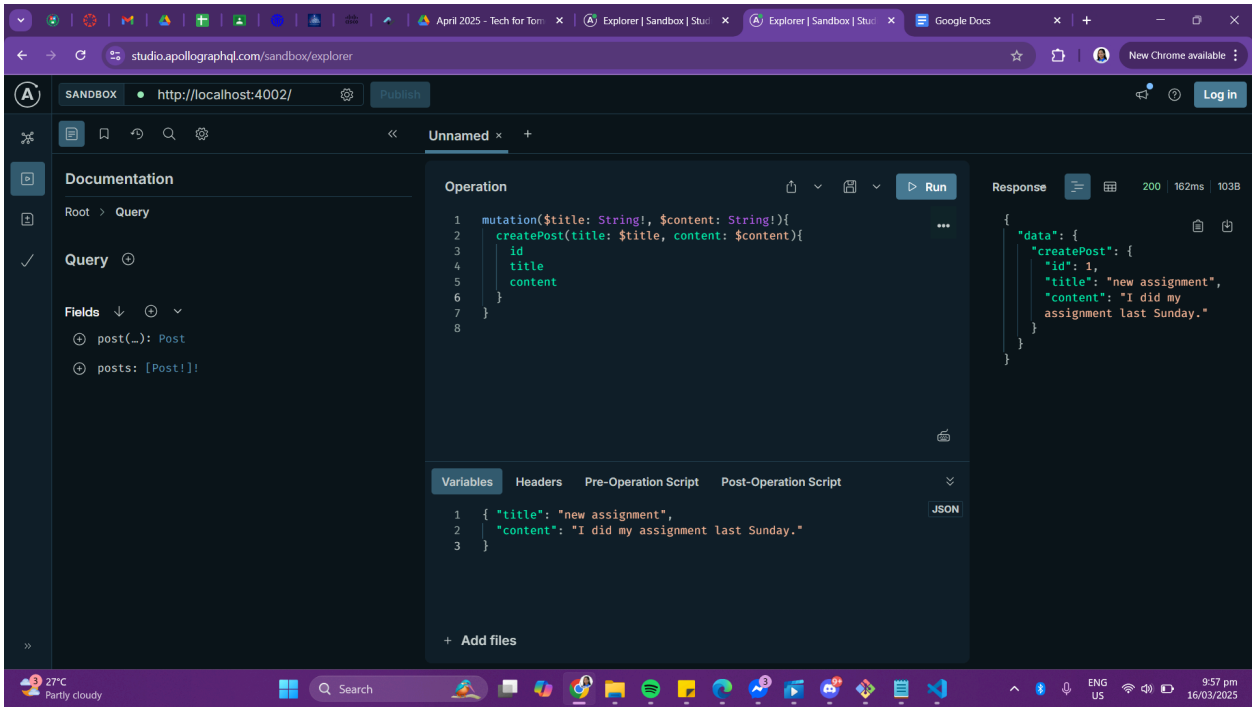
