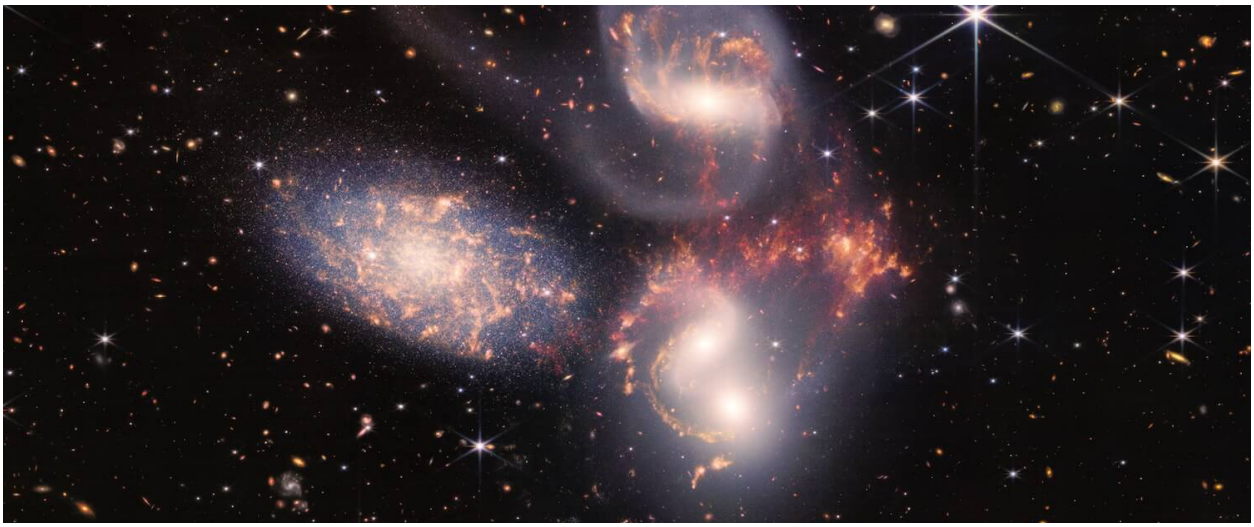

GraphQL API Documentation

Overview

This document provides a detailed description of the GraphQL API endpoints, including authentication, post creation, retrieval, updating, and deletion. Each section includes the endpoint, request structure, and expected responses.

---

1. Login

Description

Authenticates a user using email and password, returning a JWT token for further API requests.

Endpoint:

```
POST http://localhost:4000/graphql/login
```

Mutation:

```
mutation {
  login(email: "john@example.com", password: "securePassword1
```

```
23")
}
```

Response:

```
{
    "data": {
        "login": "<JWT_TOKEN>"
    }
}
```

---

2. Register

Description

Creates a new user account with a unique username and email.

Endpoint:

```
POST http://localhost:4000/graphql/register
```

Mutation:

```
mutation {
  register(username: "john_doe2", email: "john1@example.com",
password: "securePassword123") {
    id
    username
    email
  }
}
```

Response:

```
{
    "data": {
```

```
        "register": {
            "id": "67a0c2fabd4da284b0ca91ed",
            "username": "john_doe2",
            "email": "john1@example.com"
        }
    }
}
```

---

3. Create Post

Description

Creates a new post with a title and content.

Endpoint:

```
POST http://localhost:4000/graphql
```

Mutation:

```
mutation {
  createPost(title: "My 100th Post", content: "This is the co
ntent of the post.") {
    id
    title
    content
  }
}
```

Response:

```
{
    "data": {
        "createPost": {
            "id": "67a0c34bbd4da284b0ca91ef",
            "title": "My 100th Post",
```

```
            "content": "This is the content of the post."
        }
    }
}
```

---

4. Get All Posts

Description

Retrieves a list of all posts with their respective authors.

- **Performance Optimization:** Utilizes Redis caching for frequently accessed resources. This reduces the load on the primary database and improves the API response time for retrieving posts.

Endpoint:

```
GET http://localhost:4000/graphql
```

Query:

```
query {
  getPosts {
    id
    title
    content
    author {
        username
    }
  }
}
```

Response:

```
{
    "data": {
```

```
        "getPosts": [
            {
                "id": "67a05d72b6096617fb6c5138",
                "title": "My 10th Post",
                "content": "This is the content of the pos
t.",
                "author": {
                    "username": "john_doe"
                }
            },
            {
                "id": "67a079d428034dbdc2edc880",
                "title": "My 100th Post",
                "content": "This is the content of the pos
t.",
                "author": {
                    "username": "john_doe"
                }
            }
        ]
    }
}
```


5. Get Single Post

Description

Retrieves a single post by its unique ID.

Endpoint:

Retrieves a single post by its unique ID.

- **Performance Optimization:** Implements Redis caching for frequently accessed post data. This approach minimizes database hits and enhances the performance of fetching individual post details.

Endpoint:

```
GET http://localhost:4000/graphql
```

Query:

```
query {
  getPost(id: "67a079d428034dbdc2edc880") {
    id
    title
    content
    author {
        username
    }
  }
}
```

Response:

```
{
    "data": {
        "getPost": {
            "id": "67a079d428034dbdc2edc880",
            "title": "My 100th Post",
            "content": "This is the content of the post.",
            "author": {
                "username": "john_doe"
            }
        }
    }
}
```

---

6. Update Post

Description

Updates the title and content of an existing post.

Endpoint:

```
POST http://localhost:4000/graphql
```

Mutation:

```
mutation {
  updatePost(id: "67a059e5bc42b60c9b405f90", title: "Updated
Title", content: "Updated content") {
    id
    title
    content
  }
}
```

Response:

```
{
    "data": {
        "updatePost": {
            "id": "67a079d428034dbdc2edc880",
            "title": "Updated Title",
            "content": "Updated content"
        }
    }
}
```

---

7. Delete Post

Description

Deletes a post by its unique ID.

Endpoint:

```
POST http://localhost:4000/graphql
```

Mutation:

```
mutation {
  deletePost(id: "67a05b623ef36ac595a70a47")
}
```

Response:

```
null
```

Conclusion

This API documentation provides a structured reference for the authentication and post-management functionalities available in the system. Ensure that appropriate authentication is used when interacting with protected endpoints.