Plano de Teste

Amanda Eufrasio, Gabriel Martins, Jéssica Serqueira,
Marcus Vinicius, Rafael Nascimento

PLANO DE TESTE PROJETO EIXO 3 - TURMA 6 - 19:20

Belo Horizonte

2023

1. Introdução

Plano de teste do projeto do 3º semestre “Desenvolvimento de uma Aplicação Móvel em um Ambiente de Negócio”. Visando facilitar a adoção de animais de estimação e evitar o agravamento do quadro emocional dos pets, desenvolvemos este app. Neste documento está contido o plano de teste da aplicação, podendo ser alterado e revisado a qualquer momento, se assim for de desejo dos envolvidos no projeto, principalmente considerando a certeza de imprevistos durante a implementação dos teste, bem como, alterações da documentação oriundas de verificação e/ou validação de documentos.

2. Requisitos a serem testados

Os requisitos a serem testados estão listados abaixo de acordo com o documento oficial do projeto.

2.1. Requisitos Funcionais

RF - 01 : A página principal vai contar com botões que levam às telas de cadastro e login. - ALTA

RF - 02 : Tela de cadastro onde usuários poderão se cadastrar. - ALTA

RF - 03 : Tela de login de usuários. - ALTA

RF - 04 : A página inicial do usuário vai mostrar os cards com os pets cadastrados para adoção. - ALTA

RF - 05 : Dentro dos cards vai ter as listas com as informações com os animais para adoção - MÉDIA

RF - 06 : Os cadastros podem fazer edições em seus perfis. - MÉDIA

RF - 07 : Os cadastros podem fazer excluir seus perfis seus perfis. - MÉDIA

2.2. Requisitos não funcionais

RNF - 01 : A aplicação deve ser responsiva respeitando a maioria dos aparelhos - ALTA

RNF - 02 : O banco de dados deve ser armazenado dentro de um ambiente acessível publicamente na internet(Heroku ou AWS) - ALTA

RNF - 03 : O aplicativo deve ser compatível com os principais sistemas operacionais do mercado (android e IOS) - ALTA

RNF - 04 : O aplicativo deve ser compatível com os principais navegadores do mercado (Opera, Mozilla, Google Chrome, Firefox e Microsoft Edge) - ALTA

RNF - 05 : O aplicativo deve ter bom nível de contraste entre os elementos da tela em conformidade - MEDIA

RNF - 06 : O aplicativo irá contar com autenticação de login para reforçar a segurança dentro da nossa plataforma - BAIXA

RNF - 07 : O aplicativo deve ser publicado em um ambiente acessível publicamente na internet como site(GitHub ou Heroku) - BAIXA

3. Estratégias e ferramentas de teste

Aqui é descrito estratégias que poderão ser utilizadas para realização dos teste, bem como as possíveis ferramentas que estarão em uso durante todo o processo de implementação dos teste propostos aqui.

3.1. Tipos de teste

Abaixo estão um conjunto de testes e termos que serão utilizados durante o processo de desenvolvimento da aplicação.

3.2. Testes de integridade do banco de dados

O banco de dados deve seguir princípios **ACID**, os quais estão inclusos e descritos no link acima. Onde temos:

Atomicity (Atomicidade) - Todas as transações ou terão sucesso por completo, ou não acontecerão

Consistency (Consistência) - Todos os dados deverão ser consistentes e íntegros. Todos os dados devem ser válidos de acordo com as regras definidas

Isolation (Isolamento) - Todas as transações devem ser executadas de forma isolada, sem afetarem umas às outras

Durability (Durabilidade) - A integridade das transações não pode ser violada em caso de falha do sistema e/ou de perda de dados. O estado deve ser preservado

3.3. Testes de função

Os testes de função estão relacionados com o teste de métodos e funções síncronas e assíncronas do sistema. Aqui temos:

Testes de entrada e saída de dados - Testar com a entrada de múltiplos dados diferentes, esperados e não esperados para garantir que a saída será adequada. Aqui pode ser muito interessante a utilização de testes unitários para automatização do processo de input com dados variados

3.4. Testes de desempenho e segurança

Testes relacionados tanto ao tempo de resposta de aplicações, como para responder perguntas, tais quais:

Como o sistema responde sob pressão?
O quão rápido o sistema responde?
É rápido o suficiente?
Como podemos melhorar o desempenho?
Como o sistema responde em diversos aparelhos diferentes?
O sistema é seguro? Consegue lidar bem com métodos de invasão variados?
Quais vulnerabilidades o sistema pode apresentar?
O usuário é quem diz ser?
O usuário está devidamente autorizado a acessar esses dados?