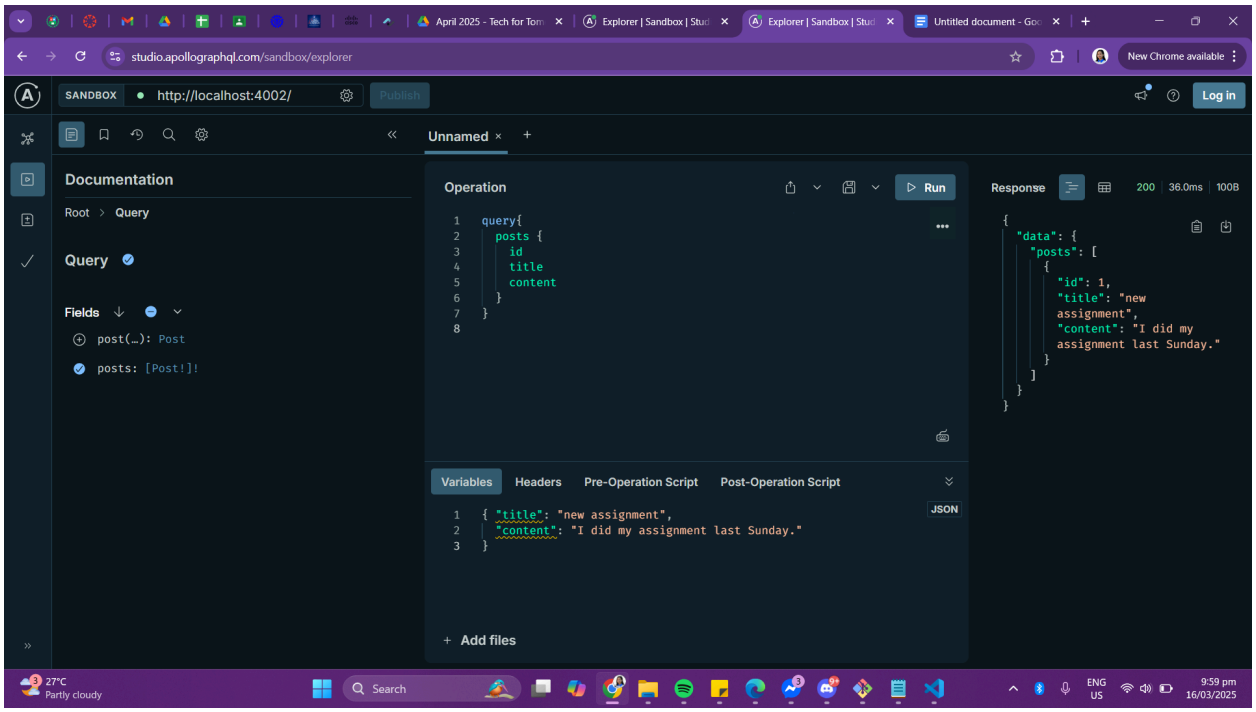
Post-Services

