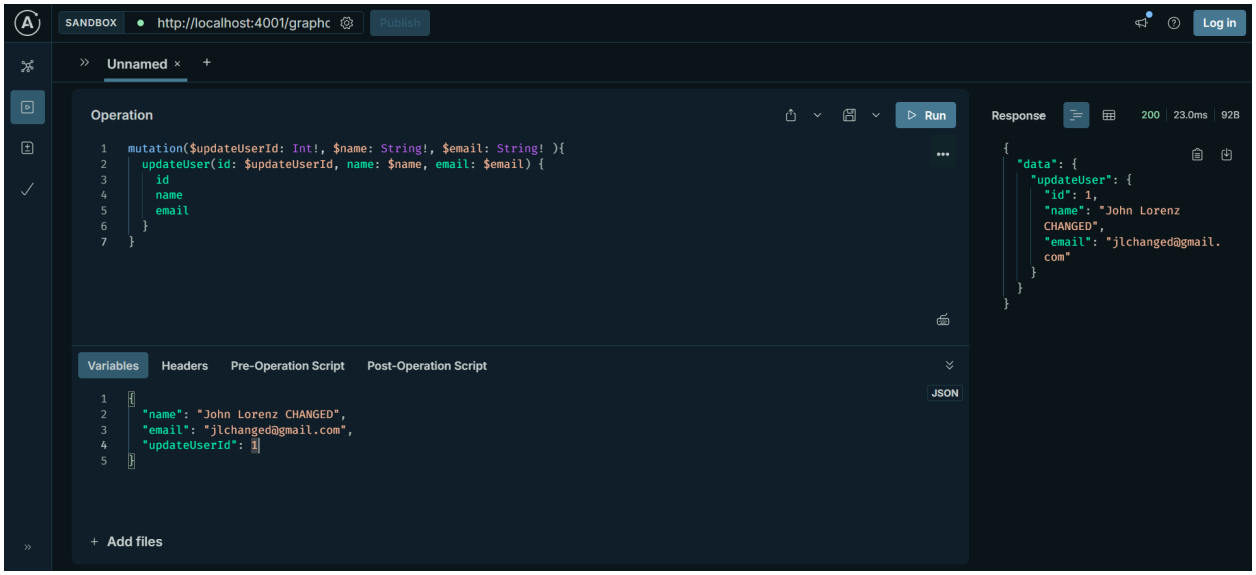
Create:



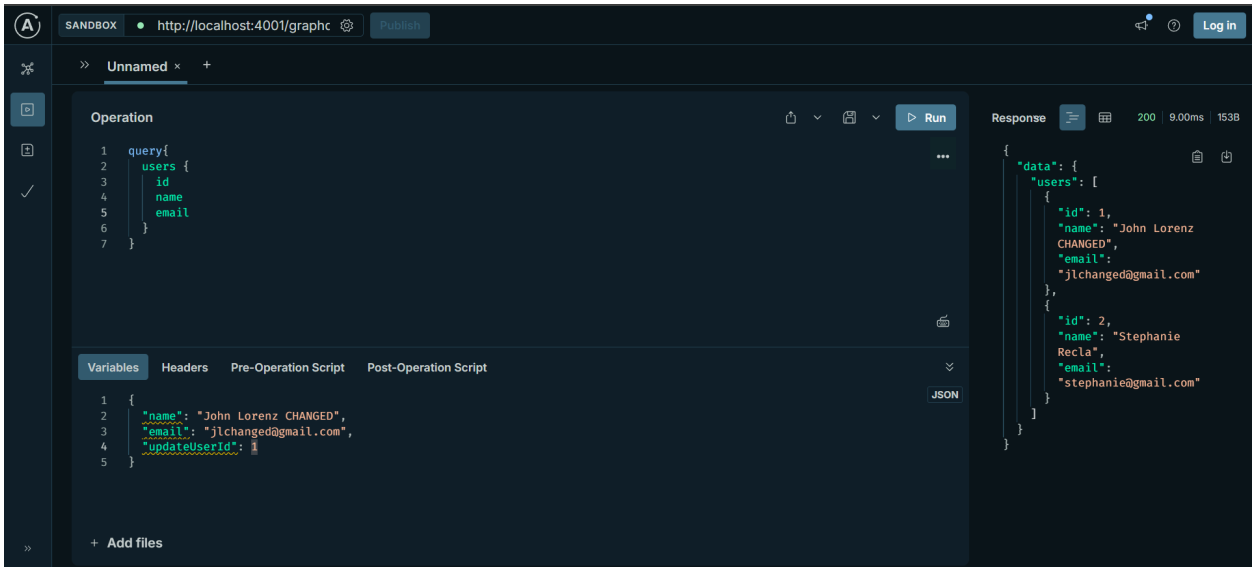
Read:



