LEVANDO O MUNDO GRAPHQL PARA O DJANGO COM O GRAPHENE

DANIEL CÂNDIDO
dcs2@cesar.org.br

MARLON CHALEGRE
mcp@cesar.org.br

C.E.S.A.R

Vai

QUEM SOMOS

DANIEL CÂNDIDO

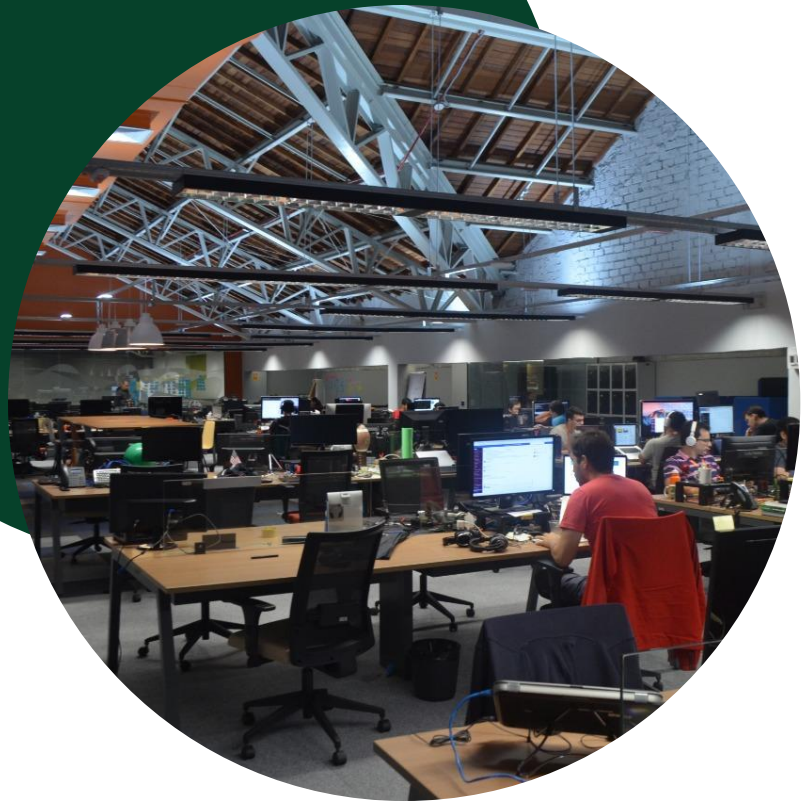
ENGENHEIRO DE SOFTWARE

MARLON CHALEGRE

ENGENHEIRO DE SOFTWARE

C.E.S.A.R

Vai



C . e . S . A . R

Centro de Inovação que utiliza Design e Tecnologia da Comunicação e Informação para resolver problemas complexos de um mercado muito diverso.

Em torno de 650 colaboradores espalhados em alguns estados do país.