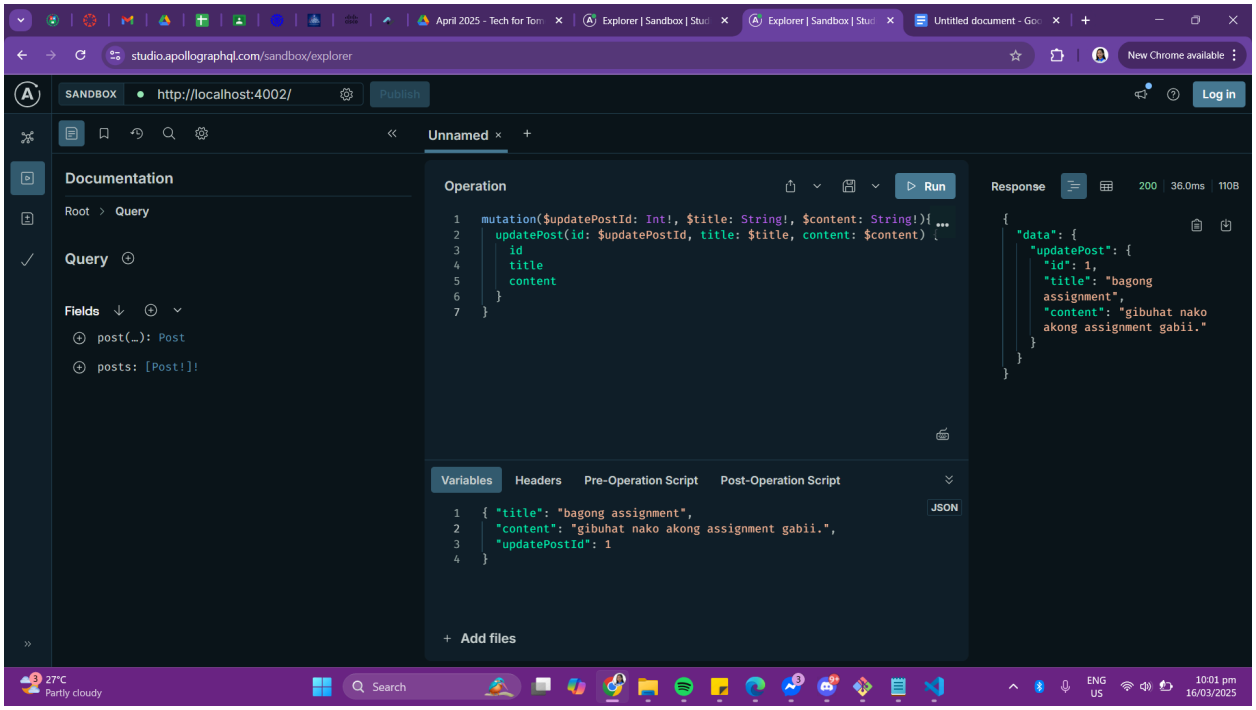
Create



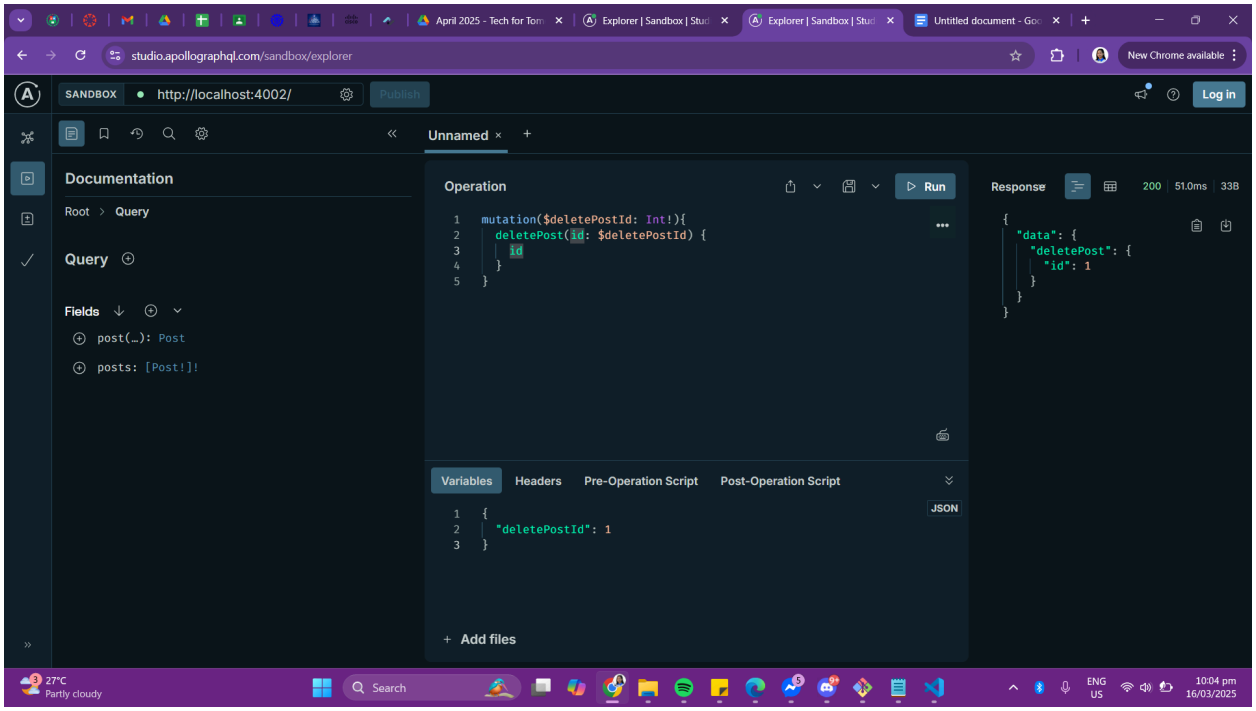
Read



Update

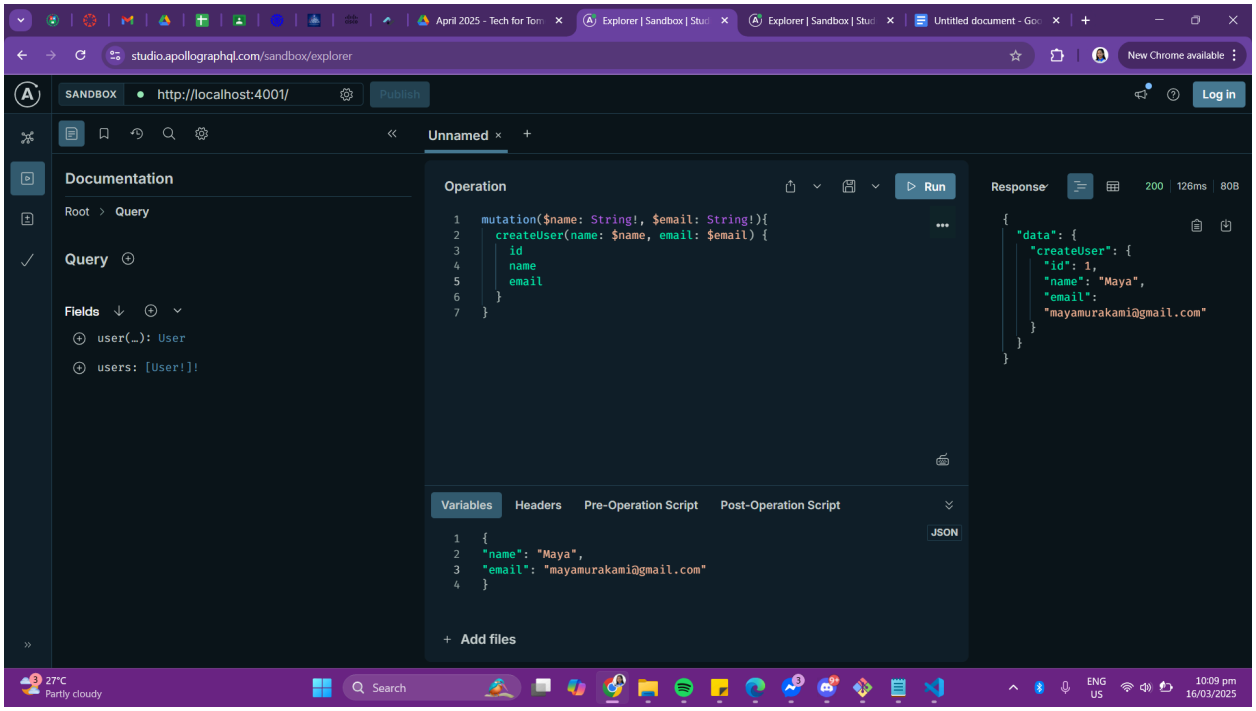

studio.apollographql.com/sandbox/explorer
SANDBOX
http://localhost:4002/
Publish
Log in
Unnamed
Documentation
Root > Query
Query
Fields
post(…): Post
posts: [Post!]!
Operation
Run
mutation($updatePostId: Int!, $title: String!, $content: String!){
updatePost(id: $updatePostId, title: $title, content: $content) {
id
title
content
}
}
Response
200
36.0ms
110B
{
"data": {
"updatePost": {
"id": 1,
"title": "bagong assignment",
"content": "gibuhat nako akong assignment gabii."
}
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{ "title": "bagong assignment",
"content": "gibuhat nako akong assignment gabii.",
"updatePostId": 1
}
Add files
27°C
Partly cloudy
Search
ENG
US
10:01 pm
16/03/2025

