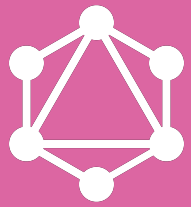
TESTANDO API'S



GRAPHQL

CARLOS LEAL
cels@cesar.org.br

DANIEL CÂNDIDO
dcs2@cesar.org.br

C.E.S.A.R

Vai

QUEM SOMOS

CARLOS LEAL

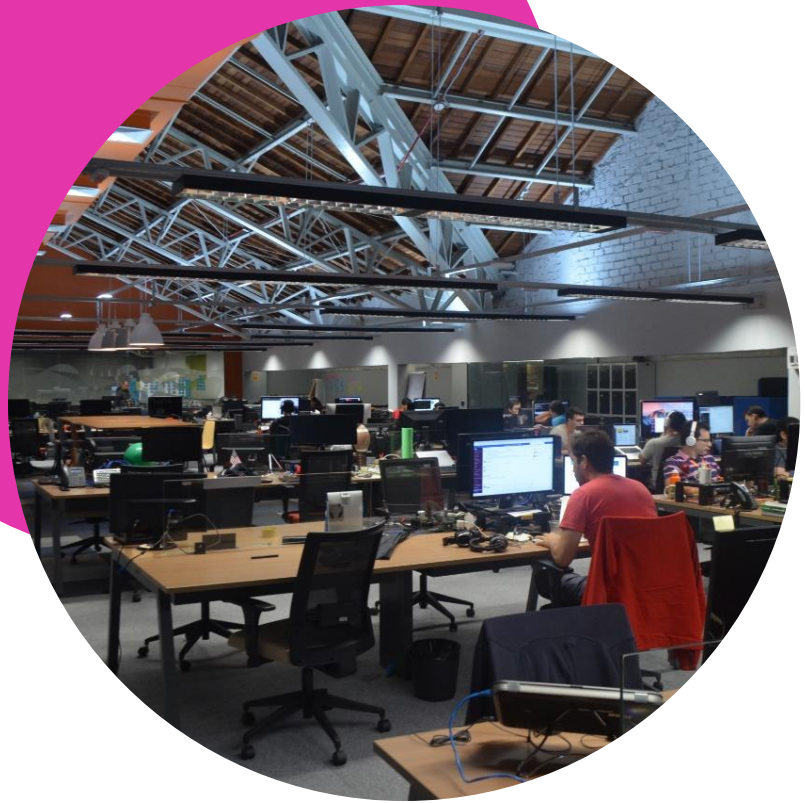
ENGENHEIRO DE TESTES

DANIEL CÂNDIDO

ENGENHEIRO DE SISTEMAS

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

- COMO FAZÍAMOS TESTES EM REST

MAPA DA APRESENTAÇÃO

- COMO FAZÍAMOS TESTES EM REST
- INTRODUÇÃO AO GRAPHQL

MAPA DA APRESENTAÇÃO

- COMO FAZÍAMOS TESTES EM REST
- INTRODUÇÃO AO GRAPHQL
- COMO FAZEMOS TESTES EM GRAPHQL

MAPA DA APRESENTAÇÃO

- COMO FAZÍAMOS TESTES EM REST
- INTRODUÇÃO AO GRAPHQL
- COMO FAZEMOS TESTES EM GRAPHQL
- BENEFÍCIOS GERAIS

MAPA DA APRESENTAÇÃO

- COMO FAZÍAMOS TESTES EM REST
- INTRODUÇÃO AO GRAPHQL
- COMO FAZEMOS TESTES EM GRAPHQL
- BENEFÍCIOS GERAIS
- DEMONSTRAÇÃO DO SETUP NECESSÁRIO

MAPA DA APRESENTAÇÃO

- COMO FAZÍAMOS TESTES EM REST
- INTRODUÇÃO AO GRAPHQL
- COMO FAZEMOS TESTES EM GRAPHQL
- BENEFÍCIOS GERAIS
- DEMONSTRAÇÃO DO SETUP NECESSÁRIO
- DÚVIDAS, DORES, AFLIÇÕES, FEEDBACKS (?)

