mts20t2-d3-g8

460ms | 11 | 22

✓ 22 | × 0

GraphQL External - Transfer

/test/graphql/external/transferExternal.test.js

17ms | 4 | ✓ 4

✓ Validar que é possível transferir grana entre duas contas 6ms

```
const respostaEsperada = require('../fixture/respostas/transferencia/validarQueEPossivelTransferi
const respostaTransferencia = await request(process.env.BASE_URL_GRAPHQL)
    .post('')
    .set('Authorization', `Bearer ${token}`)
    .send(createTransfer);
expect(respostaTransferencia.status).to.equal(200);
expect(respostaTransferencia.body.data.createTransfer)
    .excluding('date')
    .to.deep.equal(respostaEsperada.data.createTransfer);
```

✓ Testando a regra relacionada a Saldo insuficiente 3ms

```
const respostaTransferencia = await request(process.env.BASE_URL_GRAPHQL)
    .post('')
    .set('Authorization', `Bearer ${token}`)
    .send(teste.createTransfer);
expect(respostaTransferencia.status).to.equal(200);
expect(respostaTransferencia.body.errors[0].message).to.equal(teste.mensagemEsperada);
```

✓ Testando a regra relacionada a Remetente não encontrado 3ms

```
const respostaTransferencia = await request(process.env.BASE_URL_GRAPHQL)
    .post('')
    .set('Authorization', `Bearer ${token}`)
    .send(teste.createTransfer);
expect(respostaTransferencia.status).to.equal(200);
expect(respostaTransferencia.body.errors[0].message).to.equal(teste.mensagemEsperada);
```

✓ Testando a regra relacionada a Destinatário não encontrado 5ms

```
const respostaTransferencia = await request(process.env.BASE_URL_GRAPHQL)
    .post('')
    .set('Authorization', `Bearer ${token}`)
    .send(teste.createTransfer);
expect(respostaTransferencia.status).to.equal(200);
expect(respostaTransferencia.body.errors[0].message).to.equal(teste.mensagemEsperada);
```

mts20t2-d3-g8

460ms | 11 | 22

✓ | ×

/test/rest/controller/transferController.test.js

27ms | 3 | ✓ 3

✓ Quando informo remetente e destinatario inexistentes recebo (400 BAD REQUEST) 11ms

```
const resposta = await request(app)
    .post('/transfers')
    .set('Authorization', `Bearer ${token}`)
    .send({
        from: "julio",
        to: "isabelle",
        value: 100
    });

expect(resposta.status).to.equal(400);
expect(resposta.body).to.have.property('error', 'Usuário remetente ou destinatário não encontra
```

✓ Usando Mocks: Quando informo remetente e destinatario inexistentes recebo (400 BAD REQUEST)
8ms

```
// Mocar apenas a função transfer do Service
const transferServiceMock = sinon.stub(transferService, 'transfer');
transferServiceMock.throws(new Error('Usuário remetente ou destinatário não encontrado'));
const resposta = await request(app)
    .post('/transfers')
    .set('Authorization', `Bearer ${token}`)
    .send({
        from: "julio",
        to: "priscila",
        value: 100
    });

expect(resposta.status).to.equal(400);
expect(resposta.body).to.have.property('error', 'Usuário remetente ou destinatário não encontra
```

✓ Usando Mocks: Quando informo valores válidos eu tenho sucesso (201 CREATED) 8ms

```
// Mocar apenas a função transfer do Service
const transferServiceMock = sinon.stub(transferService, 'transfer');
transferServiceMock.returns({
    from: "julio",
    to: "priscila",
    value: 100,
    date: new Date().toISOString()
});
const resposta = await request(app)
    .post('/transfers')
    .set('Authorization', `Bearer ${token}`)
    .send({
```

mts20t2-d3-g8

460ms 11 22

✓ ×

```
const respostaEsperada = require('../fixture/respostas/quandoInformoValoresValidosEuTenhoSucess
delete resposta.body.date;
delete respostaEsperada.date;
expect(resposta.body).to.deep.equal(respostaEsperada);
// Um expect para comparar a Resposta.body com a String contida no arquivo
// expect(resposta.body).to.have.property('from', 'julio');
// expect(resposta.body).to.have.property('to', 'priscila');
// expect(resposta.body).to.have.property('value', 100);
```

GET /transfers

/test/rest/controller/transferController.test.js

33ms 2 ✓ 2

✓ Quando consulto a lista de transferências autenticado, recebo um array 29ms

```
// Realiza login para obter token
const respostaLogin = await request(app)
    .post('/users/login')
    .send({ username: 'julio', password: '123456' });
const token = respostaLogin.body.token;
const resposta = await request(app)
    .get('/transfers')
    .set('Authorization', `Bearer ${token}`);
expect(resposta.status).to.equal(200);
expect(resposta.body).to.be.an('array');
```

✓ Quando consulto a lista de transferências sem token, recebo erro (401 UNAUTHORIZED) 4ms

```
const resposta = await request(app)
    .get('/transfers');
expect(resposta.status).to.equal(401);
expect(resposta.body).to.have.property('message', 'Token não fornecido.');
```