Delete

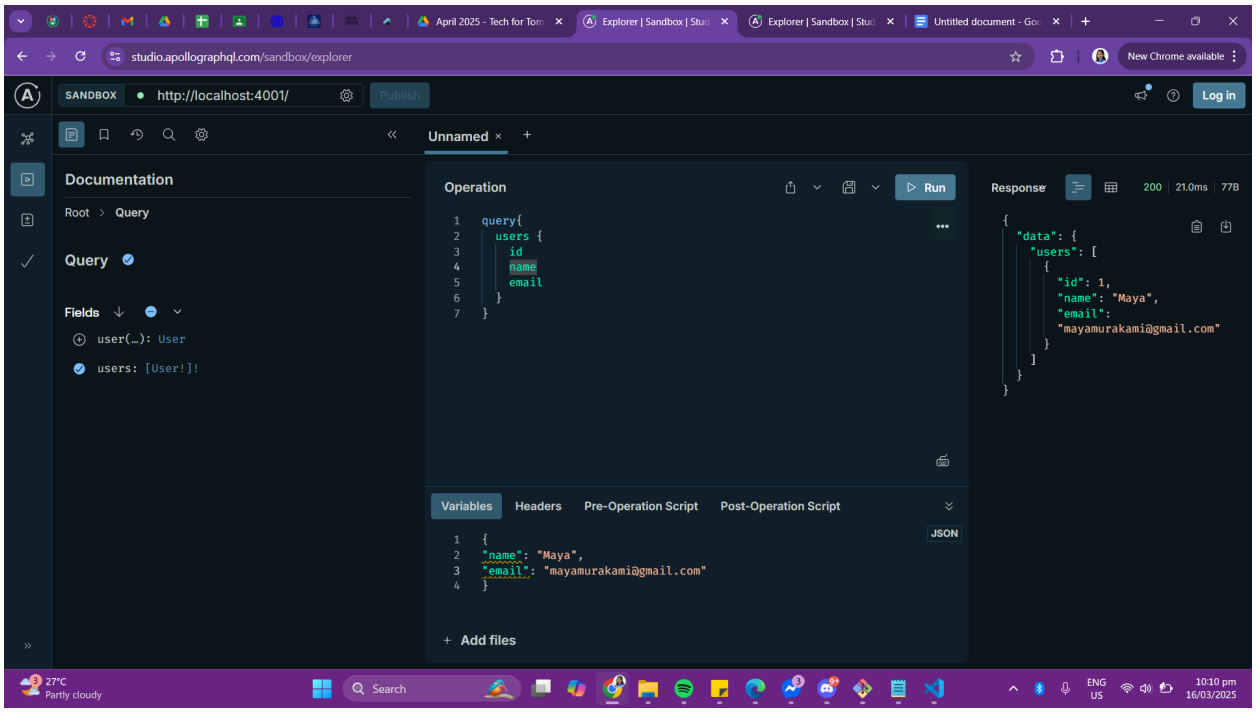

studio.apollographql.com/sandbox/explorer
SANDBOX
http://localhost:4002/
Publish
Log in
Unnamed
Documentation
Root > Query
Query
Fields
post(…): Post
posts: [Post!]!
Operation
Run
mutation($deletePostId: Int!){
deletePost(id: $deletePostId) {
id
}
}
Response
200
51.0ms
33B
{
"data": {
"deletePost": {
"id": 1
}
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"deletePostId": 1
}
Add files
27°C
Partly cloudy
Search
ENG
US
10:04 pm
16/03/2025