O TIME

DESIGNER

FULL STACK

DEV BACKEND

DEV FRONTEND

DEV/QA

O TIME

DESIGNER

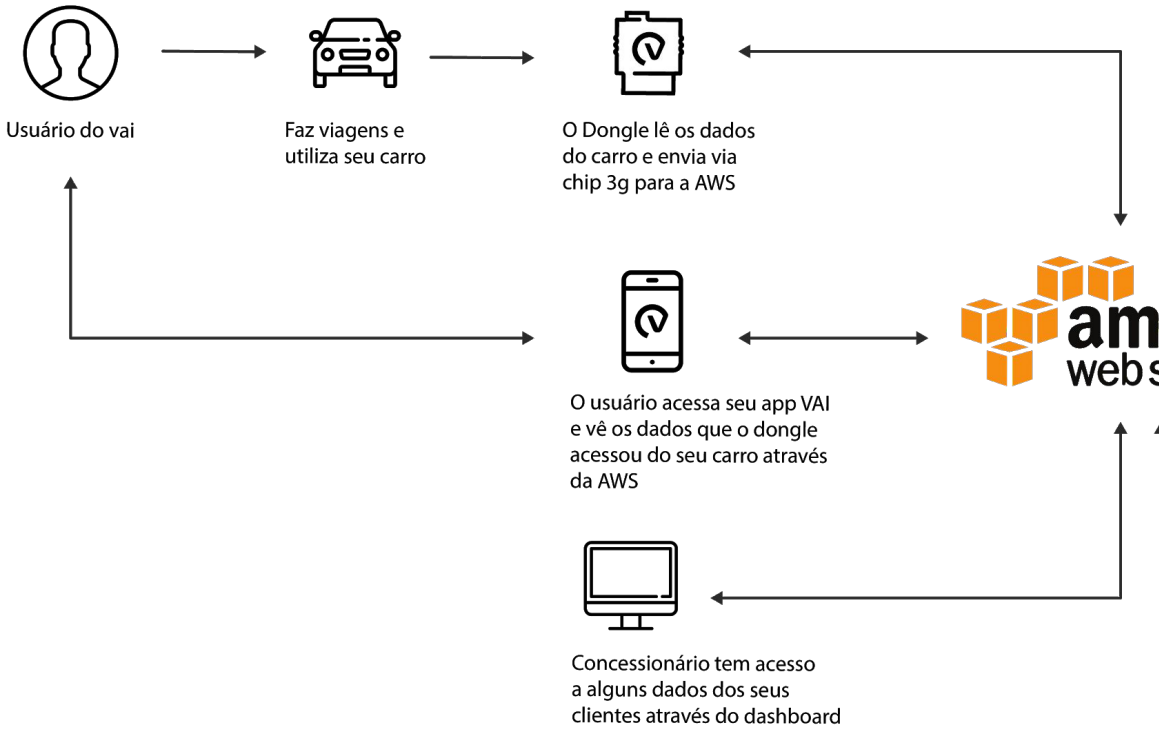
FULL STACK

DEV BACKEND

DEV FRONTEND

DEV/QA

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
amazon web services
O usuário acessa seu app VAI e vê os dados que o dongle acessou do seu carro através da AWS
Alguns serviços utilizam software de terceiros
Concessionário tem acesso a alguns dados dos seus clientes através do dashboard

