# Plano de Testes

## 1. Visão Geral

Serão implementados testes em relação às features do segundo protótipo da aplicação VisualSocial. Entre os testes a serem implementados, estão:

- Criação, visualização e deleção de post(F1, F2 e F3)
    - UC8 : RF5
    - UC9 : RF5
    - UC20: RF6
    - UC18 : RF6
- Criação de likes/comentários(F4 e F5)
    - UC17 : RF3
    - UC14 : RF2
- Exclusão de likes/comentários apenas pelo dono do post(F6 e F7)
    - UC16 : RF3

## 2. Casos de Testes

### 2.1 - Testes Unitários

#### 2.1.1 - Teste de unidade para a classe PostService

- **CT-TU-01-c1:** Testar se todos os posts são retornados paginados como o esperado
- **CT-TU-01-c2:** Testar se todos os likes de um post são retornados paginados como o esperado
- **CT-TU-01-c3:** Testar se todos os comentários de um post são retornados paginados como o esperado
- **CT-TU-01-c4:** Testar se um comentário é criado da maneira correta
- **CT-TU-01-c5:** Testar se uma exceção é lançada caso um usuário tente comentar mais de uma vez no mesmo post
- **CT-TU-01-c6**: Testar se um like é criado da maneira correta
- **CT-TU-01-c7:** Testar se uma exceção é lançada caso um usuário tente dar like mais de uma vez no mesmo post
- **CT-TU-01-c8:** Testar se um post em específico é retornado com os dados corretos
- **CT-TU-01-c9:** Testar se exceções são lançadas para erros internos nos casos de teste acima

#### 2.1.2 - Teste de unidade para a classe PostController

- **CT-TU-02-c1:** Testar se todos os posts são retornados paginados como o esperado
- **CT-TU-02-c2:** Testar se todos os likes de um post são retornados paginados como o esperado
- **CT-TU-02-c3:**Testar se todos os comentários de um post são retornados paginados como o esperado

- **CT-TU-02-c4:** Testar se o id do post criado é retornado de forma correta
- **CT-TU-02-c5:** Testar se o id do comentário criado é retornado de forma correta
- **CT-TU-02-c6:** Testar se o id do like criado é retornado de forma correta

### 2.2 - Testes end-to-end(e2e)

#### 2.2.1 - Teste end-to-end para as features do protótipo 2

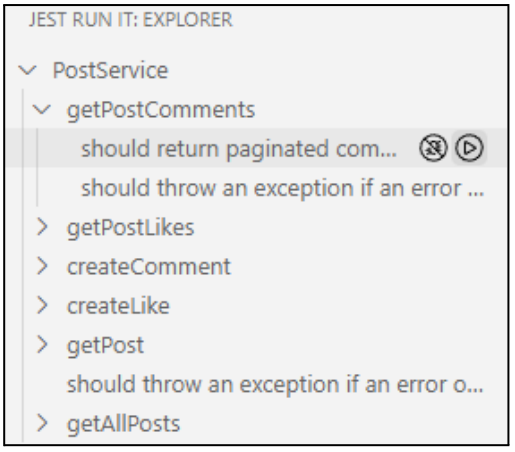
- **CT-TE2E-01-c1:** Testar se um post é retornado com as propriedades corretas após devidas transformações do DTO(Data-Transfer-Object) e retorno do status code correto
- **CT-TE2E-01-c2:** Testar se um post é criado corretamente por um usuário autorizado
- **CT-TE2E-01-c3:** Testar se uma exceção é lançado ao tentar criar um post com uma imagem em um formato inválido
- **CT-TE2E-01-c4:** Testar se uma exceção é lançada quando um usuário não autorizado tenta criar um post
- **CT-TE2E-01-c5:** Testar se todos os posts são retornados paginados como esperado
- **CT-TE2E-01-c6:** Testar se um comentário é criado de forma correta
- **CT-TE2E-01-c7:** Testar se uma exceção é lançada caso um usuário tente comentar mais de uma vez no mesmo post
- **CT-TE2E-01-c8:** Testar se um like é criado da maneira correta
- **CT-TE2E-01-c9:** Testar se todos os likes de um post são retornados paginados como o esperado
- **CT-TE2E-01-c10:** Testar todos os comentários de um post são retornados paginados como o esperado

## 3. Procedimentos de Teste

### 3.1 - Testes Unitários

O NestJs oferece recursos que suportam a realização de testes unitários através da biblioteca Jest. Ao criar um novo módulo na aplicação, arquivos de testes relacionados com a extensão *.spec* são criados automaticamente. Após as devidas configurações de dependências realizadas e a criação dos testes, a execução pode ser realizada via terminal, com o comando **npm test***,* que executará todos os testes encontrados.

Uma outra alternativa para a execução seria a utilização de uma extensão, como, por exemplo, a extensão **Jest Run It ,**que oferecerá uma interface interativa para tal tarefa.




### 3.2 - Testes e2e

Assim como os testes unitários, o NestJs também oferece suporte para testes e2e, fazendo uso da biblioteca supertest. Os arquivos possuem a extensão *.e2e-spec.* Ao fazer testes que interagem diretamente com a base de dados, é importante separar os dados da base de dados real dos dados que são utilizados para teste. Portanto, é recomendado utilizar uma outra instância da base de dados que sua estrutura seja uma cópia da real.

Para isso, antes de iniciar qualquer teste, o script a ser executado no caso da presente aplicação é **dotenv -e .env.test -- prisma migrate reset --force && dotenv -e .env.test -- prisma migrate deploy.** Esse script utiliza as variáveis de ambiente do ambiente de desenvolvimento para derrubar e subir novamente a base de dados de teste, de modo que os testes sejam executados em uma base de dados íntegra, sem perigo de conflitos.

Após isso, o script utilizado para rodar o teste é **dotenv -e .env.test -- jest -i --no-cache --detectOpenHandles --config ./test/jest-e2e.json,** que também utiliza as variáveis de ambiente de desenvolvimento para se conectar na base de dados correta.