User-Services

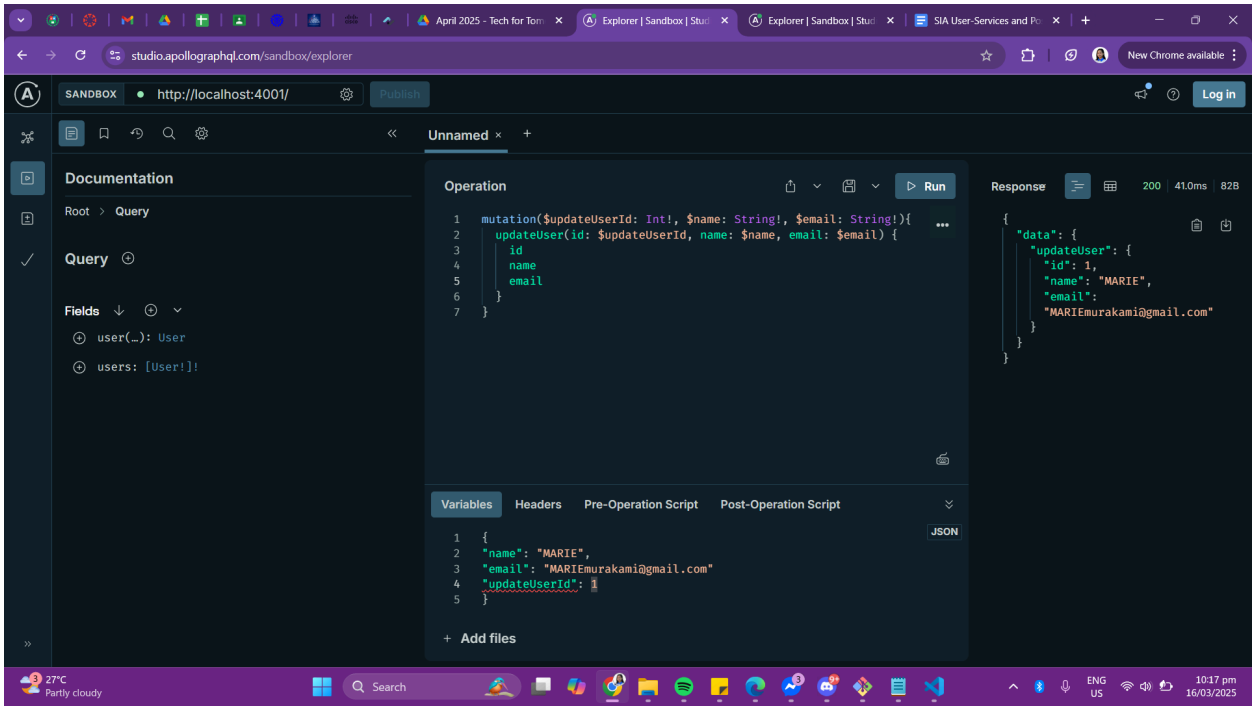
Create



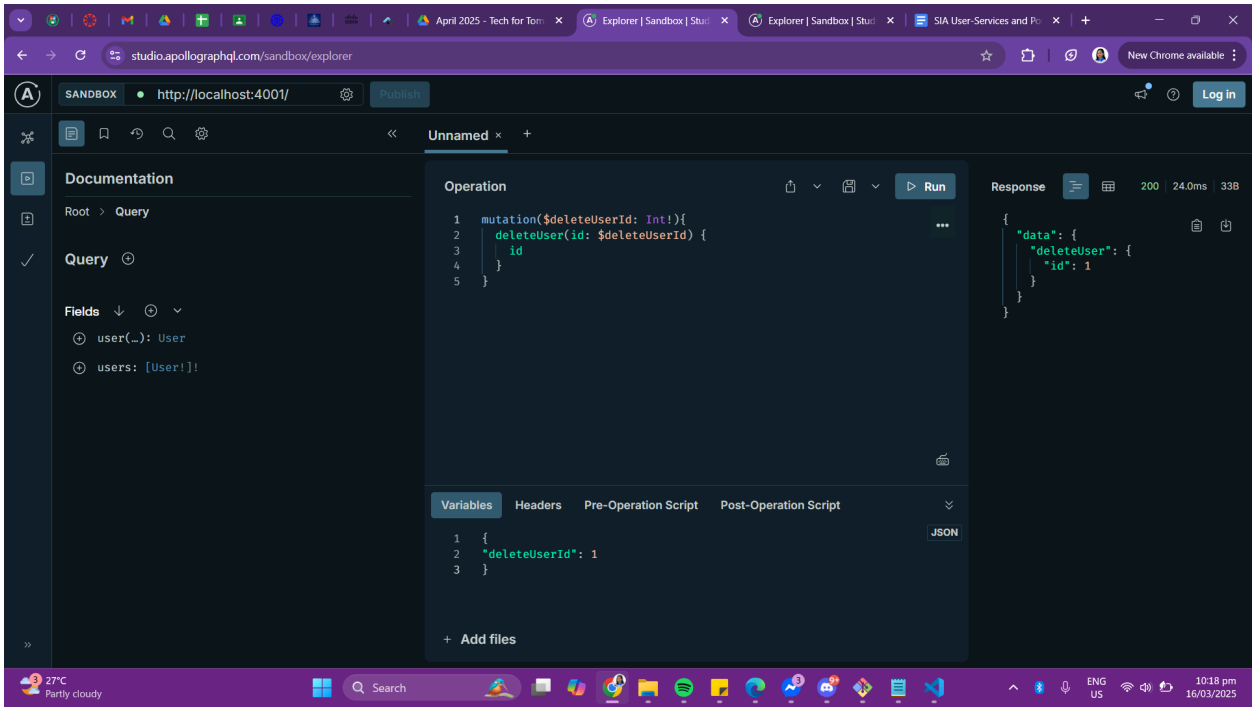
Read



Update


studio.apollographql.com/sandbox/explorer
SANDBOX
http://localhost:4001/
Publish
Log in
Unnamed
Documentation
Root > Query
Query
Fields
user(…): User
users: [User!]!
Operation
Run
mutation($updateUserId: Int!, $name: String!, $email: String!){
updateUser(id: $updateUserId, name: $name, email: $email) {
id
name
email
}
}
Response
200
41.0ms
82B
{
"data": {
"updateUser": {
"id": 1,
"name": "MARIE",
"email":
"MARIEmurakami@gmail.com"
}
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"name": "MARIE",
"email": "MARIEmurakami@gmail.com"
"updateUserId": 1
}
Add files
27°C
Partly cloudy
Search
ENG
US
10:17 pm
16/03/2025

Delete


studio.apollographql.com/sandbox/explorer
SANDBOX
http://localhost:4001/
Publish
Log in
Unnamed
Documentation
Root > Query
Query
Fields
user(…): User
users: [User!]!
Operation
Run
mutation($deleteUserId: Int!){
deleteUser(id: $deleteUserId) {
id
}
}
Response
200
24.0ms
33B
{
"data": {
"deleteUser": {
"id": 1
}
}
}
Variables
Headers
Pre-Operation Script
Post-Operation Script
JSON
{
"deleteUserId": 1
}
Add files
27°C
Partly cloudy
Search
ENG
US
10:18 pm
16/03/2025

Reflection

- What do database migrations do and why are they useful?
  - Database migrations manage changes to a database schema over time, such as adding or modifying tables and columns. They are useful for ensuring consistency between the database structure and application code, enabling seamless updates and collaboration across development environments.

- How does GraphQL differ from REST for CRUD operations?
  - GraphQL allows clients to request exactly the data they need in a single query, reducing over-fetching or under-fetching of data, whereas REST typically requires multiple endpoints for different resources, leading to fixed responses. Additionally, GraphQL uses a single endpoint for all operations (queries, mutations, and subscriptions), while REST relies on separate HTTP methods (GET, POST, PUT, DELETE) for CRUD operations.