Testes de tempo de resposta - Interessante aqui testar como funcionaria o sistema em caso de throttling, slow 3G, fast 3G, offline, etc. Podemos testar também como ele funciona e responde em equipamentos de diferentes custos

Testes de estresse - Testar como o sistema funciona em caso de situações 'extremas', onde temos menos controle e ferramentas. Podemos considerar aqui, também, a utilização de técnicas de pentesting, como, XSS, SQL injection, CSRF, etc

Testes de carga - Testar como o servidor lida em caso de muitas pessoas usando o sistema ao mesmo tempo. Verificar também, como o aplicativo como um todo, reage ao uso em grande escala, tanto na parte do banco de dados, quanto em caso de envios constantes de formulários

Testes de autenticação e autorização - Aqui visamos garantir que a verificação de identidade do usuário está de acordo com o que esperamos dela. Garantir também, que um usuário com as permissões adequadas, tenha acesso àquilo que diz respeito a ele. Outro ponto importante, é garantir que não tenhamos violação de integridade de constraints, em caso de usuários com email e/ou nome de usuário já presentes na base de dados, tentando um segundo registro sem exclusão do

anterior. Interessante considerar a autenticação em dois fatores e recuperação de senhas também. Por fim, é fundamental a utilização da criptografia em caso de dados sensíveis do usuário no banco de dados

3.5. Testes de interface (UI/UX)

Testes relacionados a parte estética da aplicação, onde aqui, encaixam-se assuntos como acessibilidade, usabilidade, engenharia cognitiva, comunicabilidade, harmonia de fontes e cores

Testes de comunicação - Visa responder se o sistema é comunicativo, e se comunica da forma certa e adequada com o usuário. Podemos citar métodos da engenharia semiótica (inspeção semiótica), tais quais:

Método de Inspeção Semiótica (MIS) - Ele avalia a comunicabilidade considerando a emissão da metacomunicação

Método de Avaliação da Comunicabilidade (MAC) - É um método de avaliação por observação. Avalia a comunicabilidade pela recepção da metacomunicação do designer codificada na interface

Testes de acessibilidade - Testes que verificam o quão bem minorias se adaptam ao sistema; em outras palavras, o quão inclusivo o sistema é. Podemos citar aqui o **Web Content Accessibility Guidelines (WCAG)** ou princípios de acessibilidade, que são: perceptível, operável, compreensível e robusto. A WCAG é útil na análise de acessibilidade

Testes de usabilidade - Busca quantificar, o quão fácil é para um sistema, ser utilizado pelos mais variados tipos de usuário. Podemos citar, por exemplo: EyeTracking, Clickstream e o **SUS**

3.6. Testes extra

Testes que não se encaixam nas situações anteriores

Testes de build e deploy - Aqui, eu proponho testar como o sistema funciona quando já hospedado, tanto na parte client side, quanto server side e data layer

Testes de verificação e validação - Verificar e validar o sistema (se possível a cada sprint ou ciclo), para garantir que o produto está de acordo com a documentação (DER, Wireframes, etc)

4. Equipe e infraestrutura

4.1. Equipe/Integrantes

A equipe do projeto é formada pelos seguintes integrantes: Amanda Eufrasio, Gabriel Martins, Jéssica Serqueira, Marcus Vinicius, Rafael Nascimento

4.2. Infraestrutura

O grupo se comunica através do Microsoft Teams e do Whatsapp principalmente. Boa parte da documentação foi feita utilizando ferramentas de planilha, documentação de texto e software, dando destaque ao Github como repositório remoto e ferramenta parcial para backlogs, definição de milestones, etc e ao Git como ferramenta de versionamento. Nota-se o uso de IDEs variadas, ferramentas de testes automatizados, possíveis pipelines de integração contínua, documentações de software, etc

5. Cronograma de atividades

O cronograma em maior parte é definido pela instituição de ensino, mas pode ser verificado de forma adaptada na aba de **‘milestones’ do projeto**, no Github

6. Documentação complementar

Toda a documentação pertinente ao projeto pode ser encontrada **no projeto do Github** na pasta **'docs'**

6.1. Bibliografia

https://www.cin.ufpe.br/~gta/rup-vc/extend.formal_resources/guidances/examples/resources/test_plan_v1.htm#_Toc449511158

https://database.guide/what-is-acid-in-databases/

https://en.wikipedia.org/wiki/System_usability_scale