MAPA DA APRESENTAÇÃO

- INTRODUÇÃO AO GRAPHQL

MAPA DA APRESENTAÇÃO

INTRODUÇÃO AO GRAPHQL

OVERVIEW DO GRAPHENE

C.E.S.A.R

Vai

MAPA DA APRESENTAÇÃO

- INTRODUÇÃO AO GRAPHQL
- OVERVIEW DO GRAPHENE
- SETUP DE SERVIDOR GRAPHQL COM DJANGO E GRAPHENE

C.E.S.A.R

Vai

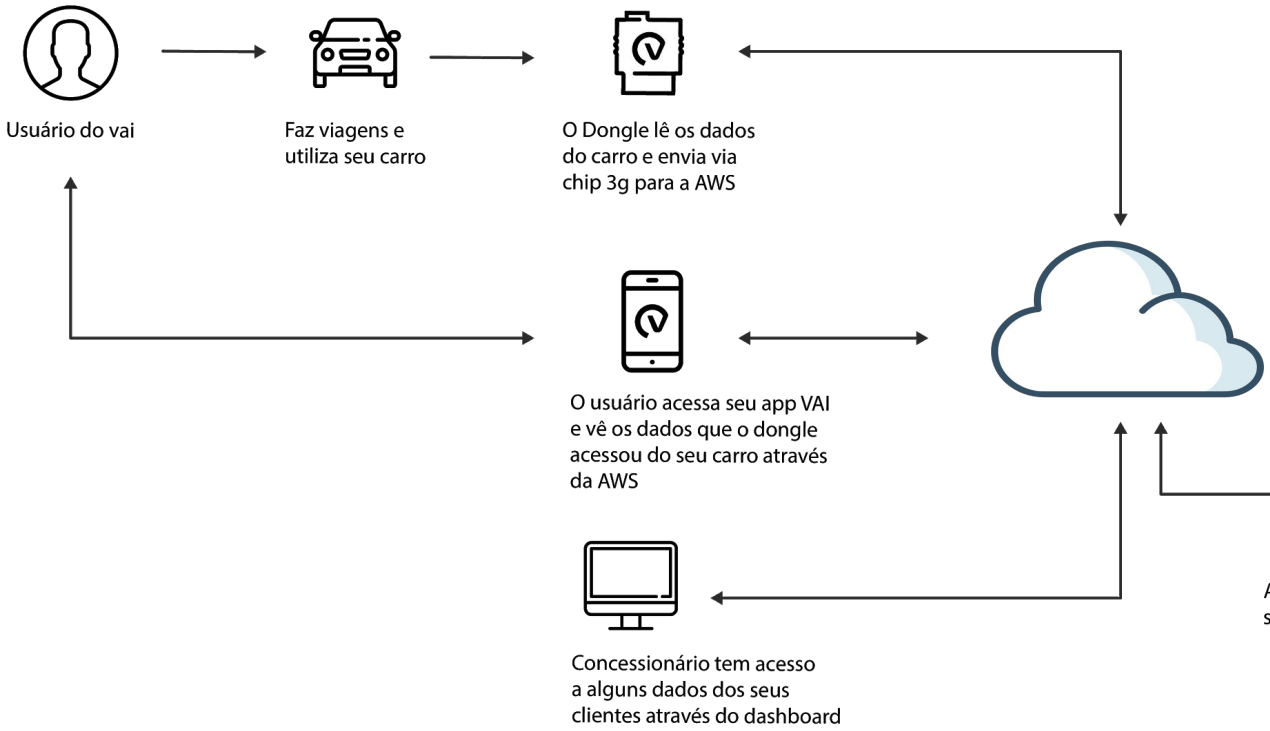
MAPA DA APRESENTAÇÃO

- INTRODUÇÃO AO GRAPHQL
- OVERVIEW DO GRAPHENE
- SETUP DE SERVIDOR GRAPHQL COM DJANGO E GRAPHENE
- DÚVIDAS, DORES, AFLIÇÕES, FEEDBACKS (?)

C.E.S.A.R

Vai

O PROJETO


Usuário do vai
Faz viagens e utiliza seu carro
O Dongle lê os dados do carro e envia via chip 3g para a AWS
O usuário acessa seu app VAI e vê os dados que o dongle acessou do seu carro através da AWS
Concessionário tem acesso a alguns dados dos seus clientes através do dashboard
Alguns serviços utilizam software de terceiros

