Universidade Federal de Ouro Preto

Engenharia de Software II

Sistema de *Loja de Jogos*

Grupo: *CDGMV*

Alunos:	Carlos Eduardo Gonzaga Romaniello
	Daniel Monteiro Valério
	Gabriel Mace dos Santos Ferreira
	Marcus Vinícius Souza Fernandes
	Vinicius Gabriel Angelozzi Verona de Resende
Professor:	Msc prof. Johnatan Oliveira
Horário:	Ter & Qui - 15:20 - 17:00

Ouro Preto, 02 de Dezembro de 2021

Conteúdo

1 Histórico de Revisões

Data	Versão	Descrição	Autor
01/11/2021	1.0	Início da escrita do documento	Grupo
12/11/2021	1.1	Especificação de Requisitos	Grupo
02/12/2021	1.2	Justificativa do processo de desenvolvimento e Levantamento de requisitos	Grupo
06/12/2021	1.3	Plano de VVT	Grupo
13/12/2021	1.4	Medição e Qualidade de Software + Observações	Grupo

Tabela 1: Revisões do Documento

2 Processo e Software

Para este projeto, o modelo de processo de software escolhido foi o SCRUM no qual o projeto deve ser dividido em pequenos ciclos de atividades, denominados *sprints*, sendo necessária a realização de reuniões periódicas com o intuito de planejar o projeto, analisar as ideias propostas e finalmente é feita uma averiguação dos resultados.

2.1 Justificativa do Processo de Desenvolvimento

O Scrum foi escolhido devido a agilidade que ele fornece ao desenvolvimento do projeto, dado que o período para entrega do produto é curto. Além disso, a flexibilidade do modelo ao permitir a adaptação deste segundo as necessidades e experiências passadas dos integrantes do grupo, tornou-o mais atrativo, pois ele independe de ferramentas específicas. Ademais, sua estrutura heurística permite que os membros da equipe aprendam a utilizar as tecnologias ao longo do processo de desenvolvimento, fator crítico ao projeto, visto que nem todos os membros da equipe dominam a linguagem e o framework utilizados, fator esse que acarretaria em um impacto negativo sobre o produto final em um processo de desenvolvimento mais rígido.

3 Cronograma

Nome	Tarefa	Prazo
CDGMV	Justificativa do processo de desenvolvimento escolhido	01/11 a 04/11
CDGMV	Especificação e Levantamento de requisitos	04/11 a 18/11
CDGMV	Plano de verificação, validação e teste de software	18/11 a 25/11
CDGMV	Planejar a medição e qualidade do software	18/11 a 25/11
CDGMV	Desenvolvimento do protótipo	25/11 a 28/11
CDGMV	Desenvolvimento do front-end e do back-end	25/11 a 13/12

Tabela 2: Cronograma

4 Levantamento de Requisitos

A técnica escolhida foi a prototipagem, devido a faciliade para identificar e validar os requisitos desejados pelo cliente. Sendo assim, foi desenvolvido um protótipo do projeto (disponível no figma) que foi apresentado com o intuito de que ele validasse as versões do projeto para que a equipe pudesse evoluí-lo de acordo com requisitos desejados.

5 Especificação de Requisitos

5.1 Requisitos Funcionais

RF01– O portal deve oferecer a busca de jogos no catálogo. **Informações:** Título do jogo. **Regras:** O cliente ou visitante tem permissão para executar uma pesquisa de jogo fornecendo as informações mencionadas. Caso haja resultado, devem ser listados os resultados da pesquisa, isso é, o(s) jogo(s), contendo foto, nome e descrição do(s) jogo(s). Caso não haja resultado de pesquisa, deve ser retornada uma mensagem informando que o título não faz parte do catálogo da loja.

- O portal deve validar se o título do jogo ou título similar está registrado na plataforma.
- O portal deve fornecer uma janela para inserção de data de nascimento caso o jogo possua uma faixa classificatoria indicativa para maiores de 16 anos. Após inserção do dado, deve ser feita a verificação de faixa etária.

RF02– O portal deve oferecer o aluguel de jogos. **Informações:** ID do jogo e ID do usuário. **Regras:** O cliente tem permissão para executar um aluguel de jogo ao clicar no botão de aluguel. Caso seja um visitante o sistema não permitirá a realização do aluguel e redirecionará o usuário para a tela de login. Caso o cliente possua uma conta será feito o aluguel após a validação dos dados pessoais da conta e o pagamento do valor do aluguel.

- O portal deve verificar se o usuário está de acordo com as condições de aluguel, ou seja, se o cliente pertence a faixa etária indicativa do jogo.

RF03– O portal deve permitir o login de usuários. **Informações:** Email e senha. **Regras:** O visitante tem permissão de fazer login no sistema. Caso este possua uma conta cadastrada o login será efetuado, caso contrário o sistema deverá informar que houve um erro.

- O portal deve validar se o email e senha pertencem a uma mesma conta cadastrada.

RF04– O portal deve permitir o cadastro de usuários. **Informações:** Nome completo, email, senha, data de nascimento e CPF. **Regras:** O email e CPF não devem pertencer a uma conta existente. No caso de um administrador adicionar uma conta, deverá disponibilizar a opção de transformar o novo usuário em administrador. Caso seja um visitante, deve ser disponibilizado o mesmo formulário do administrador, mas sem a opção mencionada. Caso o suposto usuário exista, deverá ser fornecida uma mensagem de notificação.