TESTES EM SERVIÇO REST

ARQUITETURA DE ALTO NÍVEL

ARQUITETURA DE ALTO NÍVEL

FLUXO DE TESTE

SUBIR SERVIÇO HTTP

DEPENDENDO DO FRAMEWORK

ENVIAR REQUISIÇÕES

VALIDAR RESPONSES

DATABASE TEMPORÁRIO

VALIDAÇÕES

RESPONSE BODY

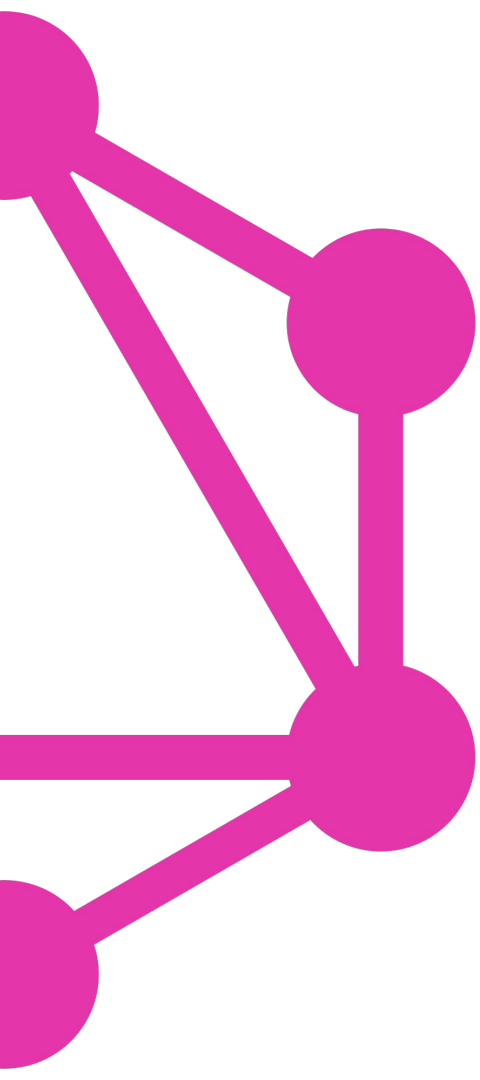
STATUS CODE

VALIDAÇÕES

```
it('Should list all Wings plans and prices', function(done) {
  chai
    .request(protheus_url)
    .get('/tabelapreco')
    .auth(username, password)
    .end((err, res) => {
      const response = res.body;

      expect(res.status).to.be.equal(200);
      expect(response).to.be.deep.equal(
        protheus_resource.expected_table_prices
      );
      done();
    });
});
```

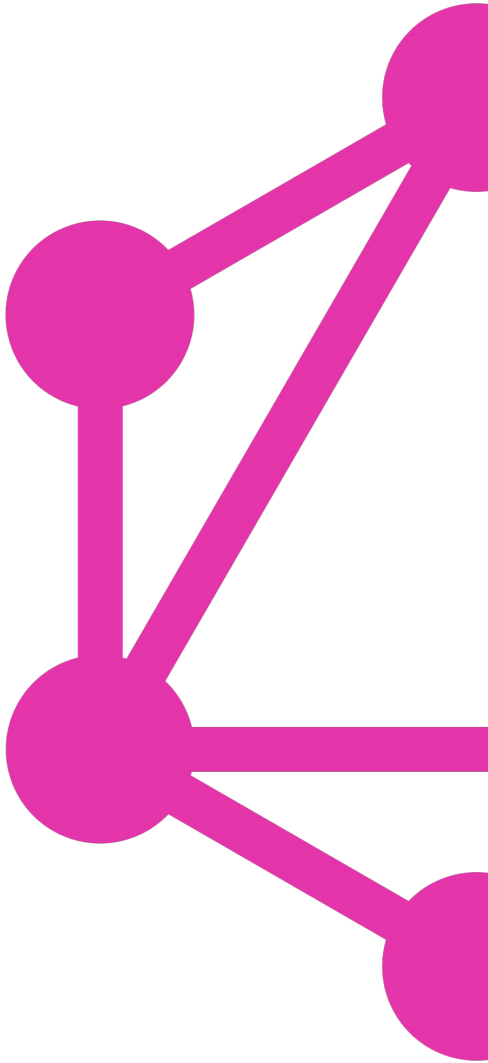
GRAPHQL
PARA QUEM *(NUNCA)*
OUVIU FALAR DE
GRAPHQL