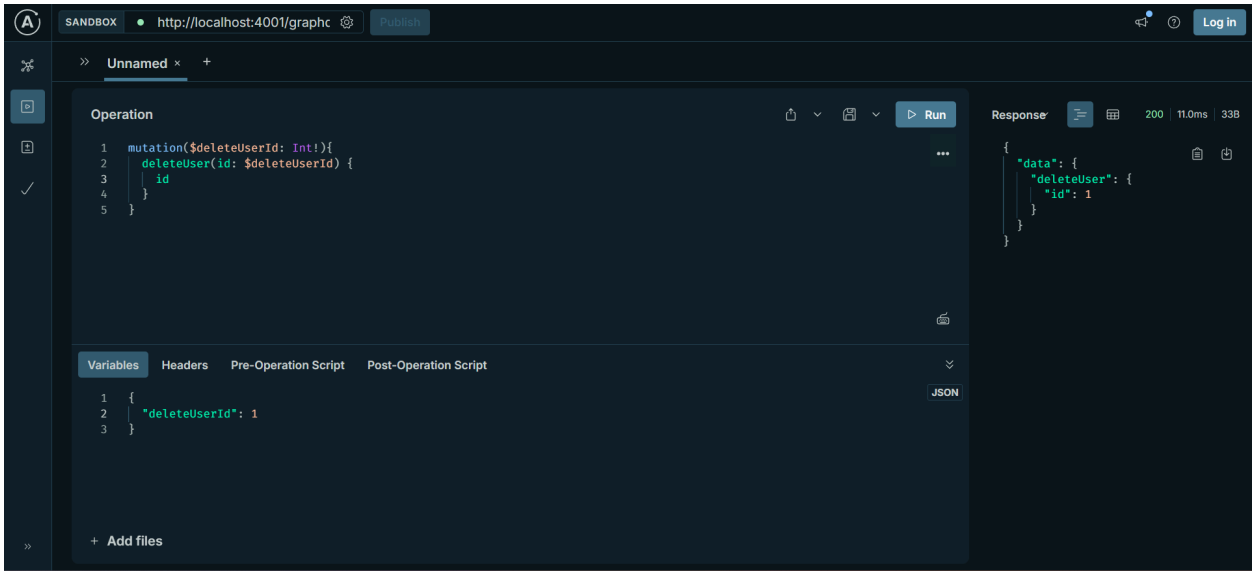
Update:


SANDBOX
http://localhost:4001/graphc
Publish
Log in
Unnamed
Operation
Run
mutation($updateUserId: Int!, $name: String!, $email: String!){
updateUser(id: $updateUserId, name: $name, email: $email) {
id
name
email
}
}
Response
200
23.0ms
92B
{
"data": {
"updateUser": {
"id": 1,
"name": "John Lorenz CHANGED",
"email": "jlchanged@gmail.com"
}
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"name": "John Lorenz CHANGED",
"email": "jlchanged@gmail.com",
"updateUserId": 1
}
Add files

