Trabalho final 1

Ludum - Assistente virtual

Alunos:

Nome:	**GitHub:**	**Matricula**
André Bargas	@andrebargas	12/0110237
Brian Lui	@brian2397	15/0006802
Gustavo Duarte	@gustavoduartemoreira	15/0059957
Lucas Machado Martins	@lmmLucasMachado	15/0015917
Mateus Oliveira	@omateusp	16/0015006
João de Assis	@Jonjon667	17/0036634
Tâmara Barbosa Tavares	@tamarabarbosa	12/0022613

Universidade de Brasília - UNB
Faculdade Gama - FGA

Brasília, DF
Outubro, 2019

1. Indíce

2. Resumo

O presente trabalho, inserido na disciplina Testes de software, descreve como será o processo de validação definido para o assistente virtual Ludum. Aborda-se as técnicas utilizadas pela equipe para definir as questões a serem respondidas e define-se os testes utilizados que se adequam ao projeto e assim ao fim de toda a análise poder chegar a uma conclusão sobre o aspecto mensurar os critérios de qualidade. Para assim criar um plano de testes embasado, utiliza-se usar o modelo GQM (Goal Question Metric), que auxilia a definir os objetivos de teste relevantes do projeto e a estruturar o plano.

3. Introdução

3.1. Objeto

ChatBot Ludum, projeto realizado para as disciplinas Métodos de Desenvolvimento de Software (MDS) e Engenharia de Produto de Software (EPS), do curso de Engenharia de Software da Faculdade UnB Gama (FGA) da Universidade de Brasília (UnB). [1]

Essa é uma ferramenta que busca auxiliar os desenvolvedores de jogos ou pessoas que possuam interesse na utilização da biblioteca Pygame em formato de um chatbot que utiliza a linguagem de programação em ascensão Python

Ele mostra as perguntas frequentes relacionadas à esta biblioteca de jogos e recomenda materiais e links que venham a contribuir para o aprendizado, além de ensinar como desenvolver jogos com diversos níveis de complexidade e mostrar recomendações de configurações de ambiente caso o seja necessário.

3.2. Objetivo

O objetivo deste trabalho é avaliar a qualidade do produto de software do projeto Ludum. Teremos como base a norma ISO 9126 e foco principal são os aspectos ligados a **funcionalidade** do sistema.

Segundo a norma as principais sub-características do parâmetro da funcionalidade são :

- **Adequação**: Que é a capacidade do produto de fornecer um conjunto apropriado de funções para as tarefas e objetivos do usuário.

- **Acurácia**: A capacidade do produto de software de prover, com o grau de satisfação adequado, resultados ou efeitos corretos ou conforme o esperado.

- **Interoperabilidade**: Possibilidade do produto de software de interagir com um ou mais sistemas especificados.

- **Segurança de acesso**: A capacidade proteger as informações e os dados, de forma que pessoas não autorizados não tenham acesso para lê-los ou

modificá-los, além de de fornecer acesso aos sistemas autorizados ou pessoas.

- **Conformidade**: O produto de software não estar em desacordo com as normas, as convenções ou as regulamentações previstas em leis e prescrições similares relacionadas à funcionalidade.

As principais sub-características a serem exploradas neste trabalho são a **Adequação**, **Acurácia** e **Conformidade**. Foi assumido que cada uma destes pontos serão objetivos (Goals) a serem alcançados em uma modelagem baseada na abordagem GQM (Goal Question Metric). A partir destas metas foram elaborados questões que explicitam o nível operacional de cada um destes pontos, para assim modelarmos quais as métricas serão mais adequadas para a avaliação do sistema.

Nosso objetivo é o levantamento destas métricas e a avaliação do nível de conformidade das mesmas com as características desejadas e/ou planejadas para o sistema.

3.3 Motivação

O projeto foi realizado por alunos da disciplina Métodos de Desenvolvimento de Software (MDS), tendo o seu primeiro contato com a área de testes, esta é uma ótima oportunidade para testar e poder contribuir com esta ferramenta. Neste projeto não foram realizados muitos testes, fora alguns testes de usabilidade e testes unitários que cobrem parte razoável da aplicação, além disto não foi realizado quase nenhuma técnica de Teste de Software em si. -qualidade do software.

Vendo isto o projeto o Ludum, é um ótimo programa para se testar, pois é um software novo e compacto com requisitos definidos, além do fato de um dos membros do grupo ser um dos desenvolvedores do mesmo. Assim será possível realizar os tipos de testes exigidos pela disciplina de Testes da Universidade de Brasília do campus Gama.

