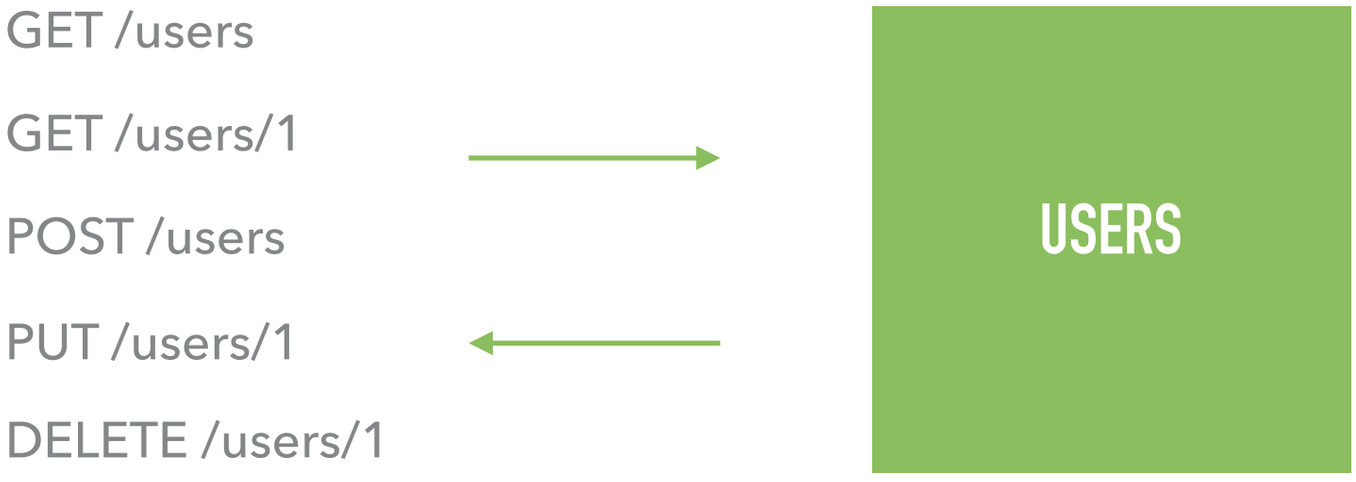
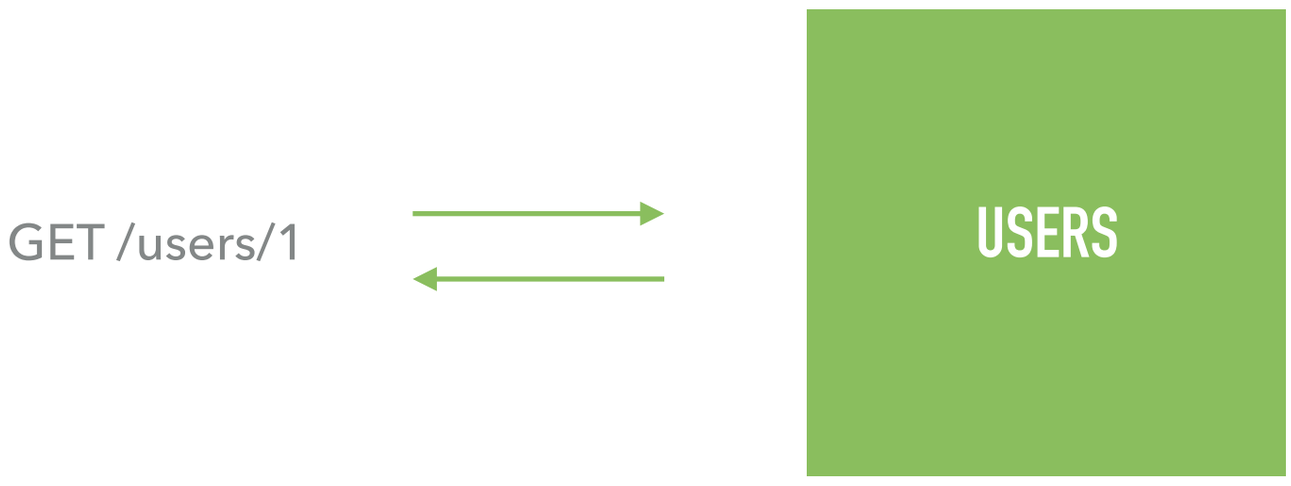
INTRODUCTION OF

