Relatório de Testes da aplicação Postify

Para realizar os testes, utilizou-se da importação dos módulos necessários, como User, EmailConfirmation, Post e outros. Além disso, utilizou-se da biblioteca Faker para gerar campos de entradas mais rapidamente, como username, email e outros.

Testes Unitários

UserModelTestCase:

- **test_user_creation**: Verifica se um usuário pode ser criado com sucesso usando o método User.objects.create. Ele cria um usuário com dados gerados aleatoriamente e verifica se a chave primária (pk) do usuário foi atribuída um valor, indicando que o usuário foi criado com sucesso.
- **test_user_delete**: Verifica se um usuário pode ser excluído com sucesso usando o método User.delete. Ele cria um usuário, o exclui e verifica se o atributo is_active do usuário é definido como False, indicando que o usuário foi excluído.
- **test_user_generate_password_reset_token**: Verifica se um token de redefinição de senha pode ser gerado com sucesso usando o método **"User.generate_password_reset_token"**. Ele cria um usuário, gera um token de redefinição de senha para esse usuário e verifica se o campo **password_reset_token** do usuário não está vazio, indicando que o token foi gerado.
- **test_user_reset_password**: Verifica se a senha de um usuário pode ser redefinida com sucesso usando o método **User.reset_password**. Ele cria um usuário, gera um token de redefinição de senha, redefine a senha do usuário para uma nova senha aleatória e verifica se a senha do usuário corresponde à nova senha definida.
- t**est_send_password_reset_email**: Verifica se um e-mail de redefinição de senha pode ser enviado com sucesso usando o método **User.send_password_reset_email**. Ele cria um usuário, gera um token de redefinição de senha, envia um e-mail de redefinição de senha e verifica se o campo **password_reset_token** do usuário não está vazio, indicando que o token foi gerado e o e-mail foi enviado.
- **tearDown**: Este método é executado após cada teste na classe **UserModelTestCase**. Ele exclui todos os usuários criados durante os testes para limpar o ambiente de teste.

EmailConfirmationModelTestCase:

- **test_email_confirmation_creation**: Verifica se uma confirmação de e-mail pode ser criada com sucesso usando o método EmailConfirmation.objects.create. Ele cria um usuário, cria uma confirmação de e-mail para esse usuário e verifica se a chave primária (pk) da confirmação de e-mail foi atribuída um valor, indicando que a confirmação de e-mail foi criada com sucesso.
- **test_email_confirmation_delete**: Verifica se uma confirmação de e-mail pode ser excluída com sucesso usando o método **EmailConfirmation.delete.** Ele cria um usuário, cria uma confirmação de e-mail para esse usuário, exclui a confirmação de e-mail e verifica se a confirmação de e-mail não existe mais no banco de dados.

PostModelTestCase:

- **test_post_creation:** Verifica se uma postagem pode ser criada com sucesso usando o método **Post.objects.create**. Ele cria um usuário, cria uma postagem para esse usuário e verifica se a chave primária (pk) da postagem foi atribuída um valor, indicando que a postagem foi criada com sucesso.
- **test_post_delete:** Verifica se uma postagem pode ser excluída com sucesso usando o método Post.delete. Ele cria um usuário, cria uma postagem para esse usuário, exclui a postagem e verifica se a postagem não está mais ativa no banco de dados.

CommentModelTestCase:

- **test_comment_creation**: Verifica se um comentário pode ser criado com sucesso usando o método **Comment.objects.create**. Ele cria um usuário, cria uma postagem para esse usuário, cria um comentário para essa postagem e verifica se a chave primária (pk) do comentário foi atribuída um valor, indicando que o comentário foi criado com sucesso.
- **test_comment_delete**: Verifica se um comentário pode ser excluído com sucesso usando o método **Comment.delete**. Ele cria um usuário, cria uma postagem para esse usuário, cria um comentário para essa postagem, exclui o comentário e verifica se o comentário não existe mais no banco de dados.

LikeModelTestCase:

- **test_like_creation**: Verifica se um like pode ser criado com sucesso usando o método Like.objects.create.
- **test_like_delete:** Verifica se um like pode ser excluído com sucesso usando o método Like.delete.

DeslikeModelTestCase:

- **test_deslike_creation:** Verifica se um deslike pode ser criado com sucesso usando o método Deslike.objects.create.
- **test_deslike_delete**: Verifica se um deslike pode ser excluído com sucesso usando o método Deslike.delete.

Testes de Integração

UserRegisterIntegrationTestCase:

- test_user_register: Verifica se um usuário pode ser registrado com sucesso através da rota /api/v1/register/ com o método POST.
- test_user_register_invalid_email: Verifica se um usuário não pode ser registrado com um e-mail inválido através da rota /api/v1/register/ com o método POST.

UserPostIntegrationTestCase:

- test_user_post: Verifica se um usuário pode criar uma postagem com sucesso através da rota /api/v1/users/{user.pk}/posts/create com o método POST.
- test_user_post_invalid_image: Verifica se uma postagem não pode ser criada sem uma imagem válida através da rota /api/v1/users/{user.pk}/posts/create com o método POST.

UserPostCommentIntegrationTestCase:

test_user_post_comment: Verifica se um usuário pode comentar em uma postagem com sucesso através da rota /api/v1/posts/{post.pk}/comments/create com o método POST.

Testes de Sistema

SystemTestCase:

test_user_interaction_flow: Simula o fluxo de interação de um usuário com o sistema, incluindo registro, login, criação de postagem, comentário em postagem, adição e remoção de likes e deslikes em postagens, e exclusão de postagens e usuários.