4. Refencial Teórico

Neste tópico, estarão detalhadas as bases teóricas fundamentais para a compreensão deste trabalho.

4.1 ISO 9126

A ISO 9126 é um modelo de qualidade que utiliza uma abordagem top-down para decompor atributos que possuem alto nível de abstração em categorias e subcategorias. Ela subdivide manutenibilidade em analisabilidade, estabilidade, testabilidade e modificabilidade (DROMEY, 1995).

4.2 GQM

Goal Question Metric - **GQM** é uma abordagem de métrica de Software em engenharia de Software. A abordagem GQM estabelece um modelo de medição baseado em três níveis: o nível conceitual(Objetivo), o nível operacional (Questões), e o nível quantitativo (Métricas).

5. Métodos e Técnicas de Teste

5.1 Teste de sistema

Objetivo da Técnica:	Verifica se as unidades são compatíveis, além de validar a conformidade com a documentação.
Técnica:	Experimentar os recursos e fluxos ou funções de cada uma das histórias de usuário, utilizando dados válidos e inválidos para verificar se: ● os resultados esperados ocorrerão quando forem usados dados válidos; ● as mensagens de erro ou de aviso apropriadas serão exibidas quando forem usados dados inválidos; ● cada regra de negócio será aplicada de forma adequada.
Estratégias:	Realizar testes manuais verificando a funcionalidade das histórias de usuário e a interação das mesmas.
Critérios de Êxito:	Testar todas histórias de usuário.

5.2 Teste de carga

Objetivo da Técnica:	Verificar o limite de dados que o software suporta antes de apresentar defeitos.

Técnica:	Fazer diversas transações a fim de encontrar o limite da aplicação.
Estratégias:	Buscar uma ferramenta que auxilie a fazer este teste, fazendo um número de transações semelhante ao número real de transações esperadas.
Critérios de Êxito:	Testar as transações mais críticas para o sistema e ao analisar obter um resultado satisfatório.

5.3 Teste de stress

Objetivo da Técnica:	Verificar o comportamento do software ao processar uma grande quantidade de dados.
Técnica:	Processar um grande número de dados.
Estratégias:	Buscar uma ferramenta que auxilie a fazer este teste, colocando uma grande quantidade de dados um número maior que o esperado para a aplicação.
Critérios de Êxito:	Testar as mais partes mais críticas para o sistema e não ocorrer falhas.

5.4 Teste funcional

Objetivo da Técnica:	Verificar se o software se comporta adequadamente ao que se foi proposto.

Técnica:	Para este tipo de teste existem diferentes tecnicas para efetua-lo. ● Teste de Caixa-branca: Avalia o comportamento interno do software. ● Teste de Caixa-preta: Neste teste não se considera o funcionamento interno você deve apenas fornecer dados ou efetuar ações e ver o resultado.
Estratégias:	Realizar testes de usabilidade com grupos que venha a ser o público alvo desta aplicação.
Critérios de Êxito:	Testar as mais partes mais críticas para o sistema e não ocorrer falhas.

6. Ambiente de Teste

Os testes serão realizados no computador Dell G5 5590, com a seguinte Configuração:

- Intel Core i7-9750H (2.6 GHz até 4.5 GHz, cache de 12MB, hexa-core, 9ª geração);
- Memória de 16GB (2x8GB), DDR4, 2666MHz;
- Unidade de estado sólido (SSD) NVMe PCIe M.2 de 256 GB + Disco rígido (HDD) SATA 2,5" de 1TB (5400 RPM);

A aplicação será executada na mesma máquina em que será testada através de um ambiente Docker, para que seja evitado problemas de latência de rede e incompatibilidade de sistema operacional.

7. Métricas

GQM

Objetivo:

Verificar se as histórias de usuário estão sendo corretamente atendidas, de acordo com as funcionalidades do software.

Questões:

Quantas histórias de usuário estão sendo atendidas?

Quantos fluxos o software consegue atender sem erros?

Qual a quantidade de erros encontrados por história?

Métricas:

Histórias corretamente atendidas/Número de histórias

Fluxos com erros/Total de fluxos

Quantidade de erros/Número de histórias

Objetivo:

Verificar se o usuário atinge seu objetivo após a utilização do software

Questões:

Realizou todas as atividades propostas pelo chatbot?

O usuário se encontra feliz após a utilização do software?

O feedback foi bom ou ruim ?

Métricas:

Número de tarefas propostas pelo software que foram realizadas pelo usuário

Nível de felicidade do usuário após a utilização do software

Qualidade e quantidade do feedback

Objetivo:

Verificar se o software segue métricas e padrões de desenvolvimento de Software

Questões:

O código segue algum guia de estilo?