GRAPHQL

RESTFUL API


GET /users
GET /users/1
POST /users
PUT /users/1
DELETE /users/1
USERS

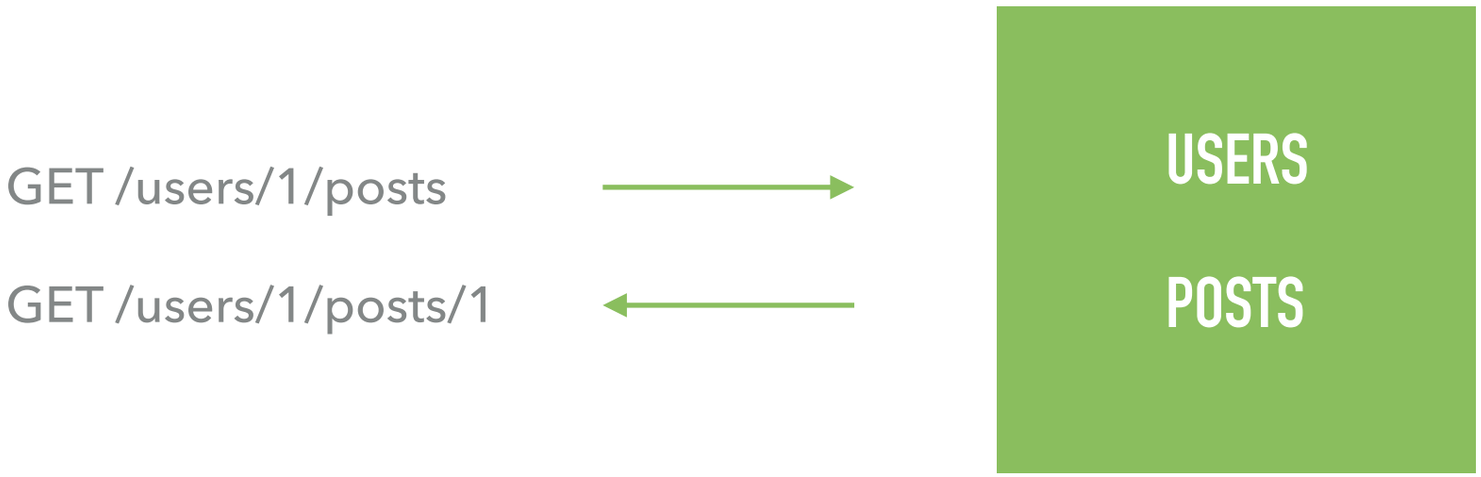
INDIVIDUAL USER



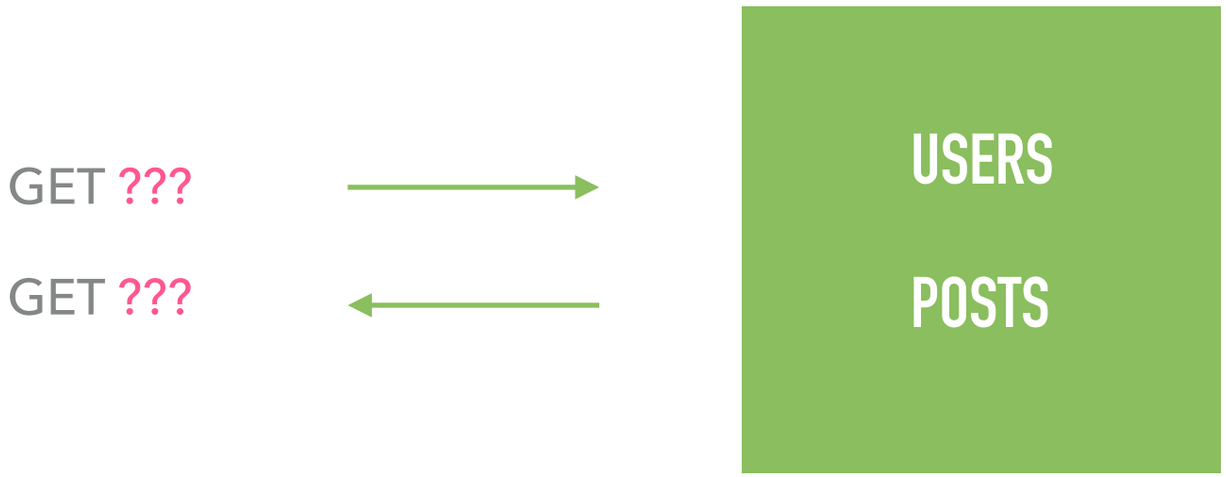
USER'S POSTS

GET ???

