

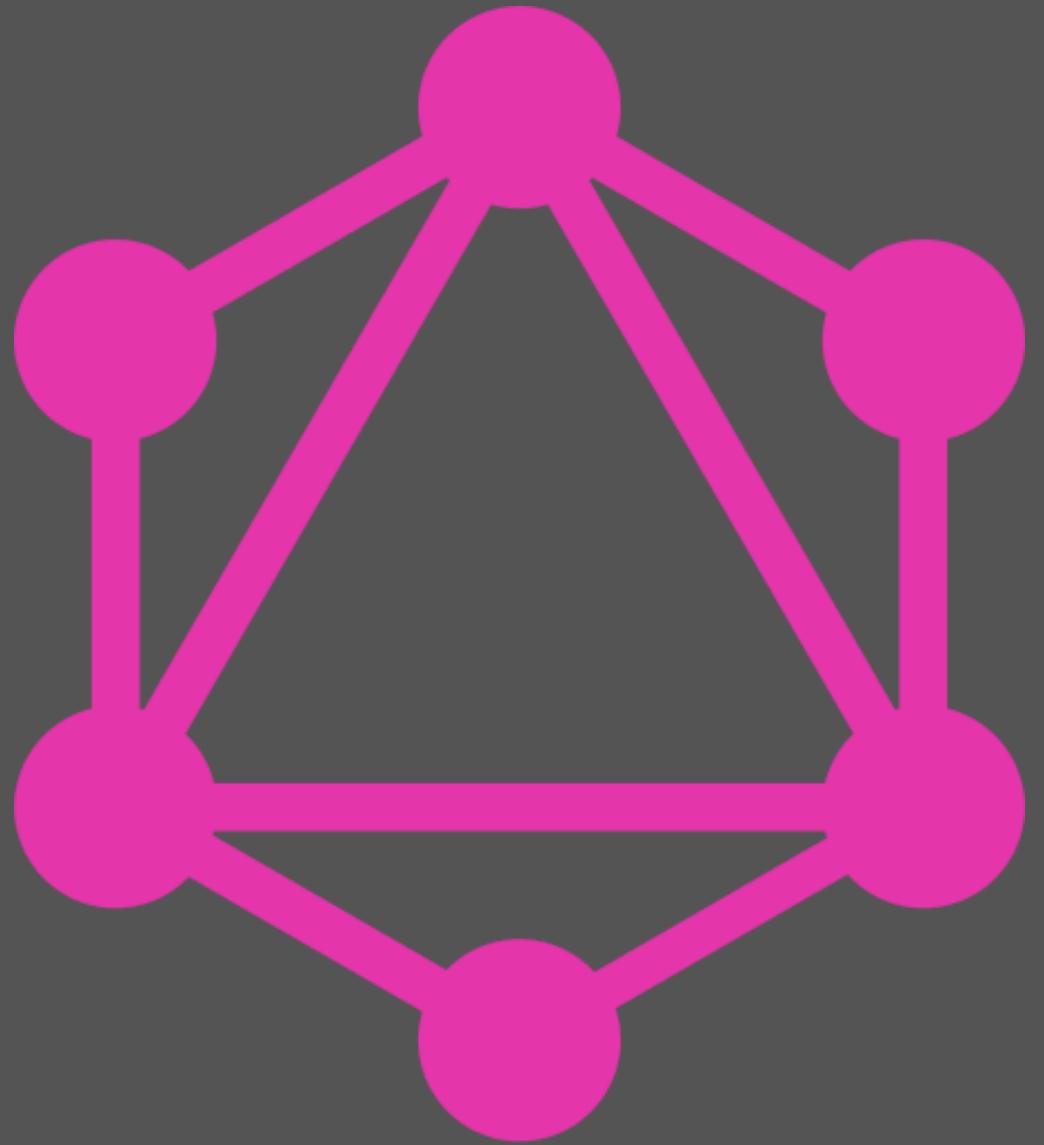
APLICACIONS I SERVEIS WEB

GraphQL

Jorgina Arrés
Gisela Ruzafa
Cristian Ferrer
Quim Lázaro



QUÈ ÉS GRAPHQL?



GraphQL és un llenguatge per a gestionar querys a la nostre API utilitzant un sistema de tipus que definim sobre les nostres dades. GraphQL no està subjecte a cap tipus específic de base de dades i té suport per a diversos llenguatges de programació. Pretén ser una alternativa a la ja coneguda API REST, tot i que hi ha gent que ho considera un tipus concret de REST.