O TIME

DESIGNER

FULL STACK

DEV BACKEND

DEV FRONTEND

DEV/QA

O TIME

DESIGNER

FULL STACK

DEV BACKEND

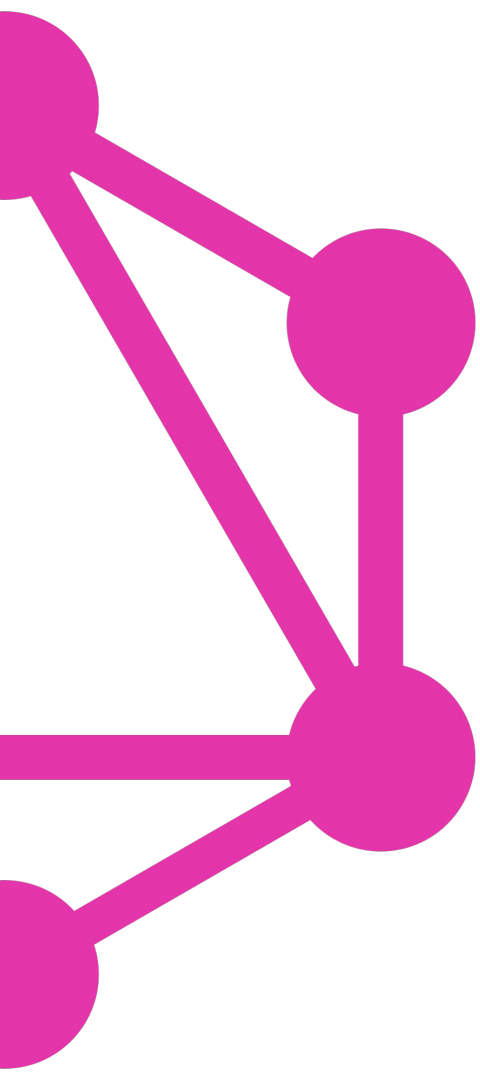
DEV FRONTEND

DEV/QA

GRAPHQL
PARA QUEM *(NUNCA)*
OUVIU FALAR DE
GRAPHQL

C.E.S.A.R

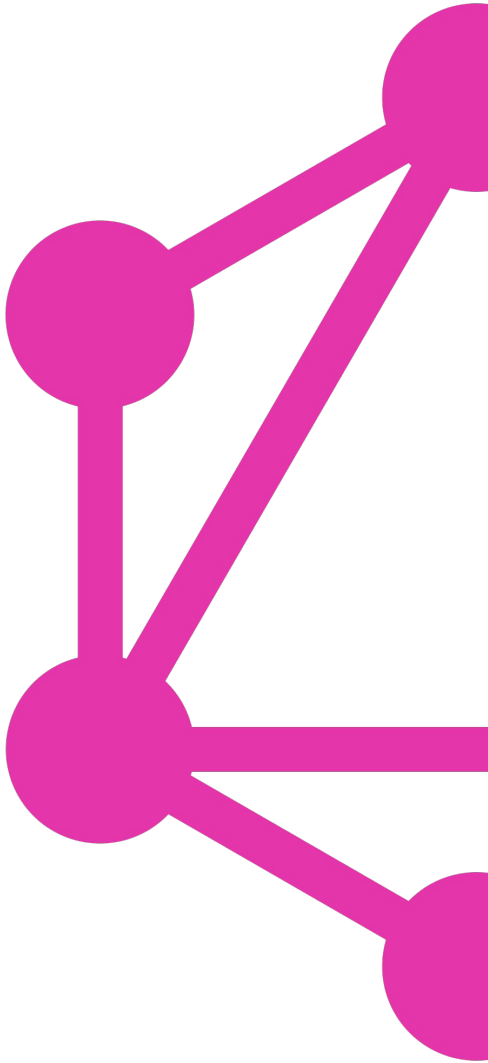
Vai



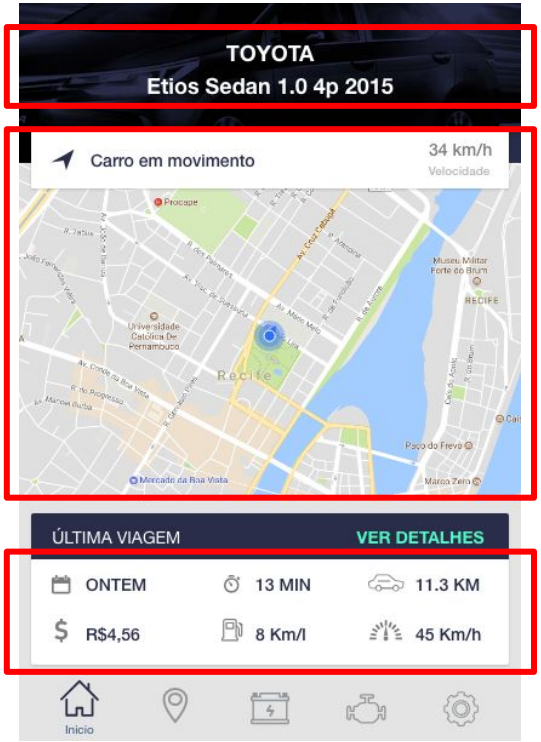
LINGUAGEM DE **CONSULTA**,
DESENVOLVIDA PELO
FACEBOOK EM 2015

ALTERNATIVA AO **REST**
PARA DESENVOLVIMENTO
DE API'S

O **FRONTEND** DECIDE
QUAIS **DADOS** QUER