Rest External - Transfer

/test/rest/external/transferExternal.test.js

POST /transfers

/test/rest/external/transferExternal.test.js

25ms 5 ✓ 5

✓ Quando informo valores válidos eu tenho sucesso (201 CREATED) 5ms

```
const postTransfer = require('../fixture/requisicoes/transferencias/postTransfer.json');
const resposta = await request(process.env.BASE_URL_REST)
```

mts20t2-d3-g8

460ms 11 22

✓ ×

```
    .excluding('date')
    .to.deep.equal(respostaEsperada);
```

✓ Quando tento transferir sem token, recebo erro (401 UNAUTHORIZED) 3ms

```
const postTransfer = require('../fixture/requisicoes/transferencias/postTransfer.json');
const resposta = await request(process.env.BASE_URL_REST)
    .post('/transfers')
    .send(postTransfer);
expect(resposta.status).to.equal(401);
expect(resposta.body).to.have.property('message', 'Token não fornecido.');
```

✓ Testando a regra relacionada a Saldo insuficiente 4ms

```
const postTransfer = require('../fixture/requisicoes/transferencias/postTransfer.json');
const resposta = await request(process.env.BASE_URL_REST)
    .post('/transfers')
    .set('Authorization', `Bearer ${token}`)
    .send(teste.postTransfer);
expect(resposta.status).to.equal(teste.statusCode);
expect(resposta.body).to.have.property('error', teste.mensagemEsperada)
```

✓ Testando a regra relacionada a Remetente não encontrado 8ms

```
const postTransfer = require('../fixture/requisicoes/transferencias/postTransfer.json');
const resposta = await request(process.env.BASE_URL_REST)
    .post('/transfers')
    .set('Authorization', `Bearer ${token}`)
    .send(teste.postTransfer);
expect(resposta.status).to.equal(teste.statusCode);
expect(resposta.body).to.have.property('error', teste.mensagemEsperada)
```

✓ Testando a regra relacionada a Destinatário não encontrado 5ms

```
const postTransfer = require('../fixture/requisicoes/transferencias/postTransfer.json');
const resposta = await request(process.env.BASE_URL_REST)
    .post('/transfers')
    .set('Authorization', `Bearer ${token}`)
    .send(teste.postTransfer);
expect(resposta.status).to.equal(teste.statusCode);
expect(resposta.body).to.have.property('error', teste.mensagemEsperada)
```

GET /transfers

/test/rest/external/transferExternal.test.js

37ms 2 ✓ 2

mts20t2-d3-g8

460ms 11 22

✓ ×

```
const token = respostaLogin.body.token;
const resposta = await request(process.env.BASE_URL_REST)
    .get('/transfers')
    .set('Authorization', `Bearer ${token}`);
expect(resposta.status).to.equal(200);
expect(resposta.body).to.be.an('array');
```

✓ Quando consulto a lista de transferências sem token, recebo erro (401 UNAUTHORIZED) 2ms

```
const resposta = await request(process.env.BASE_URL_REST)
    .get('/transfers');
expect(resposta.status).to.equal(401);
expect(resposta.body).to.have.property('message', 'Token não fornecido.');
```

Rest External - User

/test/rest/external/userExternal.test.js

POST /users/register

/test/rest/external/userExternal.test.js

27ms 2 ✓ 2

✓ Quando informo dados válidos, registro o usuário com sucesso (201 CREATED) 24ms

```
const novoUsuario = {
    username: `usuario_teste_${Date.now()}`,
    password: 'senha123',
    favorecidos: ['julio']
};
const resposta = await request(process.env.BASE_URL_REST)
    .post('/users/register')
    .send(novoUsuario);
expect(resposta.status).to.equal(201);
expect(resposta.body).to.have.property('username', novoUsuario.username);
```

✓ Quando tento registrar um usuário já existente, recebo erro (400 BAD REQUEST) 3ms

```
const usuarioExistente = {
    username: 'julio',
    password: 'senha123',
    favorecidos: ['priscila']
};
const resposta = await request(process.env.BASE_URL_REST)
    .post('/users/register')
    .send(usuarioExistente);
expect(resposta.status).to.equal(400);
expect(resposta.body).to.have.property('error');
```

mts20t2-d3-g8

460ms 11 22

✓ ×

✓ Quando informo credenciais válidas, faço login com sucesso (200 OK) 25ms

```
const resposta = await request(process.env.BASE_URL_REST)
    .post('/users/login')
    .send({ username: 'julio', password: '123456' });
expect(resposta.status).to.equal(200);
expect(resposta.body).to.have.property('token');
```

✓ Quando informo senha inválida, recebo erro (400 BAD REQUEST) 25ms

```
const resposta = await request(process.env.BASE_URL_REST)
    .post('/users/login')
    .send({ username: 'julio', password: 'senha_errada' });
expect(resposta.status).to.equal(400);
expect(resposta.body).to.have.property('error');
```

✓ Quando informo usuário inexistente, recebo erro (400 BAD REQUEST) 3ms

```
const resposta = await request(process.env.BASE_URL_REST)
    .post('/users/login')
    .send({ username: 'usuario_inexistente', password: 'qualquer_senha' });
expect(resposta.status).to.equal(400);
expect(resposta.body).to.have.property('error');
```

GET /users

/test/rest/external/userExternal.test.js

9ms 1 ✓ 1

✓ Quando consulto a lista de usuários, recebo um array com usuários 9ms

```
const resposta = await request(process.env.BASE_URL_REST)
    .get('/users');
expect(resposta.status).to.equal(200);
expect(resposta.body).to.be.an('array');
```