**Plano de Testes**

GameProfile

| **Professor:**  Marcelo Rômulo Fernandes | **Equipe:**  Dina Gomes  Giovanna Melo  Franklin Barbosa  Lázaro Lawrence  Patrick Matheus  Ruan Franklin |
| --- | --- |

**Julho / 2023**

**Histórico de Revisões**

| **Data** | **Versão** | **Descrição** | **Autor** |
| --- | --- | --- | --- |
| <11/07/22> | <1.0> | Versão final do documento | Game Profile |

**Índice**

**1. INTRODUÇÃO 4**

1.1 Objetivos 4

1.2 O Game Profile 4

1.3 Escopo 4

1.4 Identificação de Projeto 5

**2. REQUISITOS A TESTAR 5**

2.1 Teste Unitário 5

2.2 Análise Estática 5

2.3 Testes de Integração

2.4 Testes de Sistema 6

**3. ESTRATÉGIA DE TESTE 6**

3.1 Tipos de Teste 6

*3.1.1 Teste Unitário 6*

*3.1.2 Análise Estática 6*

*3.1.3 Teste de Integração 7*

*3.1.4 Teste de Sistema 7*

3.2 Ferramentas 8

**4. RECURSOS 8**

4.1 Trabalhadores 8

4.2 Sistema 10

**5. CRONOGRAMA 10**

**6. RESULTADOS 10**

6.1 Análises Estáticas 11

6.2 Testes Unitários 12

6.3 Testes de Integração 14

6.4 Testes de Sistema 15

6.1 Testes Rodando 16

1. **Introdução**

* 1. **Objetivos**

Esse documento do Plano de Testes do G.P. compõe-se dos seguintes objetivos:

• Identificar informações de projeto existentes e os componentes de software que devem ser testados.

• Listar os Requisitos a Testar recomendados (alto nível).

• Recomendar e descrever as estratégias de teste a serem empregadas.

• Identificar os recursos necessários e prover uma estimativa dos esforços de teste.

• Listar os elementos resultantes do projeto de testes.

* 1. **O Gameprofile**

O GameProfile (GP) fornece uma variedade de recursos para aprimorar a experiência in-game dos jogadores de e-sports. Ele permite que os usuários interajam entre si, criem grupos e compartilhem informações relevantes. A plataforma é projetada para facilitar a busca e a formação de equipes, com base nas informações e nas avaliações dos jogadores. É importante ressaltar que o GameProfile (GP) não é um jogo em si, mas sim uma rede social destinada a conectar jogadores e fornecer recursos que melhorem a dinâmica e a jogabilidade durante as partidas.

* 1. **Escopo**

O GP passará pelos testes unitário, de integração e de sistema. Os testes unitários e de integração vão lidar com a qualidade funcional, das bases de dados, interface gráfica e do controle de acesso; enquanto que os testes de sistema tratarão as questões de performance.

Os testes de configuração não serão realizados uma vez que os terminais utilizados serão dedicados exclusivamente para uso do sistema e todos os programas e dados necessários à instalação do GP nas máquinas dos recepcionistas ou do administrador serão fornecidos pela empresa, sendo desnecessária a preocupação com os mesmos. Pelo mesmo motivo excluem-se os testes de stress, de volume e de falha/recuperação por se considerar que o ambiente de implantação do sistema trata-se de um meio equilibrado em que essas situações não têm muito espaço para ocorrer, podem ser facilmente previstos e tratados pelo cliente, ou não pertencem à parte principal do sistema e foi deixado à cargo daquele.

Para a execução dos testes serão utilizadas máquinas o mais idênticas possível, em termos de hardware, àquelas que serão implantadas no hotel, a fim de garantir a previsibilidade de performance e compatibilidade.