CONSUMIR DA API



COM REST



GET -> /CARINFO/:CARID

```
{
    "montadora": "Toyota",
    "modelo": "Etios Sedan 1.0 4p",
    "ano": 2015
    ...
}
```

GET -> /LOCATION/:CARID

```
{
    "latitude": -8.0418006,
    "longitude": -35.0787237,
    "velocidade": 34
    ...
}
```

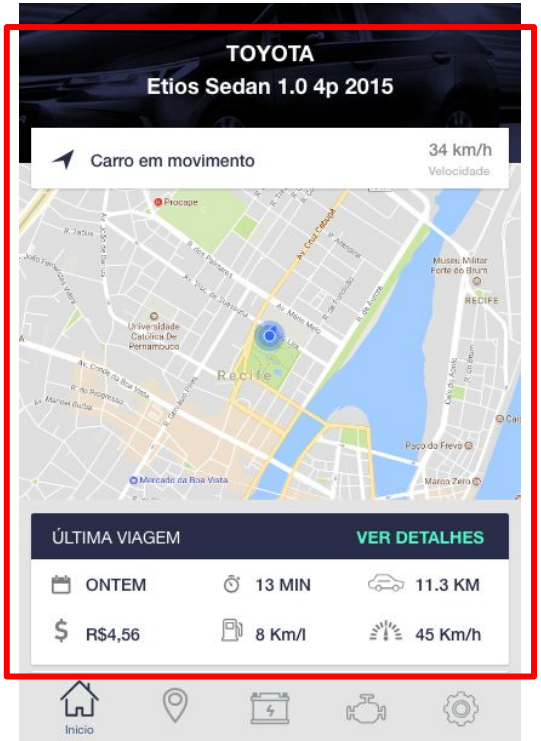
GET -> /TRIPS/:CARID/LAST

```
{
    "data": "2019-07-16 13:50",
    "duracao": 13,
    "distancia": 11300,
    "custo": 456,
    "consumo": 8.0,
    "velocidade": 45
    ...
}
```