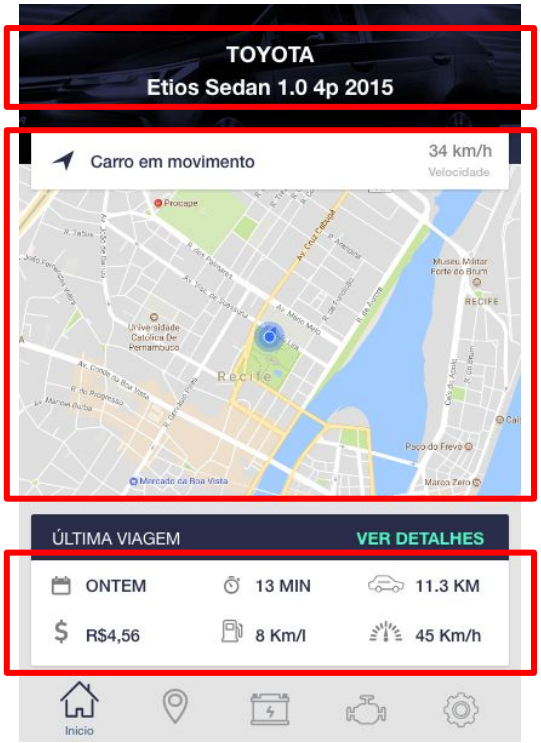
LINGUAGEM DE **CONSULTA**,
DESENVOLVIDA PELO
FACEBOOK EM 2015

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
   ...
}
```

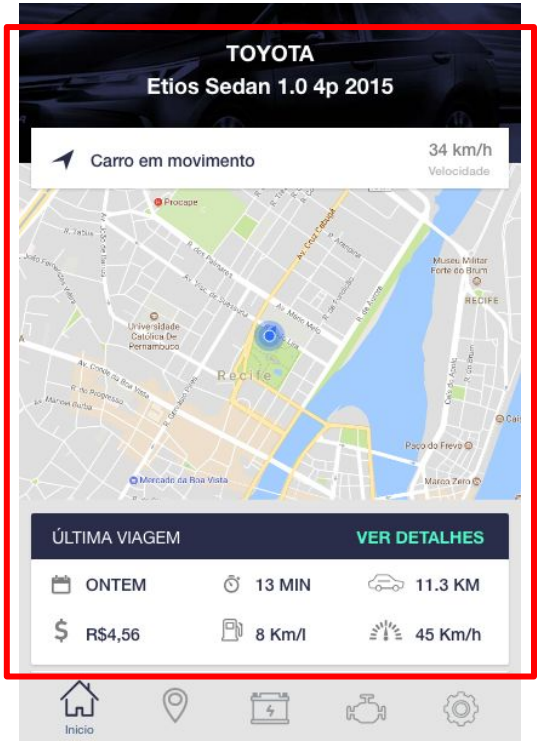
GET -> /TRIPS/:CARID/LAST