FITES IMPORTANTS

2012

Facebook inicia el
desenvolupament
de GraphQL

2015

Llençament a
Producció

2016

Llençament de
la versió estable
actual

2018

GraphQL afegeix
Schema Definition
Language (SDL) a
l'especificació

CARACTERÍSTIQUES DE GRAPHQL

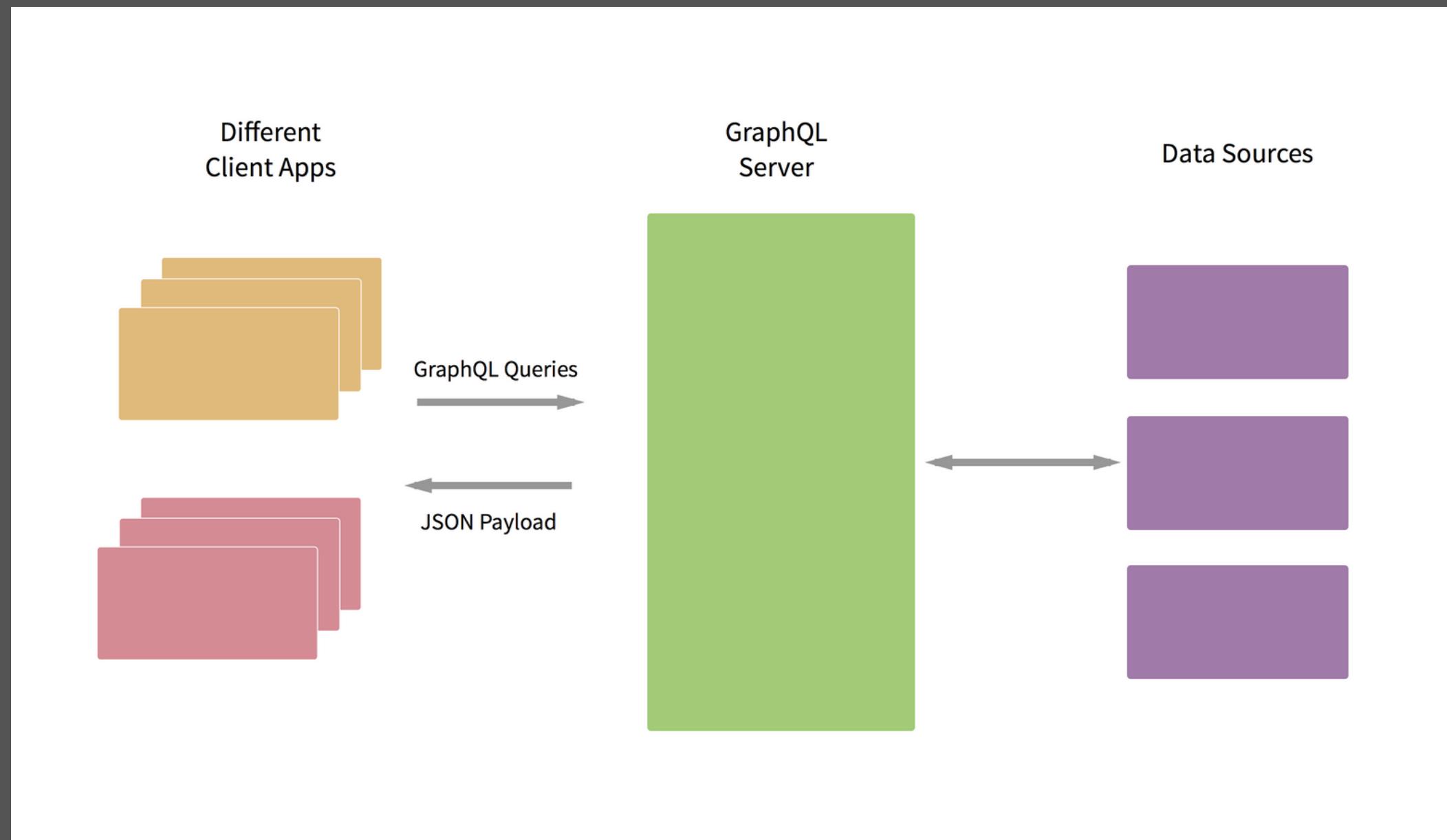
- **Query**

```
{  
  me {  
    name  
  }  
}  
  
{  
  "me": {  
    "name" : "Luke Skywalker"  
  }  
}
```

- **Response**

**Retorna només els camps
especificats en cada consulta,
reduint el BW i el temps per servir els
recursos demanats**

CARACTERÍSTIQUES DE GRAPHQL



**Es pot accedir a recursos
diferents amb una sola consulta**

CARACTERÍSTIQUES DE GRAPHQL

```
type Pizza {  
    id: Int!  
    name: String!  
    origin: String  
    ingredients: [Ingredient]  
}
```