C.E.S.A.R

Vai

COM GRAPHQL

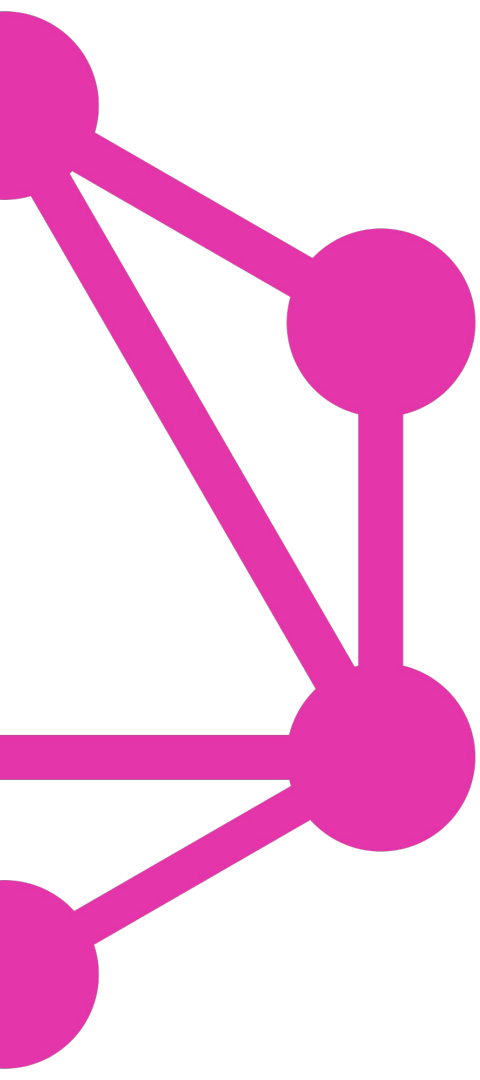


POST -> /GRAPHQL

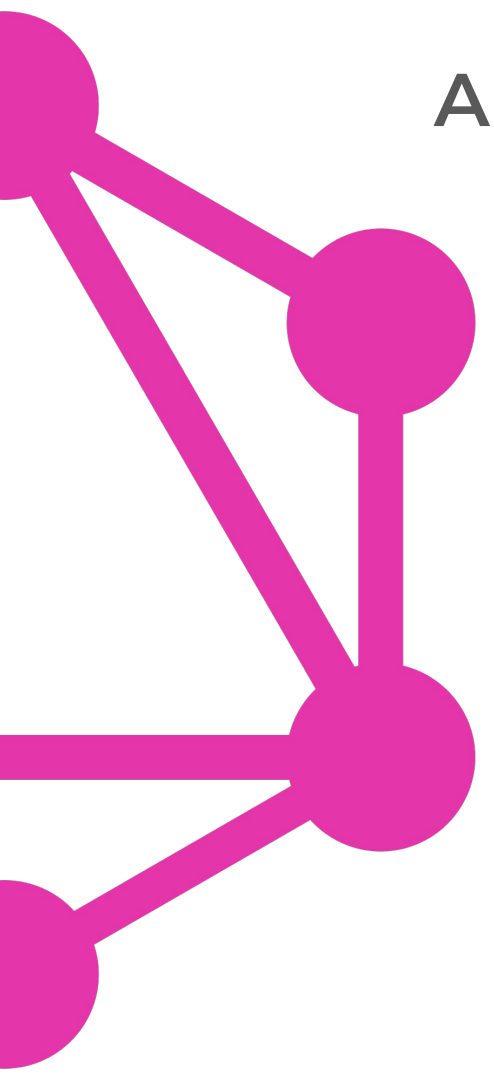
```
{
  car {
    montadora
    modelo
    ano
    location {
      latitude
      longitude
      velocidade
    }
    lastTrip {
      data
      duracao
      distancia
      custo
      consumo
      velocidade
    }
  }
}
```

```
{
  car {
    montadora: "Toyota"
    modelo: "Etios Sedan 1.0 4p"
    ano: 2015
    location {
      latitude: -8.0418006
      longitude:  -35.0787237
      velocidade: 34
    }
    lastTrip {
      data: "2019-07-16 13:50"
      duracao: 13
      distancia: 11300
      custo: 456
      consumo: 8.0
      velocidade: 45
    }
  }
}
```