```
{
   "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

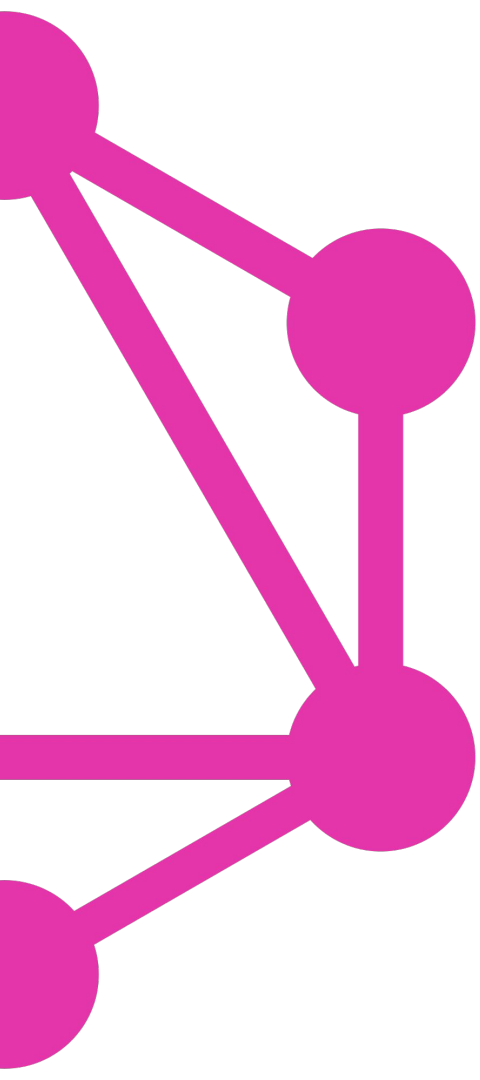


POST -> /GRAPHQL

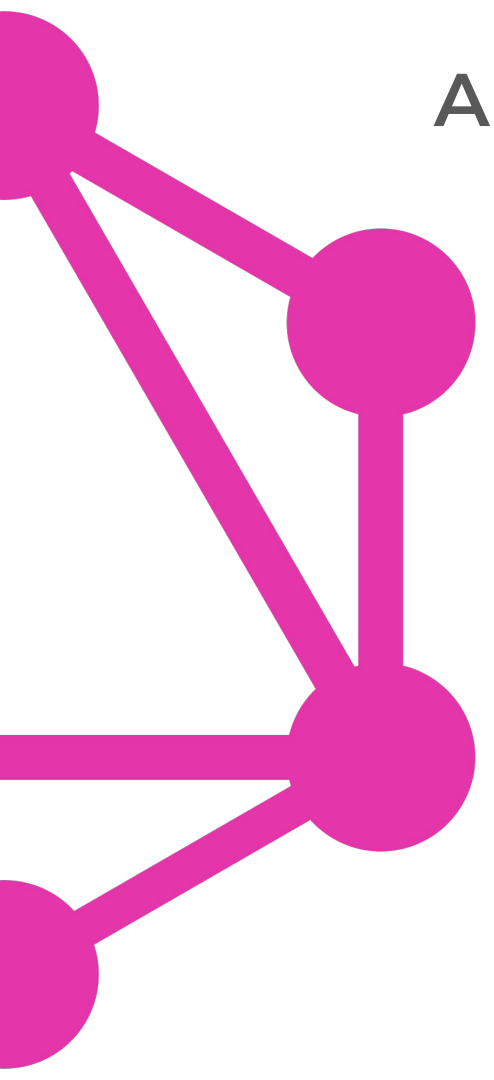
```
{
  car {
    montadora
    modelo
    ano
    location {
      latitude
      longitude
      velocidade
    }
    lastTrip {
      data
      duracao
      distancia
      custo
      consumo
      velocidade
    }
  }
}
```

```
{
  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** **COMO** OS DADOS SERÃO **CONSUMIDOS**



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

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

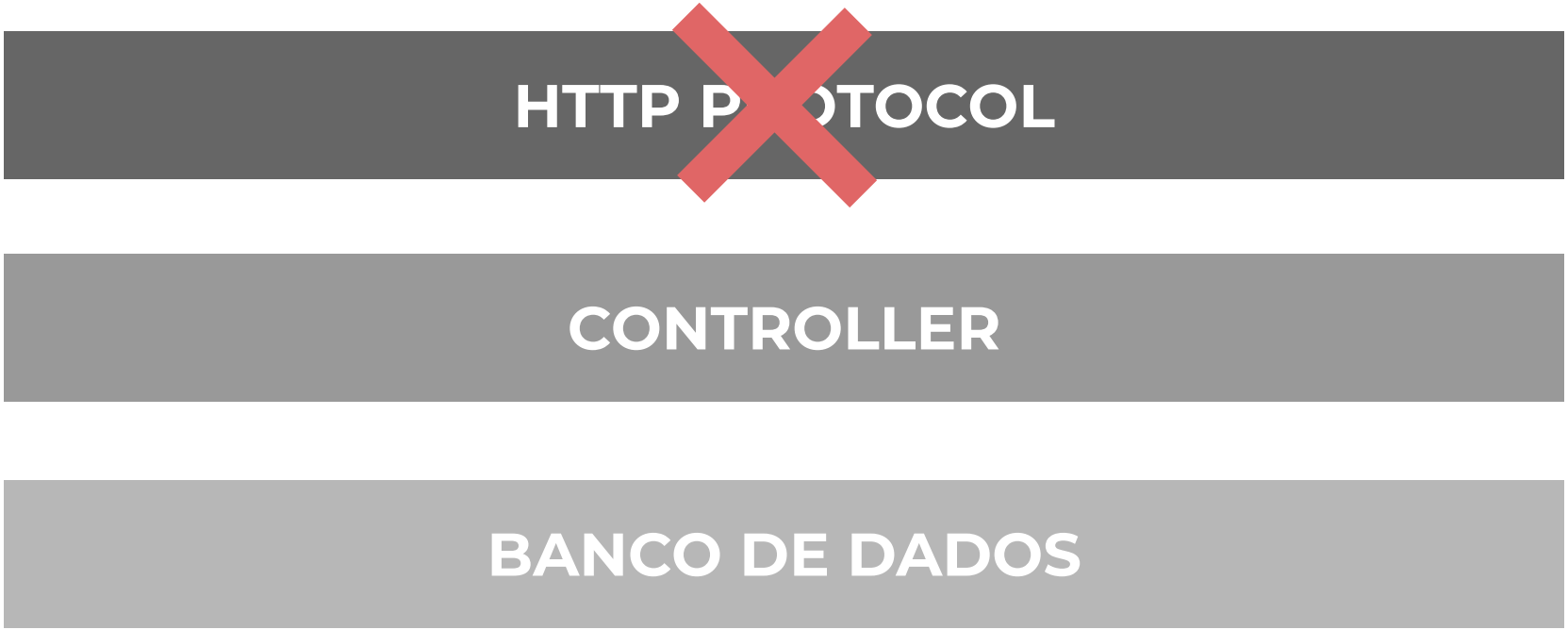
```
type Location {
  latitude: Float
  longitude: Float
  velocidade: Int
}
```

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

COMO TENTAMOS TESTAR GRAPHQL NO COMEÇO

MANDA A REQUISIÇÃO E **VALIDA** O RETORNO DO MESMO JEITO QUE **REST**!

ARQUITETURA DE ALTO NÍVEL



MAS GRAPHQL TAMBÉM USA HTTP, NÃO?

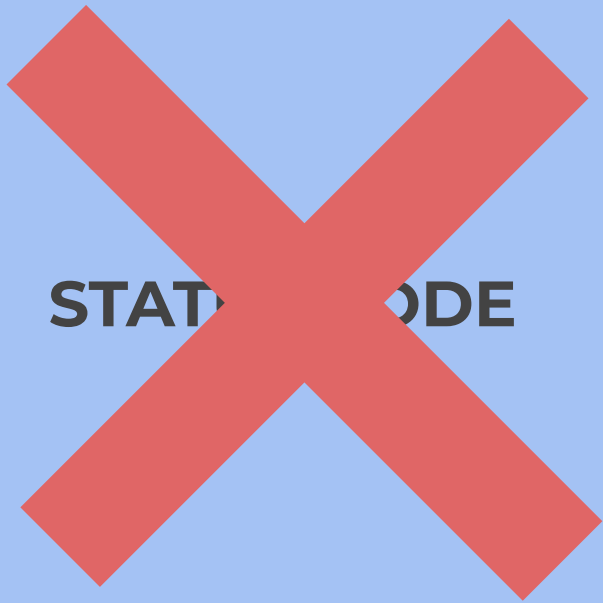
GRAPHQL UTILIZA HTTP COMO MEIO DE COMUNICAÇÃO, MAS **NÃO É ORIENTADO AOS SEUS MÉTODOS E STATUS CODE**

ARQUITETURA DE ALTO NÍVEL

VALIDAÇÕES

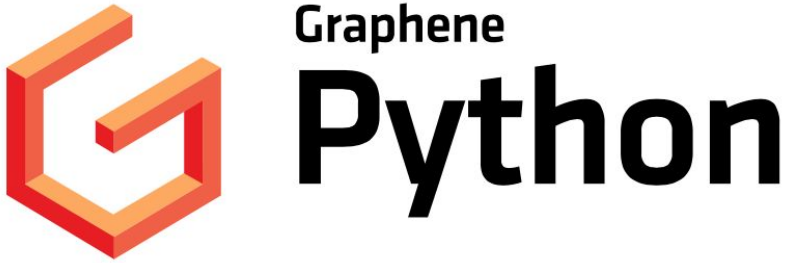
RESPONSE BODY