Os testes mais críticos serão os testes de banco de dados, que compõe a maior parte do sistema e os de performance:

1. Testaremos os dados retornados
2. E também a correta atualização do banco de dados para as funções de cadastrar, atualizar e remover.
   1. **Identificação de Projeto**

A tabela abaixo identifica a documentação e disponibilidade usados para desenvolver o plano de testes:

| **Documento** | **Criado ou Disponível** | **Recebido ou Revisado** |
| --- | --- | --- |
| Especificação de Requisitos | ■ Sim □ Não | ■ Sim □ Não |
| Plano de Projeto | ■ Sim □ Não | ■ Sim □ Não |
| Modelo de Análise | ■ Sim □ Não | ■ Sim □ Não |
| Modelo de Projeto | ■ Sim □ Não | ■ Sim □ Não |
| Documento de Arquitetura | ■ Sim □ Não | ■ Sim □ Não |
| Protótipo | □ Sim ■ Não | □ Sim ■ Não |
| Manual do Usuário | □ Sim ■ Não | □ Sim ■ Não |
| Lista de Riscos | ■ Sim □ Não | ■ Sim □ Não |

**2. Requisitos a Testar**

A lista abaixo identifica aqueles itens – use cases, requisitos funcionais e não funcionais – que foram identificados como alvos de teste. Essa lista representa o que será testado.

**2.1 Teste Unitário**

* Verifique que os métodos que produzem as respostas dos endpoints estão se comunicando bem internamente e externamente, de modo que as respostas, sejam elas falhas ou sucessos, retornem corretamente para o usuário final
* Verificar todos os possíveis casos e respostas que o módulo da API retornará

**2.1 Análise Estática**

* Verifique se a ferramenta indique possíveis melhorias no código, como correção de bugs, code smells ou código duplicado

**2.2 Testes de integração**

* Verificar se as informações do usuário cadastradas no banco de dados podem ser consultadas corretamente por outros componentes do sistema.
* Verificar se as informações específicas de cada usuário são corretamente compartilhadas e atualizadas entre os diferentes módulos do sistema.
* Verificar se o cadastro, atualização e remoção de jogadores no banco de dados reflete corretamente nos outros componentes do sistema.
* Verificar se o sistema permite a criação, atualização e remoção de grupos de forma consistente com os dados armazenados no banco de dados.
* Verificar se as informações dos grupos são corretamente exibidas e atualizadas para os jogadores relevantes.
* Verificar se os dados cadastrados podem ser corretamente associados aos grupos e jogadores relevantes.

**2.3 Testes de Sistemas**

* Verificar se o sistema funciona corretamente como um todo, integrando todos os seus componentes e módulos.
* Verificar se o sistema atende aos requisitos funcionais e não funcionais especificados pelo cliente ou usuário final.
* Verificar se o sistema é capaz de lidar com diferentes cenários de uso, incluindo situações de erro, exceção ou falha

3. **Estratégia de Teste**

**3.1 Tipos de Teste**

**3.1.1 - Teste Unitário**

| Objetivo do Teste: | * Garantir a funcionalidade apropriada das funções testadas. * Garantir uma resposta apropriada ao usuário final caso alguma método da API falhe. |
| --- | --- |
| Técnica: | * Teste unitário usando maven |
| Critério de Finalização: | * Os testes planejados foram executados * Teste validado pela build do mvn test |
| Considerações Especiais: | Nenhuma |

***3.1.2 Análise Estática***

| Objetivo do Teste: | Verificar a qualidade do código-fonte e identificar possíveis problemas de segurança, desempenho, complexidade e duplicação. |
| --- | --- |
| Técnica: | Análise estática usando sonarcloud e CpdCheckVerify do java. |
| Critério de Finalização: | O código-fonte foi analisado pelo sonarcloud e pelo CpdCheckVerify  Não há problemas críticos ou bloqueadores reportados pelo sonarcloud  Há uma classe base para evitar duplicação do código |
| Considerações Especiais: | A análise estática do SonarCloud deve ser executada durante os pull requests no repositório e commits para a main |

***3.13 - Teste de Integração***

| Objetivo do Teste: | Verificar a comunicação e a interoperabilidade entre os componentes do sistema.  Verificar o comportamento esperado do sistema diante de cenários reais de uso. |
| --- | --- |
| Técnica: | Teste de integração usando Spring Boot Test. |
| Critério de Finalização: | Os testes planejados foram executados  Teste validado pela build do mvn test |
| Considerações Especiais: | Nenhuma |