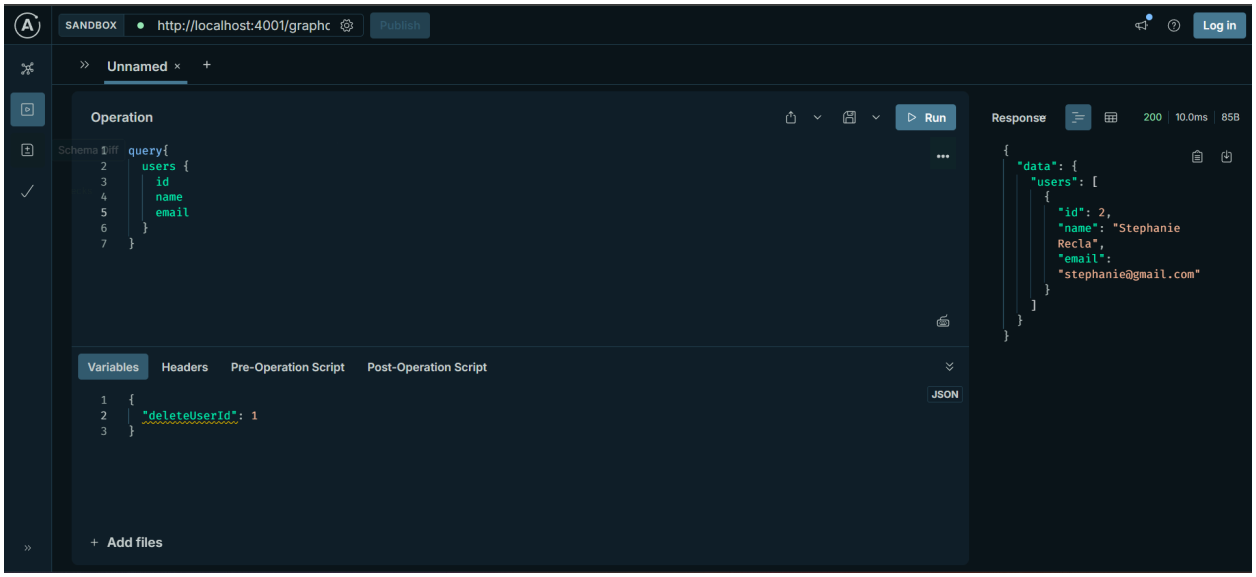
Read:


SANDBOX
http://localhost:4001/graphc
Publish
Log in
Unnamed
Operation
Run
query{
users {
id
name
email
}
}
Response
200
9.00ms
153B
{
"data": {
"users": [
{
"id": 1,
"name": "John Lorenz CHANGED",
"email": "jlchanged@gmail.com"
},
{
"id": 2,
"name": "Stephanie Recla",
"email": "stephanie@gmail.com"
}
]
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"name": "John Lorenz CHANGED",
"email": "jlchanged@gmail.com",
"updateUserId": 1
}
Add files

Delete:

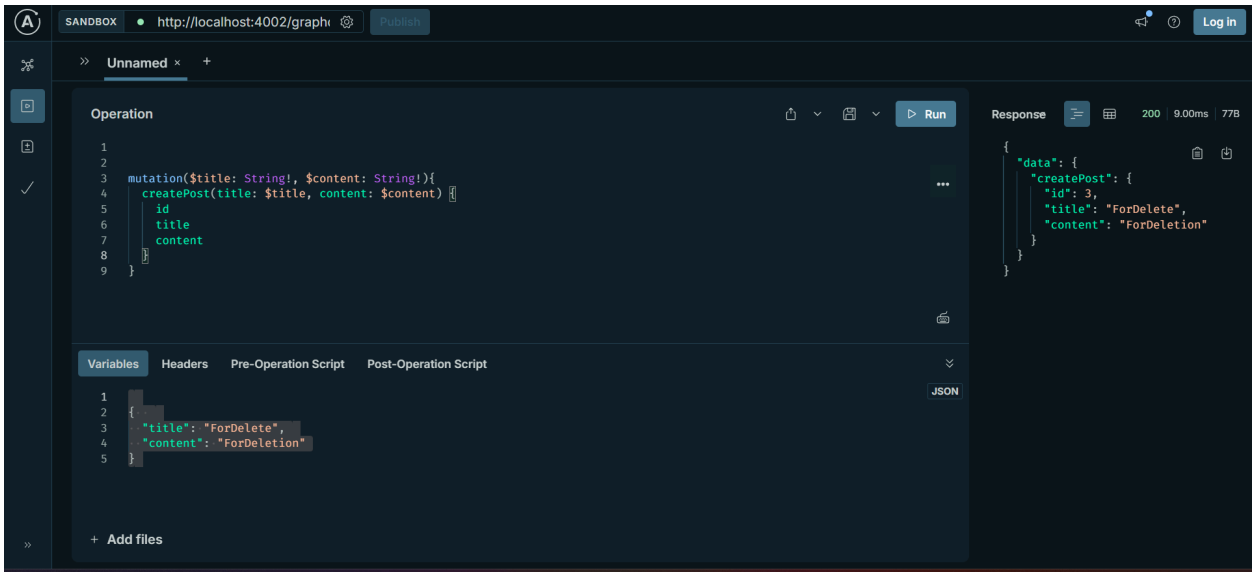

SANDBOX
http://localhost:4001/graphc
Publish
Log in
Unnamed
Operation
Run
mutation($deleteUserId: Int!){
deleteUser(id: $deleteUserId) {
id
}
}
Response
200
11.0ms
338
{
"data": {
"deleteUser": {
"id": 1
}
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"deleteUserId": 1
}
Add files

Read:

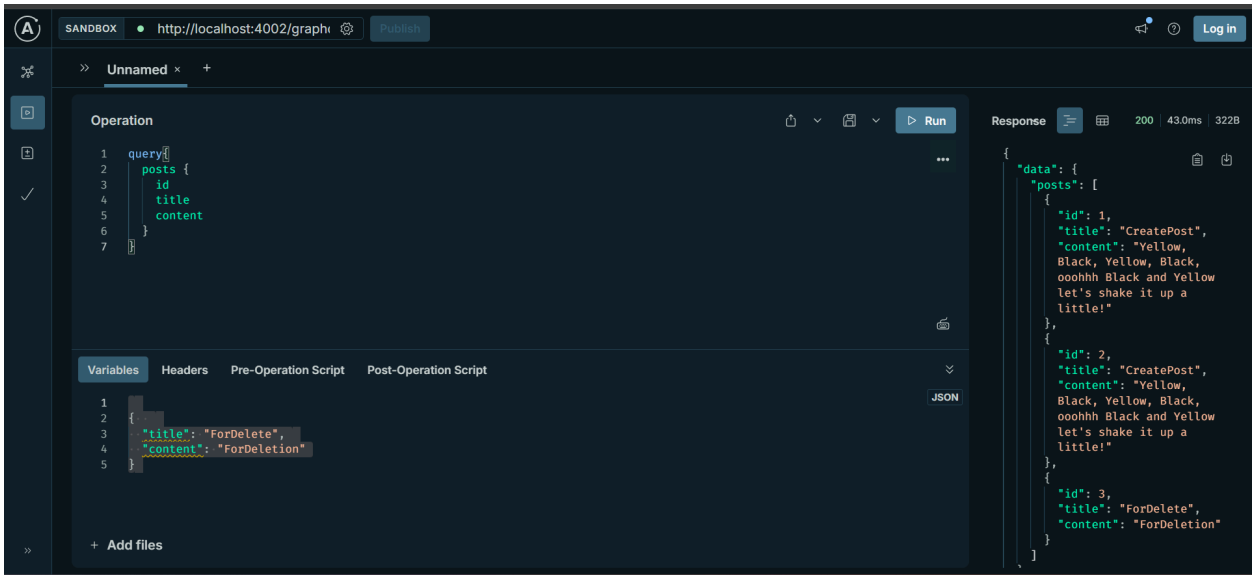

SANDBOX
http://localhost:4001/graphc
Publish
Log in
Unnamed
Operation
Run
query{
users {
id
name
email
}
}
Response
200
10.0ms
858
{
"data": {
"users": [
{
"id": 2,
"name": "Stephanie Recla",
"email": "stephanie@gmail.com"
}
]
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"deleteUserId": 1
}
Add files