C.E.S.A.R



O **BACKEND** NÃO PRECISA SE **PREOCUPAR** QUAIS DADOS SERÃO **CONSUMIDOS** POR CADA CLIENTE



APENAS **DISPONIBILIZA** OS DADOS ATRAVÉS DE UM **SCHEMA**

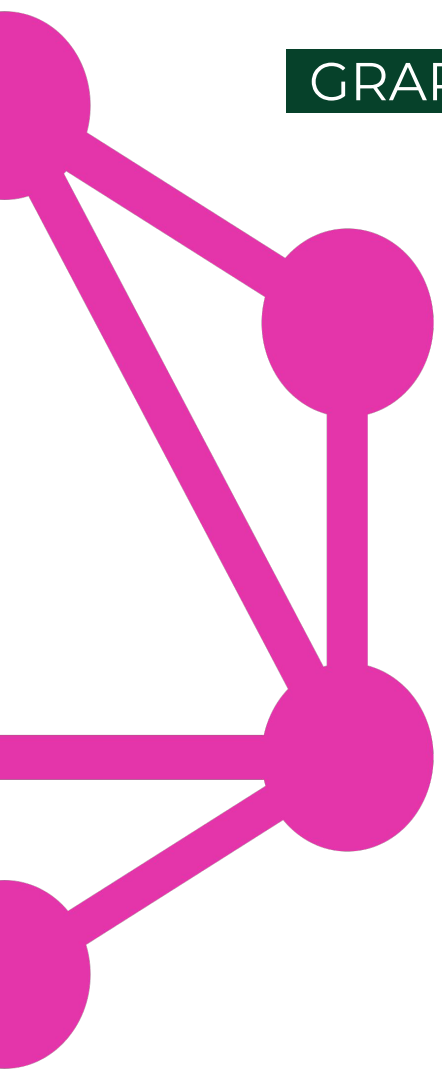
```
type Car {
  montadora: String
  modelo: String
  ano: Int
  location: Location
  trips: [Trip]
  lastTrip: Trip
}
```

```
type Location {
  latitude: Float
  longitude: Float
  velocidade: Int
}
```

```
type Trip {
  data: Date
  duracao: Int
  distancia: Int
  custo: Int
  consumo: Float
  velocidade: Int
}
```

OVERVIEW DO GRAPHENE

C.E.S.A.R



GRAPHENE

BIBLIOTECA MAIS POPULAR

O SCHEMA É DEFINIDO EM PYTHON

INTEGRAÇÃO COM DJANGO E OUTROS WEB FRAMEWORKS

GRAPHENE

```python
import graphene

class Patron(graphene.ObjectType):
    id = graphene.ID()
    name = graphene.String()
    age = graphene.Int()

class Query(graphene.ObjectType):
    patron = graphene.Field(Patron)
    def resolve_patron(self, info):
        return Patron(id=1, name="Syrus", age=27)

schema = graphene.Schema(query=Query)
```

```python
query = """
    query something{
      patron {
        id
        name
        age
      }
    }
"""
def test_query():
    result = schema.execute(query)
    assert not result.errors
    assert result.data == {"patron": {"id": "1",
"name": "Syrus", "age": 27}}

if __name__ == "__main__":
    result = schema.execute(query)
    print(result.data["patron"])
```