S'eliminen els endpoints, substituint-los per a tipus, que indiquen només els atributs que es necessiten

CARACTERÍSTIQUES DE GRAPHQL

```
type Pizza {  
    id: Int!  
    name: String!  
    origin: String @deprecated  
    ingredients: [Ingredient]  
}
```

S'intenta eliminar el versionat de les APIs, donant la opció de deprecar els atributs

CARACTERÍSTIQUES DE GRAPHQL



Postgre**SQL**



mongoDB

Desacoblala dependència
d'accès a la BD

CARACTERÍSTIQUES DE GRAPHQL



python™

Suport per a diferents
llenguatges



QUERY

Podem consultar les dades dels Schemas mitjançant querys:

```
query {  
    getAccount("id": 1){  
        id  
        avatar  
        name  
        keys: {  
            keyValidation,  
            keySession  
        }  
    }  
}
```

```
{  
    "data": {  
        "id": 1,  
        "avatar": "https://solucionessore.com/src/images/avatart/joan\_avt.png",  
        "name": "Joan Andrés Lara Mora",  
        "keys": {  
            ...  
        }  
    }  
}
```

MUTATION

Per afegir, actualitzar o eliminar dades utilitzarem les mutations:

```
mutation {
  changeName("id": 1, "newName": "Mi nuevo nombre") {
    newName
    validate
  }
}
```

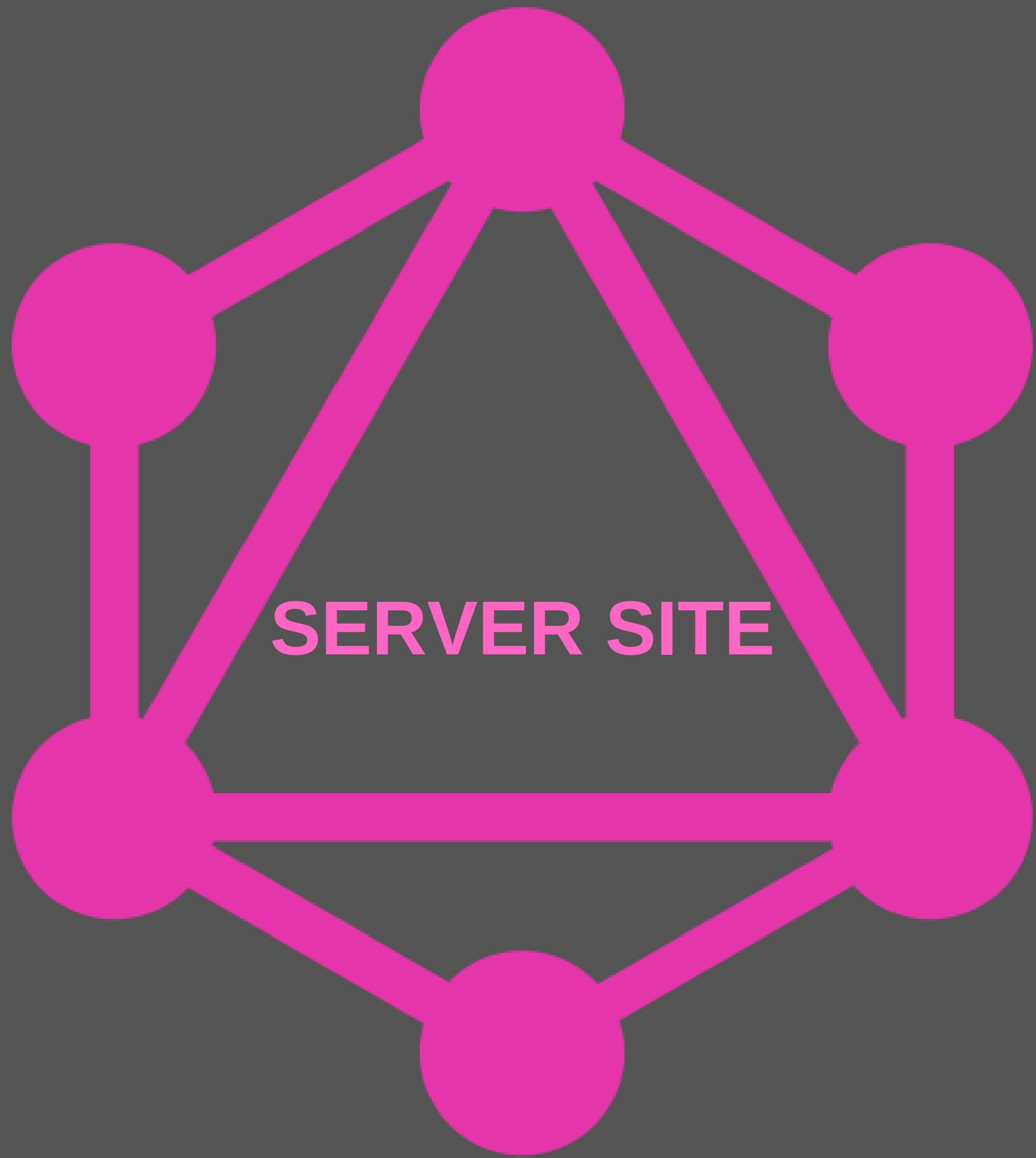
```
{
  "data": {
    "changeName": {
      "newName": "Mi nuevo nombre",
      "validate": false
    }
  }
}
```