STAT ODE



CHEGA DE TEORIA, BORA VER COMO FUNCIONA NA PRÁTICA?

PYTHON + DJANGO + GRAPHENE + PYTEST



BENEFÍCIOS A CURTO PRAZO

BENEFÍCIOS A CURTO PRAZO

MEDIR COBERTURA

MEDIR COBERTURA

obd/tasks/datapackages.py	4	1	75%
obd/tasks/dongle.py	20	15	25%
obd/tasks/mileage_avg.py	20	0	100%
obd/tasks/set_data.py	141	33	77%
obd/urls.py	3	0	100%
obd/views/facebook_login.py	4	0	100%
obd/views/roadrover_settings.py	10	1	90%
vai/celery.py	17	7	59%
vai/logger.py	2	0	100%
vai/storage.py	9	0	100%
vai/test.py	6	0	100%
vai/urls.py	11	0	100%
vai/utils/admin.py	100	57	43%
vai/utils/converters.py	83	2	98%
vai/utils/datetime.py	78	42	46%
vai/utils/exceptions.py	77	16	79%
vai/utils/fence.py	13	0	100%
vai/utils/number.py	17	4	76%
vai/utils/requests/exceptions.py	21	7	67%
vai/utils/requests/forms.py	19	2	89%
vai/utils/requests/helper.py	42	5	88%
vai/utils/slack_webhook.py	7	3	57%
vai/utils/validation.py	84	0	100%
vaiauth/admin.py	28	3	89%
vaiauth/auth.py	15	9	40%
vaiauth/forms.py	63	34	46%
vaiauth/managers.py	19	5	74%
vaiauth/models.py	104	24	77%
vaiauth/views.py	18	10	44%
TOTAL	7572	2123	72%

C.E.S.A.R

Vai

BENEFÍCIOS A CURTO PRAZO

MEDIR COBERTURA

MOCK QUALQUER COISA

MOCK QUALQUER COISA: VIACEP