***3.13 - Teste de Sistema***

| Objetivo do Teste: | Verificar se o sistema atende aos requisitos funcionais e não funcionais especificados.  Verificar se o sistema se integra corretamente com outros sistemas e componentes externos. |
| --- | --- |
| Técnica: | Teste de sistema usando Spring Boot Test |
| Critério de Finalização: | Os testes planejados foram executados  Teste validado pela build do mvn |
| Considerações Especiais: | O sistema deve estar configurado e implantado no ambiente de teste. |

**3.2 Ferramentas**

As seguintes ferramentas serão empregadas para esse projeto:

|  | **Ferramenta** |
| --- | --- |
| Projeto de Teste | Postman, Swagger, Maven, Sonar |
| Gerenciamento de Projeto | Github |
| Ferramentas do SGBD | PostgreSQL |

**4**. **Recursos**

Essa seção apresenta os recursos recomendados para o projeto do TIG, suas principais responsabilidades, e seus conhecimentos ou conjunto de habilidades.

**4.1 Trabalhadores**

Essa tabela mostra as suposições de recrutamento para o projeto.

| Recursos Humanos | | |
| --- | --- | --- |
| Trabalhador | Recursos Mínimos Recomendados | Responsabilidades Específicas ou Comentários |
| Gerente de Teste,  Gerente do Projeto de Teste | Dina  Franklin  Ruan  Lazaro  Patrick  Giovanna | Fornece supervisionamento gerencial.  Responsabilidades:   * provê direcionamento técnico * adquire recursos apropriados * fornece relatórios de gerenciamento |
| Test Designer | Dina  Franklin  Ruan  Lazaro  Patrick  Giovanna | Identifica, prioriza, e implementa os casos de teste.  Responsabilidades:   * gera o plano de teste * cria o modelo de teste * avalia a efetividade do esforço de teste |
| Testador | Ruan  Giovanna | Executa os testes.  Responsabilidades:   * executar os testes * registrar os resultados * reestabelecer-se dos erros * documentar solicitações de mudança |
| Administrador do Sistema de Teste | Giovanna | Garante que o ambiente e os bens de teste sejam gerenciados e mantidos.  Responsabilidades:   * administrar o sistema de gerenciamento teste * instalar e gerenciar o acesso do trabalhador ao sistema de testes |
| Gerente do Banco de Dados,  Administrador do Banco de Dados | Patrick | Garante que o ambiente e bens de teste de dados (banco de dados) sejam gerenciados e mantidos.  Responsabilidades:   * administrar os dados de teste (base de dados) |
| Designer | Giovanna | Identifica e define as operações, atributos, e associações das classes de teste.  Responsabilidades:   * identificar e definir as classes de teste * identificar e definir os pacotes de teste |
| Implementador | Giovanna | Implementa e faz os testes unitários das classes e pacotes de teste.  Responsabilidades:   * cria as classes e pacotes de teste implementados no modelo de teste |

**4.2 Sistema**

A tabela seguinte expõe os recursos do sistema para o projeto de teste.

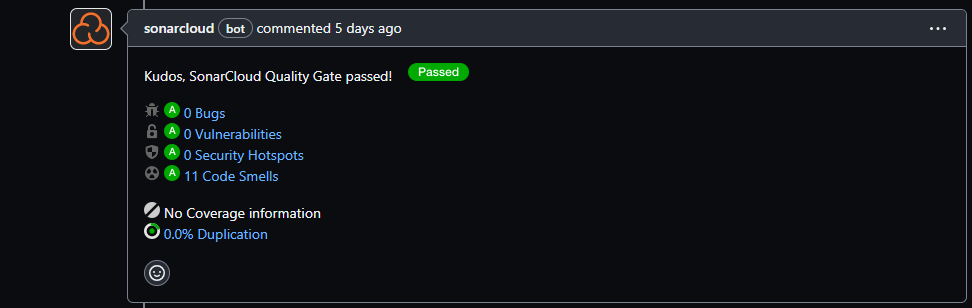
| Recursos do Sistema |
| --- |
| Servidor de Banco de Dados |
| — PostgreSQL |
| Terminais Clientes |
| —1 PC |
|  |
|  |
| Repositório de Testes |
| —1 PC |
| —3 PCs de Desenvolvimento de Teste |