Post Service

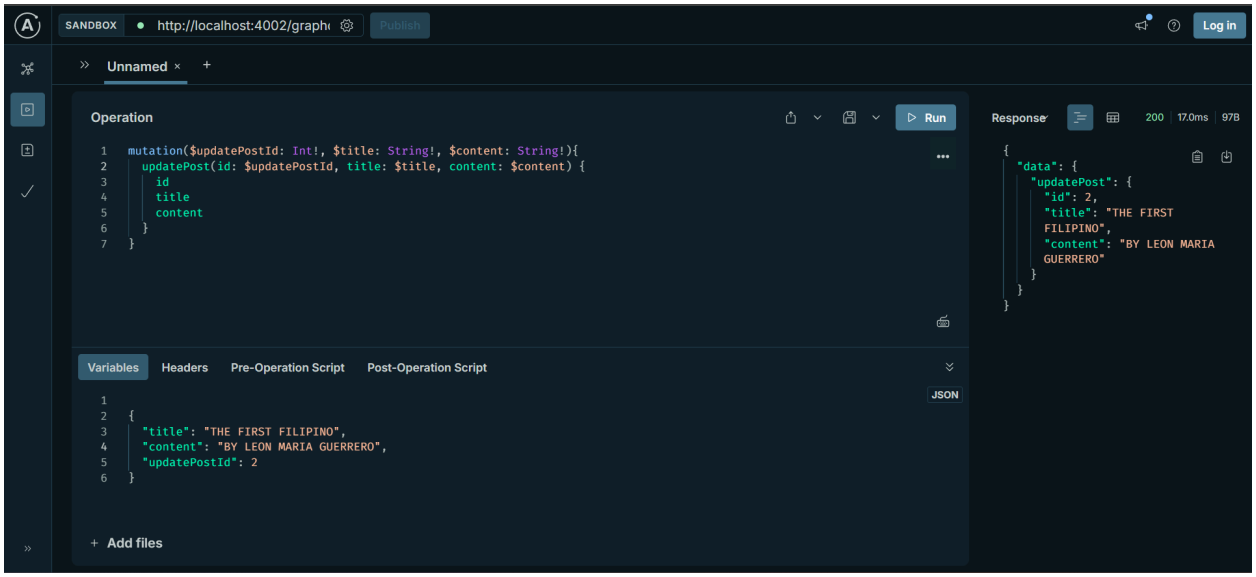
Create:



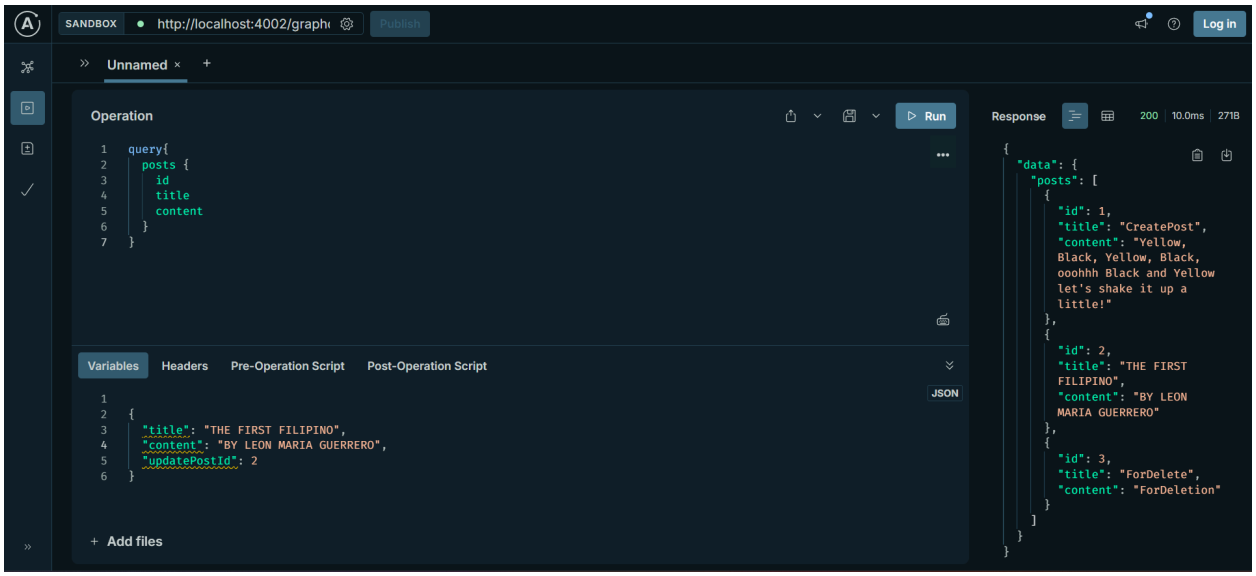
Read:



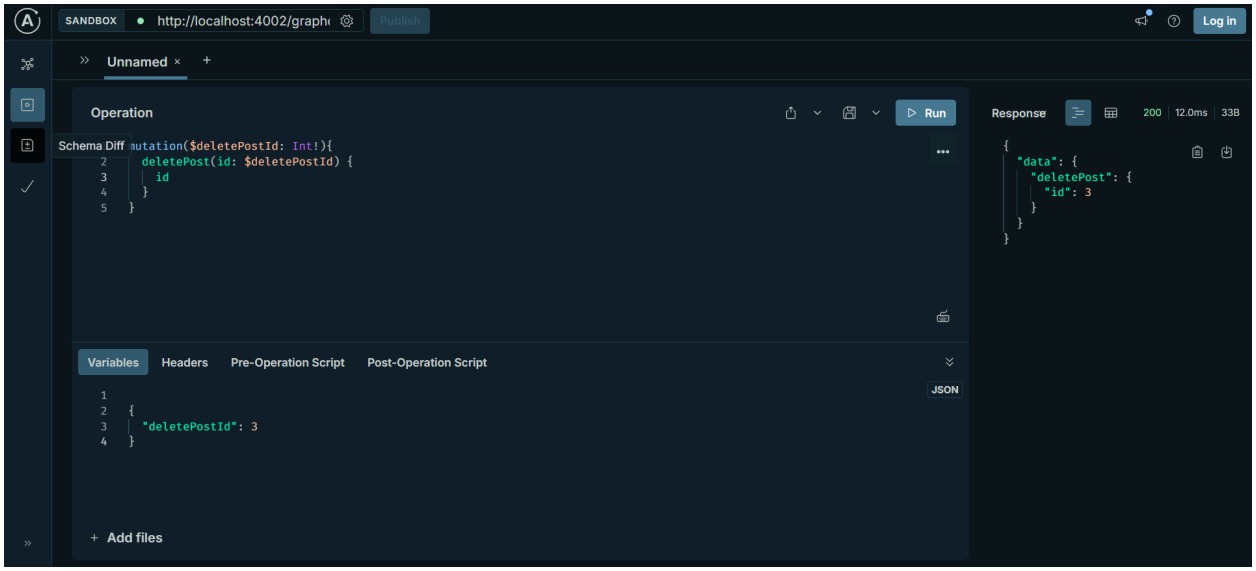
Update:


SANDBOX
http://localhost:4002/graph
Publish
Log in
Unnamed
Operation
Run
mutation($updatePostId: Int!, $title: String!, $content: String!){
updatePost(id: $updatePostId, title: $title, content: $content) {
id
title
content
}
}
Response
200
17.0ms
97B
{
"data": {
"updatePost": {
"id": 2,
"title": "THE FIRST FILIPINO",
"content": "BY LEON MARIA GUERRERO"
}
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"title": "THE FIRST FILIPINO",
"content": "BY LEON MARIA GUERRERO",
"updatePostId": 2
}
Add files