INTEGRANDO GRAPHENE COM O DJANGO

C.E.S.A.R

Vai

PYTHON + DJANGO + GRAPHENE + GRAPHENE_DJANGO

Graphene
Python

TALK IS CHEAP.
DEMONSTRAÇÃO

SETUP

- GRAPHENE + GRAPHENE_DJANGO

C.E.S.A.R

Vai

SETUP

GRAPHENE + GRAPHENE_DJANGO

SETTINGS DO DJANGO

C.E.S.A.R

Vai

SETUP

- GRAPHENE + GRAPHENE_DJANGO
- SETTINGS DO DJANGO
- CRIAR O APP + DEFINIR AS URLS

C.E.S.A.R

Vai

SETUP

- GRAPHENE + GRAPHENE_DJANGO
- SETTINGS DO DJANGO
- CRIAR O APP + DEFINIR AS URLS
- DECLARAR SCHEMA

C.E.S.A.R

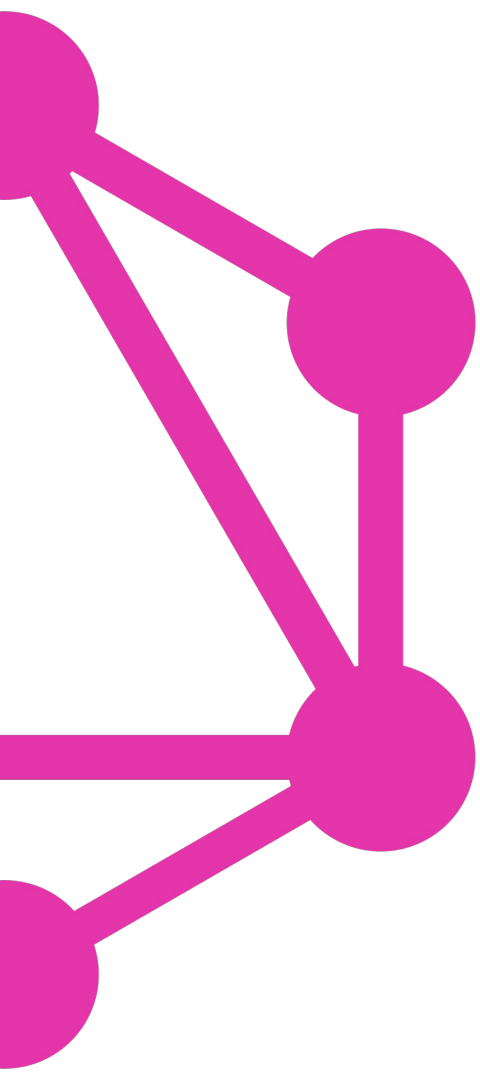
Vai

SETUP

- GRAPHENE + GRAPHENE_DJANGO
- SETTINGS DO DJANGO
- CRIAR O APP + DEFINIR AS URLS
- DECLARAR SCHEMA
- CRIAR QUERIES E MUTATIONS

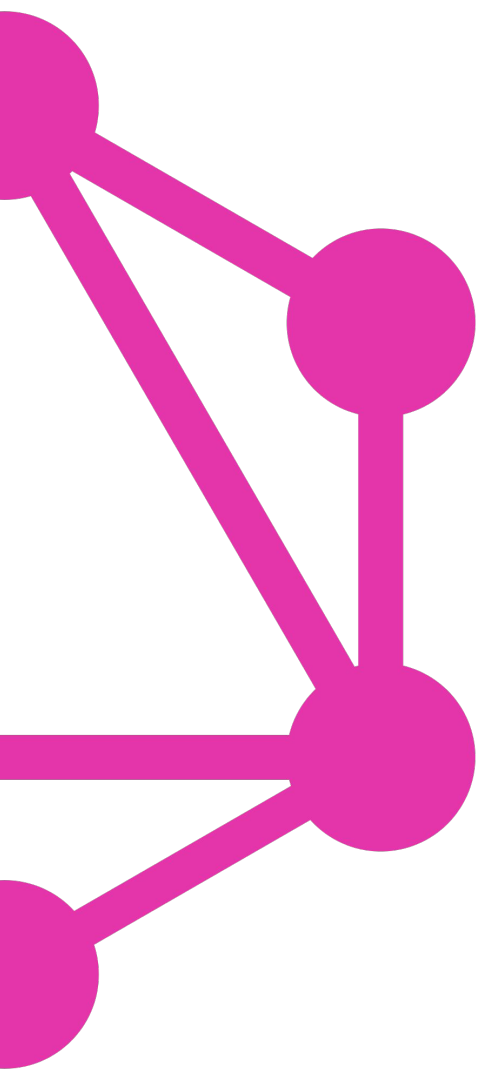
C.E.S.A.R

Vai



COM GRAPHQL NEM TUDO SÃO FLORES

O **BACKEND** PRECISA SE **PREOCUPAR COMO** OS DADOS SERÃO **BUSCADOS** INTERNAMENTE



```
query maliciousQuery {
  thread(id: "some-id") {
    messages(first: 99999) {
      thread {
        messages(first: 99999) {
          thread {
            messages(first: 99999) {
              thread {
                # ...repeat times 10000...
              }
            }
          }
        }
      }
    }
  }
}
```

DÚVIDAS, APLICABILIDADE DE PROJETO, EXPERIÊNCIAS?

VALEU!

TEMOS VAGAS!

cesar.org.br

CÓDIGO EM:

...

DANIEL CÂNDIDO
dcs2@cesar.org.br

MARLON CHALEGRE
mcp@cesar.org.br

vai.com.vc