**5. Cronograma**

| **Milestone** | **Data de Início** | **Data de Término** |
| --- | --- | --- |
|  |  |  |
| Planejar Teste | 06/06/23 | 06/07/23 |
| Projetar Teste | 20/06/23 | 07/07/23 |
| Implementar Teste | 20/06/23 | 08/07/23 |
| Executar Teste | 24/06/23 | 10/07/23 |
| Avaliar Teste | 11/06/23 | 11/07/23 |

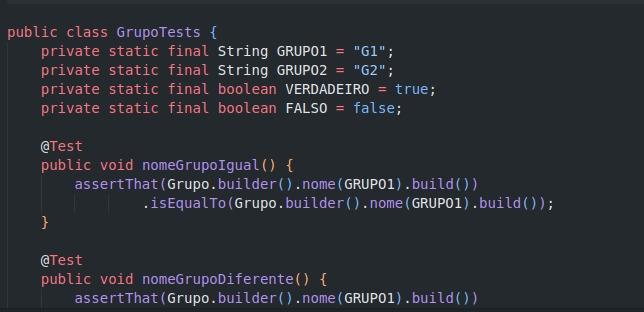
6. Resultados

**6.1 Análises Estáticas**

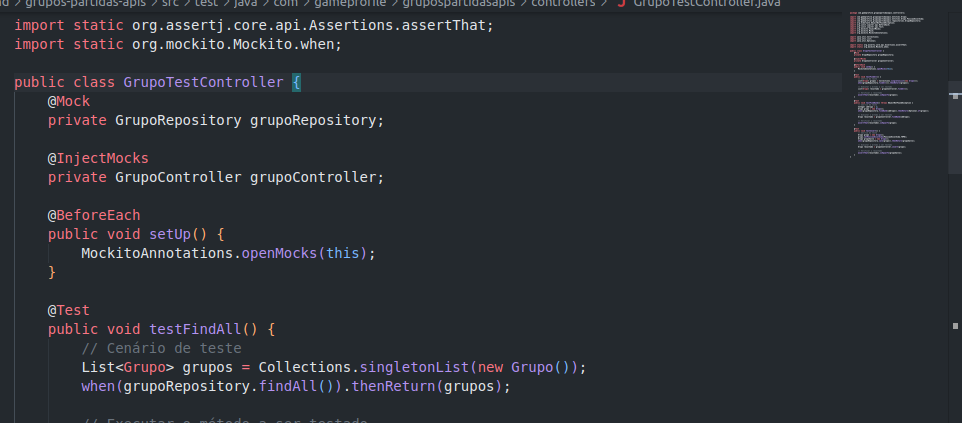
****

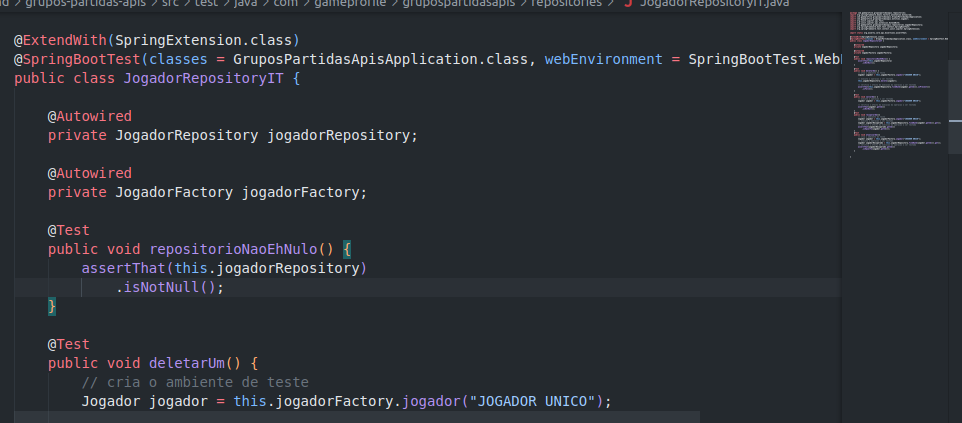
**6.2 Testes Unitários**

****

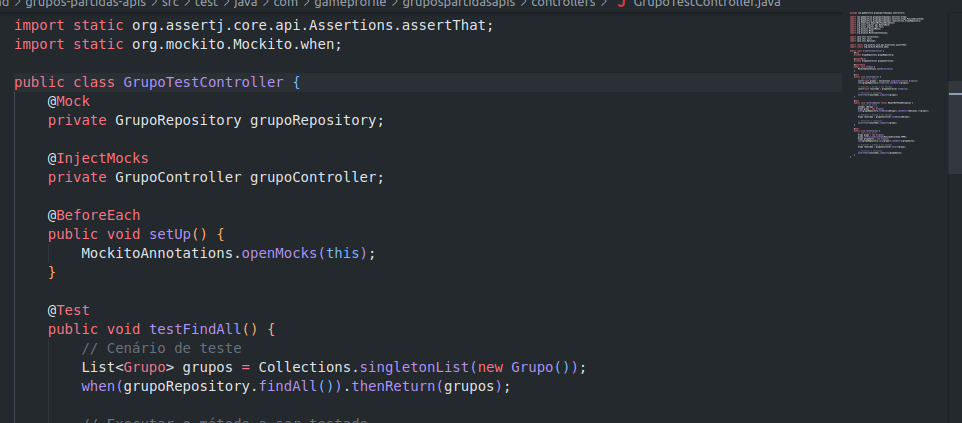
****

**6.3 Testes de Integração**



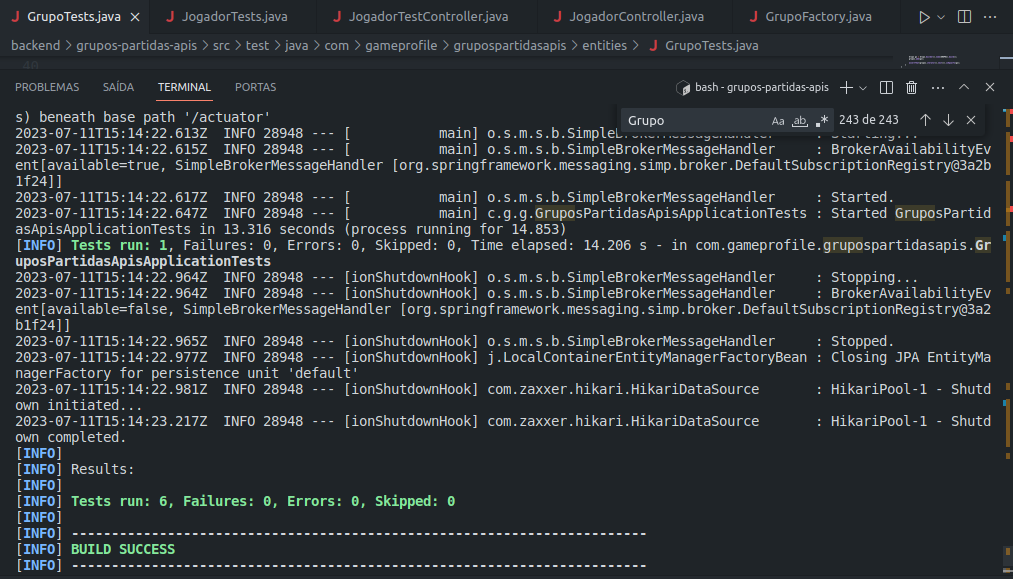


**6.4 Testes de Sistema**

****



**6.5 Testes rodando**

****