O código é de fácil entendimento?

O desenvolvimento foi bem documentado?

Métricas:

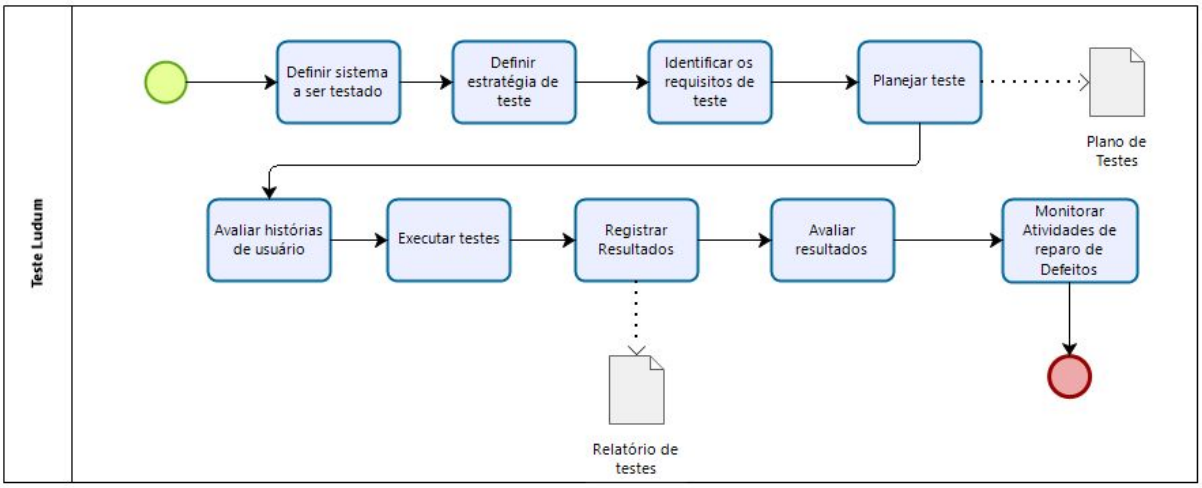
Porcentagem do código bem formatada

Quantidade de trechos de código comentado

Número de documentos gerados.

8. Fluxo de Trabalho de Teste

O procedimento do projeto será de acordo com o modelo abaixo:



1. Definir sistema a ser testado - Atividade onde é definido qual projeto será testado e as respectivas motivações.
2. Definir estrategia de teste - Atividade para definir qual técnica será utilizada nos testes
3. Identificar os requisitos de teste - Atividade em que se definem os requisitos necessários para realização dos testes como pessoas, hardware e software.
4. Planejar teste - atividade onde o plano de testes é contemplado.
5. Avaliar histórias de usuário - Atividade onde serão estudadas as histórias de usuário.
6. Executar testes - Nessa atividade serão utilizadas ferramentas para realizar os testes, além dos testes manuais necessários, como por exemplo testes de usabilidade.
7. Registrar resultados - Nessa atividade é registrado tudo que se encontrou na atividade anterior de erros e defeitos e assim estruturando o relatório de testes.
8. Avaliar resultados - Atividade para verificação do relatório de testes, estabelecer conclusões e fornecer para a equipe de desenvolvimento os problemas

encontrados.

9. Monitorar atividades de reparo de defeitos - Atividade de acompanhamento das possíveis correções dos problemas encontrados.

9. Referências

1. Ludum - O assistente virtual sobre o PyGame. <https://fga-eps-mds.github.io/2019.1-Ludum>. Acesso em 20 de outubro de 2019.
2. Vuk Vukovic. A Business Software Testing Process-Based Model Design. <https://www-worldscientific-com.ez54.periodicos.capes.gov.br/doi/pdf/10.1142/S0218194018500201>. Acesso em 20 de outubro de 2019.
3. Sommerville, Ian. Engenharia de Software. 9ª edição. São Paulo: Pearson Prentice Hall, 2011.
4. TESTE DE SOFTWARE. <https://www.testar.me/teste-de-software>.Acesso em 20 de outubro de 2019.
5. VON WANGENHEIM, Christiane Gresse. **Análise de custo benefício de mensuração baseada em GQM.** Florianópolis: A, 199.
6. ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR ISO/IEC 9126-1**: Engenharia e Software - Qualidade de produto. Rio de Janeiro: Abnt Editora, 2003. Disponível em: <https://jkolb.com.br/wp-content/uploads/2014/02/NBR-ISO_IEC-9126-1.pdf>. Acesso em: 20 out. 2019.