GraphQL

Goodbye Rest!!

What is GraphQL

A query language for your API. GraphQL is a query language for APIs and a runtime for fulfilling those queries with your existing data. GraphQL provides a complete and understandable description of the data in your API, gives clients the power to ask for exactly what they need and nothing more, makes it easier to evolve APIs over time, and enables powerful developer tools.

Standard way of fetching data in REST

① HTTP GET

```
{
  "user": {
    "id": "er3tg439frjw"
    "name": "Mary".
    "address": { … },.
    "birthday": "July 26, 1982"
  }
}
```

/users/<id>
/users/<id>/posts
/users/<id>/followers

② HTTP GET

```
{
  "posts": [{
    "id": "ncwon3ce89hs".
    "title": "Learn GraphQL today",
    "content": "Lorem ipsum … ".
    "comments": [ … ].
  }]
}
```

/users/<id>
/users/<id>/posts
/users/<id>/followers

③

```
{
  "followers": [{
    "id": "leo83h2dojsu"
    "name": "John".
    "address": { … }.
    "birthday": "July 26, 1982"
  },
  …]
}
```

HTTP GET

/users/<id>
/users/<id>/posts
/users/<id>/followers

With REST, you have to make three requests to different endpoints to fetch the required data.
You're also *overfetching* since the endpoints return additional information that's not needed.

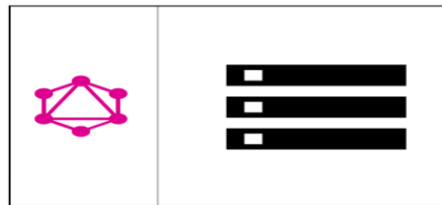# Fetching data in GraphQL

```
query {
    User(id: "er3tg439frjw") {
        name
        posts {
            title
        }
        followers(last: 3) {
            name
        }
    }
}
```

HTTP POST

```
{
    "data": {
        "User": {
            "name": "Mary",
            "posts": [
                { title: "Learn GraphQL today" }
            ],
            "followers": [
                { name: "John" },
                { name: "Alice" },
                { name: "Sarah" },
            ]
        }
    }
}
```

Using GraphQL, the client can specify exactly the data it needs in a *query*. Notice that the *structure* of the server's response follows precisely the nested structure defined in the query.

Queries and Mutation

- Queries perform the task of GET ALL , GET by ID functions that we are familiar in REST architecture.
- Whenever you want to change the application resources use Mutation.
- Mutation performs all DELETE, PUT and POST operations which we use in REST.

Advantages of using GraphQL

- Solves the problem of over-fetching and under-fetching.
- Rapid Product Iterations on the Frontend
- Insightful Analytics on the Backend
- Benefits of a Schema & Type System

Companies using GraphQL