```
def test_should_find_address_by_cep(self):
        cep = '50070-000'
        valid_mocked_json = {
            'cep': cep,
            'logradouro': 'Rua dos mocks',
            'localidade': 'Recife',
            'uf': 'PE'
        }
        viacep.ViaCEP = Mock(return_value=valid_mocked_json)

        executed = self._execute_test_graphqlapi(
            request=Query.ADDRESS_BY_CEP,
            params={'cep': cep})
        data = executed.get('data')['location']['addressByCep']

        self.assertEqual(data['city']['name'], 'Recife')
        self.assertEqual(data['street'], 'Rua dos mocks')
        self.assertEqual(data['state']['name'], 'Pernambuco')
```

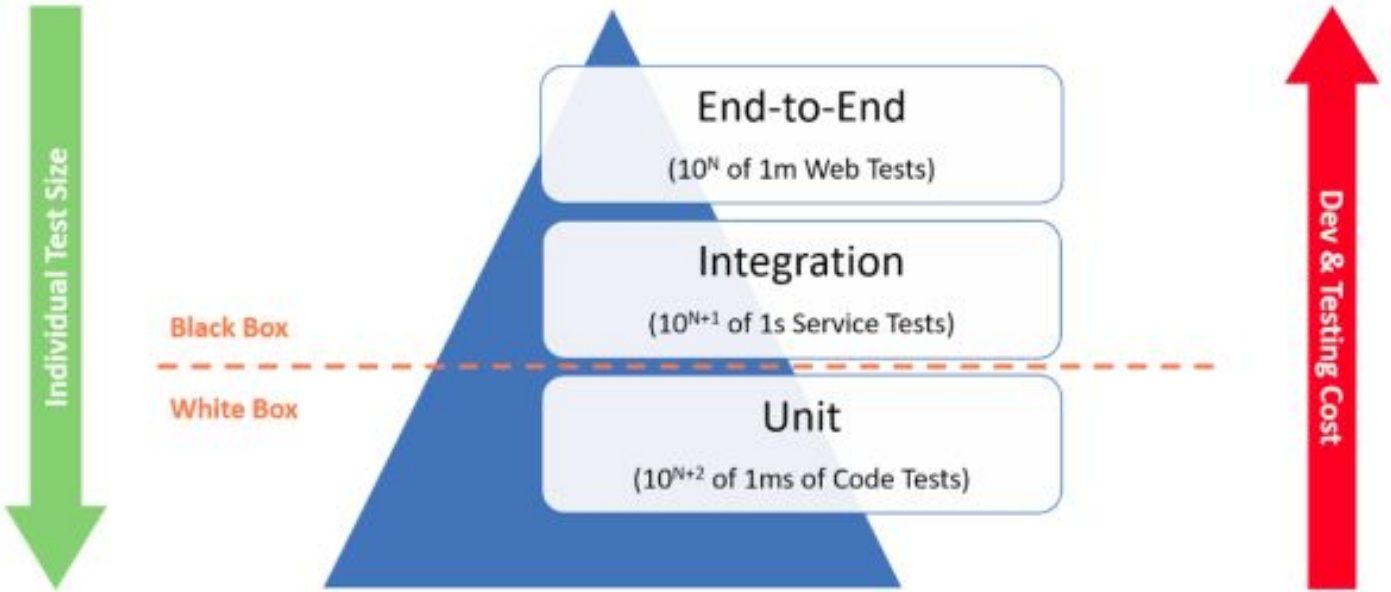
BENEFÍCIOS A CURTO PRAZO

MEDIR COBERTURA

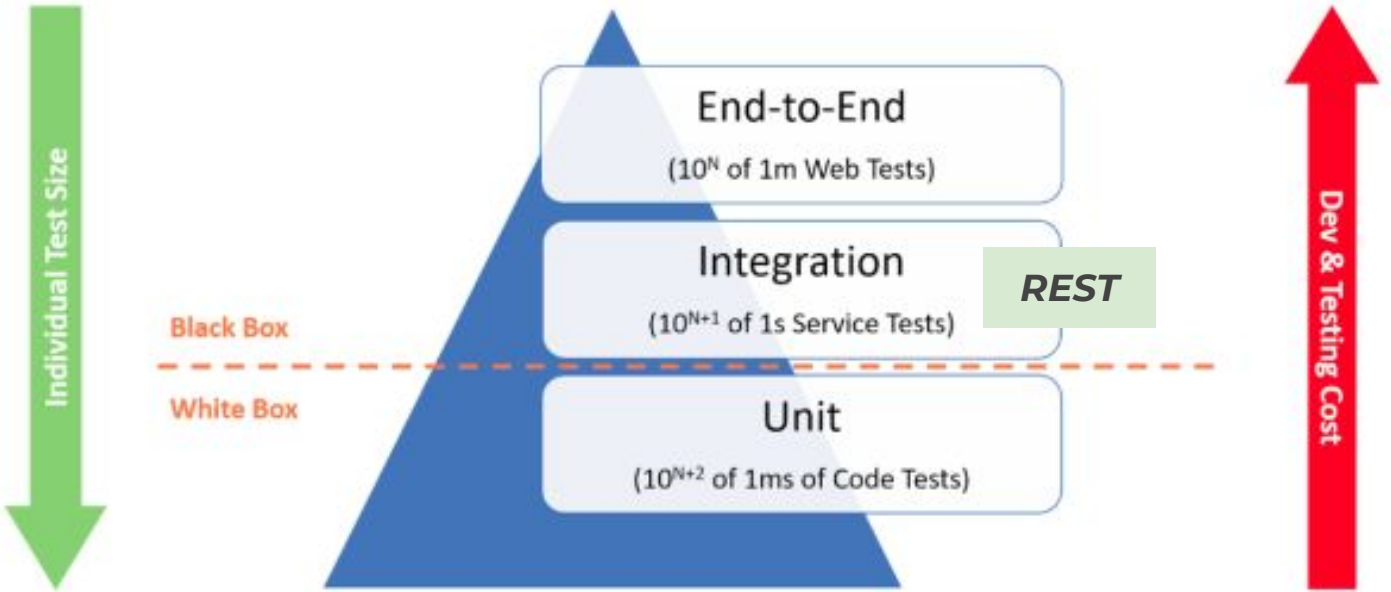
MOCK QUALQUER COISA

TEMPO DE EXECUÇÃO

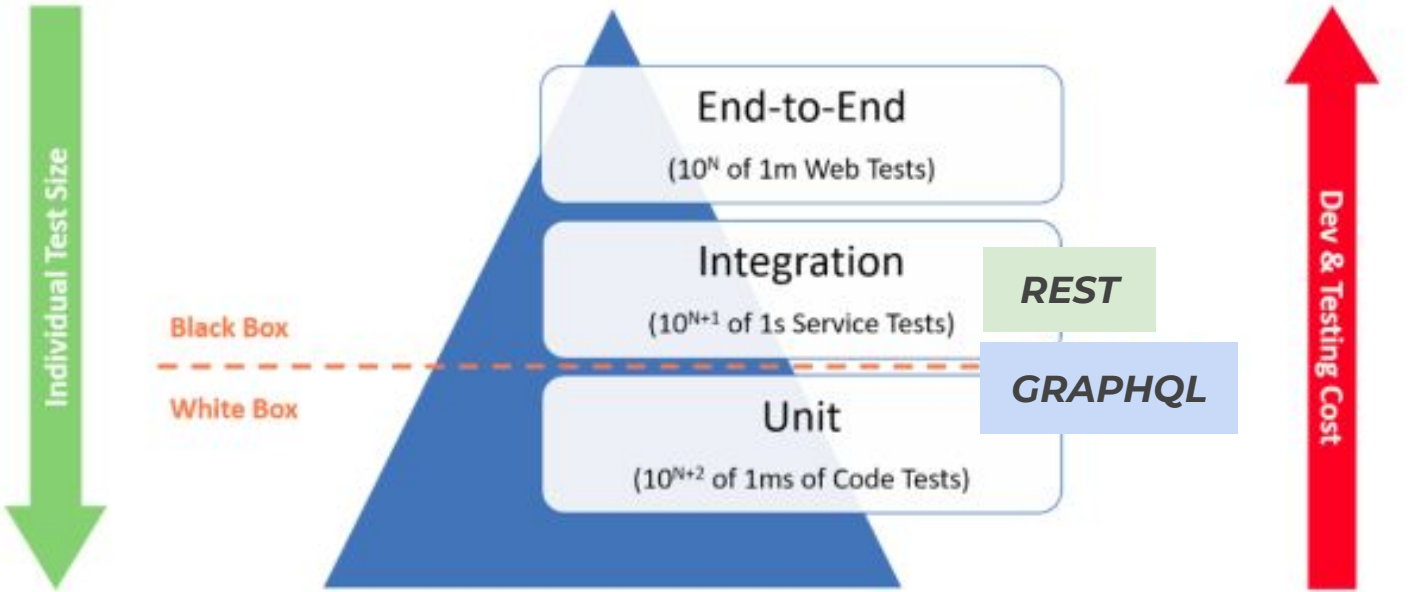
TEST PYRAMID



TEST PYRAMID



TEST PYRAMID



DÚVIDAS, APLICABILIDADE DE PROJETO, EXPERIÊNCIAS?

VALEU!

vai.com.vc

CARLOS LEAL
cels@cesar.org.br

DANIEL CÂNDIDO
dcs2@cesar.org.br

C.E.S.A.R