USER'S POSTS



USER'S POSTS AND FRIENDS



USER'S POSTS AND FRIENDS

GET /users/1/posts/friends →

GET /users/1/posts/1/friends ←

USER'S POSTS AND FRIENDS

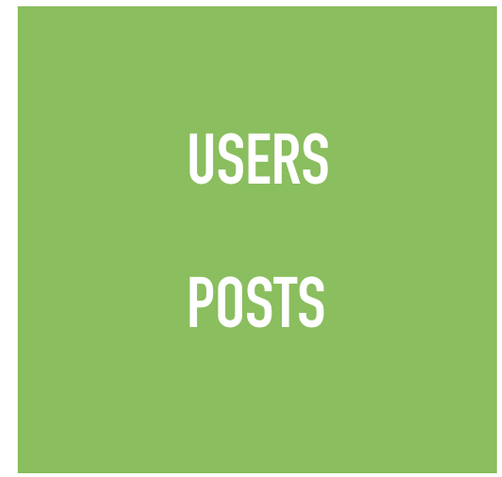
GET /users/1/postsandfriends →

GET /users/1/postsandfriends/1 ←

USER'S POSTS AND FRIENDS

GET /users/1/posts

GET /users/1/friends



GETTING STARTED

GRAPHQL

GRAPHQL

- GraphQL is a query language for APIs and a runtime for fulfilling those queries with your existing data

USER'S POSTS AND FRIENDS

Query

```
query {
  users {
    id
    name
    posts {
      text
    }
    friends {
      name
    }
  }
}
```

Result

```
{
  users: [
    {
      id: "1",
      name: "Aof",
      posts: [
        {
          text: "สวัสดี",
        },
      ],
      friends: [
        {
          name: "Ploy",
        },
      ],
    },
  ]
}
```

GRAPHQL

QUERIES AND MUTATIONS

QUERIES AND MUTATIONS

- **query** - retrieve data like method GET in RESTful API
- **mutation** - mutate data like method POST/PUT/DELETE in RESTful API

QUERIES AND MUTATIONS EXAMPLE

query

```
query {
    users {
        name
    }
}
```

mutation

```
mutation {
    addUser(name: String) {
        id
    }
}
```

GRAPHQL

TYPE SYSTEM

TYPE

- Every GraphQL service defines a set of types which completely describe the set of possible data you can query on that service

TYPE EXAMPLE

```
type Query {
    user: User
    users: [User]
}
```

```
type User {
    id: String!
    name: String
    age: Int
}
```

TYPE EXAMPLE

```
type Query {
    user: User
    users: [User]
}

type Mutation {
    addUser: User
}

type User {
    id: String!
    name: String
}
```

GRAPHQL

EXECUTION

EXECUTION

- You can think of each field in a GraphQL query as a function or method of the previous type which returns the next type

RESOLVER EXAMPLE

```
const resolvers = {
    Query: {
        user: () => {
            return “Aof”
        }
    }
}
```

GRAPHQL

GRAPHIQL

GRAPHIQL

GraphiQL

```
query TodoAppQuery($n: Int) {
  globalTodoList {
    items(first:$n) {
      edges {
        node {
          text
          complete
        }
      }
    }
  }
}
```

QUERY VARIABLES

```
{
  "n": 2
}
```

```
{
  "data": {
    "globalTodoList": {
      "items": {
        "edges": [
          {
            "node": {
              "text": "Release GraphiQL",
              "complete": true
            }
          },
          {
            "node": {
              "text": "Attend @Scale 2015",
              "complete": false
            }
          }
        ]
      }
    }
  }
}
```

STARTER PROJECT

- https://github.com/aofleejay/graphql-workshop