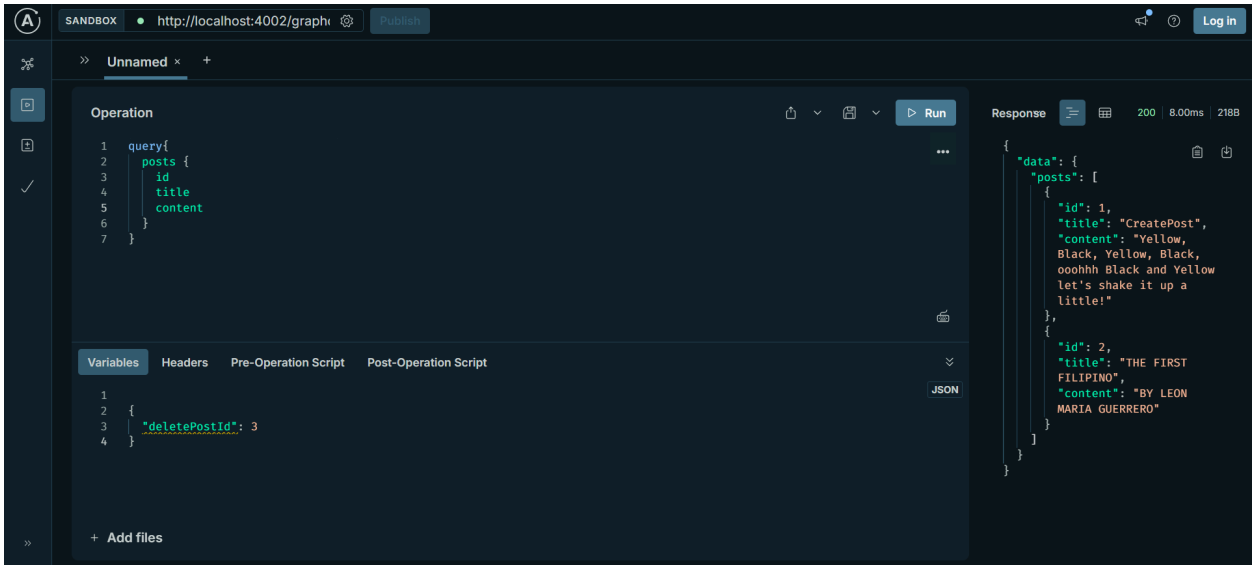
Read:


SANDBOX
http://localhost:4002/graph
Publish
Log in
Unnamed
Operation
Run
query{
posts {
id
title
content
}
}
Response
200
10.0ms
271B
{
"data": {
"posts": [
{
"id": 1,
"title": "CreatePost",
"content": "Yellow, Black, Yellow, Black, ooohhh Black and Yellow let's shake it up a little!"
},
{
"id": 2,
"title": "THE FIRST FILIPINO",
"content": "BY LEON MARIA GUERRERO"
},
{
"id": 3,
"title": "ForDelete",
"content": "ForDeletion"
}
]
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"title": "THE FIRST FILIPINO",
"content": "BY LEON MARIA GUERRERO",
"updatePostId": 2
}
Add files

Delete:


SANDBOX
http://localhost:4002/graph
Publish
Log in
Unnamed
Operation
Run
Schema Diff
utation($deletePostId: Int!){
deletePost(id: $deletePostId) {
id
}
}
Response
200
12.0ms
338B
{
"data": {
"deletePost": {
"id": 3
}
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"deletePostId": 3
}
Add files

Read:


SANDBOX
http://localhost:4002/graph
Publish
Log in
Unnamed
Operation
Run
query{
posts {
id
title
content
}
}
Response
200
8.00ms
218B
{
"data": {
"posts": [
{
"id": 1,
"title": "CreatePost",
"content": "Yellow, Black, Yellow, Black, oooohhh Black and Yellow let's shake it up a little!"
},
{
"id": 2,
"title": "THE FIRST FILIPINO",
"content": "BY LEON MARIA GUERRERO"
}
]
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"deletePostId": 3
}
Add files

Reflection:

What do database migrations do and why are they useful?

- Database migrations are like a set of instructions that help you update your database structure as your application grows. They let you add, change, or remove tables and columns in a controlled way. They're super helpful because they keep your database organized, make it easy to track changes, and ensure everyone on your team is working with the same setup. Plus, if something goes wrong, you can roll back to a previous version.

How does GraphQL differ from REST for CRUD operations?

- GraphQL lets you ask for exactly the data you want in one request, so you don't get extra stuff you don't need. REST, on the other hand, is like ordering from a fixed menu. You have to go to different places like endpoints to get different things, and sometimes you get extra stuff you didn't need. REST can mean more work. With GraphQL, you just use one URL and tell it what you want to do like read, create, or update in the request.