A parte de testes de softwares é um importante tópico para garantir o íntegro funcionamento dele, e permitir ao usuário usufruir da melhor experiência de tecnologia. Para isso, foi decidido que os testes devem simular a realidade e devem apresentar um "teste de fogo" para garantir que ele dê conta de toda a demanda gerada por usuários.

Para isso, faremos testes para as camadas do sistema (front-end, back-end e a camada de dados), garantindo assim, um teste de integração entre as partes e o perfeito funcionamento do sistema. Assim, usaremos o Pytest para integrar módulos e verificar a comunicação entre APIs ou serviços internos e identificar falhas na integração de forma antecipada.

Faremos também, testes de stress no sistema, a fim de garantir a performance ideal do software em casos de muitos acessos simultâneos, ou até mesmo, em casos de diversas requisições sendo feitas ao mesmo tempo para a API.

Ainda usando o Pytest, faremos testes de segurança, para comprovar a confiabilidade do sistema, validando com testes reais de problemas de segurança, como: injeção de SQL, XSS (Cross-Site Scripting), CSRF (Cross-Site Request Forgery), entre outros. O SonarQube também pode ser configurado para detectar possíveis falhas de segurança no código, como vulnerabilidades em bibliotecas externas.

Por último, é importante verificar a experiência do usuário final, garantindo que a interface seja intuitiva e fácil de usar. Testes manuais podem ser realizados para garantir que a interação com o software seja a esperada e garantir que fluxos principais de navegação estejam funcionando sem erros.

Para a manutenção do projeto, será garantido, junto com o Pytest, um código eficaz e otimizado, para garantir a máxima performance nas máquinas dos usuários. Uma documentação atualizada é essencial para a manutenção do software, uma vez que, ela é a primeira ferramenta de consulta do usuário para verificar a funcionalidade ideal do sistema.

A manutenção de software não se resume apenas a corrigir problemas; ela também envolve melhoria contínua. Isso significa não só aprimorar a qualidade do código, mas também adaptar o sistema às mudanças nas necessidades dos usuários ou às inovações tecnológicas.

Para a parte de DevOps, foi utilizado para controle de versão, foi utilizado o Git e GitHub, que são ferramentas muito usadas para a integração do desenvolvimento e do histórico de versões do desenvolvimento de um projeto. O GitHub Actions é integrado de forma a automatizar os testes e execução de scripts, baseado em eventos como a automatização de testes para garantir o funcionamento de versões mais recentes, garantindo assim, a integração contínua e íntegra do projeto. E por fim, haverá uma automação, após os testes de funcionamento, para que seja integrado as partes funcionais mais recentes ao projeto main e o usuário possa utilizar a novas features de forma mais rápida.