SUBSCRIPTION

Per obtenir informació a temps real dels canvis a les dades podem utilitzar les subscriptions:

```
subscription {
  notificationOf("type": "like") {
    id
    type
    data {
      photo {
        id
        name
      }
      user {
        id
        avatar
        name
      }
      date
    }
  }
}
```

```
{
  "data": {
    "id": 45,
    "type": "like",
    "data": {
      "photo": {
        "id": 76376482,
        "name": "photos_news"
      },
      "user": {
        "id": 767,
        "name": "Fernando Rívera Alarcón"
      },
      "date": "2017-12-05T16:40:46"
    }
  }
}
```



DEFINICIÓ DE L'SCHEMA

- **Schema**

```
schema {  
    query: Query  
    mutation: Mutation  
}
```
- **Query**

```
type Query {  
    pizzas (name: [String]): [Pizza]  
    ingredients: [Ingredient]  
}
```
- **Mutation**

```
type Mutation {  
    createPizzas (pizzas: [PizzaInput]): [Pizza]  
}
```

DEFINICIÓ DE L'SCHEMA

- **Types**

```
type Pizza {  
    id: Int!  
    name: String!  
    origin: String  
    ingredients: [Ingredient]  
}
```

```
type Ingredient {  
    id: Int!  
    name: String!  
    calories: String  
}
```

```
input PizzaInput {  
    name: String!  
    origin: String  
    ingredientIds: [Int]  
}
```

RESOLVERS

```
const pizzaResolver = {
  Query: {
    pizzas(root, { name }) {
      // Business logic to get Pizza Object
      return requestedPizzas;
    }
  },
}

Pizza: {
  ingredients(pizza) {
    // Business logic to resolve the field
    "ingredients" when "Pizza" object is requested
    return requestedIngredients;
  }
},
}

Mutation: {
  createPizzas(root, { pizzas }) {
    // Business logic to add new pizza to database
    // The "ingredients" field is autoresolved by
    GraphQL
    return newPizzas;
  }
}
};
```

PROS I CONTRES DE GraphQL

- **Pros**
- **Obtenció de dades**

Amb una query podem obtenir totes les dades Consultes concretes
- **Versionat senzill**
- **Esquema fortament tipat**

Qualsevol mal us pot ser detectat ràpidament
- **Obtenció d'informació de colls d'ampolla**
- **Contres**
- **Gestió d'errors complexa**
- **Swagger no soporta GraphQL**
- **No hi ha control sobre les consultes de tercets**
- **No té mecanisme d'emmegatzament de caché**

PERSPECTIVES DE FUTUR



Facebook



Github



Pinterest



Intruit



Coursera



Shopify



Natura

CONCLUSIONS

Es centra en les necessitats de desenvolupament que tenim actualment per a crear APIs que s'adequin a qualsevol tipus d'ús, el seu manteniment és menys costós i ens permet agilitzar els temps de desenvolupament i canvis en la part client. Té una comunitat molt activa que tracta de millorar el producte i sobretot la experiència de desenvolupament.

WEBGRAFIA

- <https://graphql.org/learn/>
- <https://foundation.graphql.org/>
- <https://graphql.org/blog/graphql-a-query-language/>
- <https://en.wikipedia.org/wiki/GraphQL>
- <https://jacobwgillespie.com/from-rest-to-graphql-b4e95e94c26b#.tag7nzkrb>
- <https://www.genbeta.com/desarrollo/por-que-deberiamos-abandonar-rest-y-empezar-a-usar-graphql-en-nuestras-apis>
- <https://www.moesif.com/blog/technical/graphql/REST-vs-GraphQL-APIs-the-good-the-bad-the-ugly/#>
- <https://sensedia.com/es/blog/apis/la-ascencion-de-graphql-en-la-era-de-las-apis/>

REPARTIMENT DE LES TASQUES

	CONTINGUT	TRANSPARÈNCIES	PRESENTACIÓ
Cristian	X	X	
Jorgina	X	X	
Gisela	X	X	X
Quim	X	X	X