- O portal deve validar quem está realizando o cadastro, ou seja, caso o usuário que esteja conectado seja administrador, a operação não deverá desconectar o usuário de sua conta.

- O portal deve validar quem está realizando o cadastro, ou seja, caso o usuário que esteja conectado seja administrador, a operação não deverá desconectar o usuário de sua conta.

RF05– O portal deve permitir o acesso as informações dos jogos. **Informações:** ID do jogo desejado. **Regras:** O ID do jogo deve ser válido e deve estar cadastrado corretamente. Para acessar jogos com faixa etária restrita, o usuário deve confirmar que está ciente de que está acessando uma página com conteúdo para a faixa definida.

- O portal deve verificar o ID fornecido para poder abrir a nova página com todas as informações do jogo indicado.

- O portal deve verificar a idade do usuário para poder abrir a nova página e caso a idade não seja suficiente, o portal deverá pedir a confirmação do usuário.

RF06– O portal deve permitir alterações no perfil do usuário. **Informações:** Os dados que se deseja alterar. **Regras:** O portal deve verificar se os novos dados ainda estão no mesmo domínio dos dados antigos.

- O portal deve verificar se os dados atualizados estão corretos como por exemplo verificar se a data de nascimento alterada continua sendo uma data válida.

RF07– O portal deve permitir que o administrador realize o CRUD (Create, Read, Update e Delete) dos jogos. **Informações:** Para novos cadastros deve-se informar todos os dados do novo jogo, para o update de um jogo existente deverão ser fornecidos os dados que se deseja alterar e para as outras operações deve-se fornecer o id correspondente. **Regras:** O portal deverá verificar se os dados novos ou atualizados estão corretos.

- O portal deve verificar no momento do cadastro se aquele jogo já existe. Em caso positivo deve-se informar ao administrador sobre a existência desse jogo (CREATE);

- No caso do delete o portal não poderá excluir um jogo caso ele esteja alugado naquele momento (DELETE);

- Na operação read deve-se verificar se o ID fornecido é valido (READ);

- Na operação de update, o portal deve verificar o domínio dos novos dados para ver se eles estão corretos (UPDATE).

RF08– O portal deve permitir que o administrador realize o CRUD (Create, Read, Update e Delete) dos usuários. **Informações:** Para novos cadastros deve-se informar todos os dados do usuário, para a edição de um usuário existente deverão ser fornecidos os dados que se deseja alterar e para as outras operações deve-se fornecer o CPF correspondente. **Regras:** O portal deverá verificar se os dados novos ou atualizados estão corretos.

- O portal deve verificar no momento do cadastro se aquele usuário já existe. Em caso positivo deve-se informar ao administrador sobre a existência desse usuário (CREATE);

- No caso do delete, apenas o administrador inicial pode excluir outros administradores (DELETE);

- Na operação de read deve-se verificar se o CPF fornecido está cadastrado (READ);

- Na operação de atualização, o portal deve verificar se o domínio dos novos dados estão corretos (UPDATE).

RF09– O portal deve permitir ao administrador visualizar o histórico de aluguéis gerais ou de um usuário específico. **Informações:** O ID do usuário, caso ele queira visualizar os aluguéis específicos de um usuário. **Regras:** O portal deve mostrar os usuários em ordem dos mais recentes aos mais antigos e deve verificar se o ID do usuário fornecido é válido.

- O portal deve fazer a verificação do ID do usuário para poder buscar o seu histórico de aluguéis.

RF10– O portal deve oferecer a busca de jogos no histórico de aluguel do usuário. **Informações:** Título do jogo. **Regras:** O cliente ou administrador tem permissão para executar uma pesquisa de jogos no histórico fornecendo as informações mencionadas. Caso haja resultado, devem ser listados os resultados da pesquisa, isso é, os históricos de alugueis com o(s) jogo(s) definido(s), contendo usuário, data de aluguel e nome do(s) jogo(s). Caso não haja resultado de pesquisa, deve ser retornada uma mensagem informando que o título não possui histórico registrado.

- O portal deve validar se o título do jogo ou título similar está registrado no histórico de aluguéis.

5.2 Requisitos Não Funcionais

RNF01. Efetivação de login. **Informações:** usuário e senha. **Regras:** O cliente terá acesso para alugar um jogo, acessar e alterar suas informações, excluir conta, acessar e consultar o histórico de transações relativos à mesma conta.

RNF02. Tratamento de dados de entrada. **Regras:** A plataforma deverá ser capaz de devidamente tratar os dados de entrada escritos por cada usuário de forma a impossibilitar a inserção de comandos, pois estes resultariam na existência de vulnerabilidades (XSS, por xemplo).

RNF03. Efetivação de cadastro. **Informações:** Nome, data de nascimento, CPF, email e senha. **Regras:** A plataforma deverá validar informações de email e CPF. O email não poderá ser repetido de uma outra conta existente, assim como o CPF. Além disso, deverá ser validado o CPF inserido de acordo com as regras definidas pela Receita Federal do Brasil.

RNF04. Efetivação de pagamento. **Informações:** Dados de pagamento. **Regras:** A plataforma deverá utilizar uma API externa para aprovação de pagamentos realizados durante o aluguel. Recomenda-se a utilização do *Mercado Pago*.

5.3 Diagrama de Casos de Uso

Abaixo encontra-se o diagrama de casos de uso para cada um dos atores do sistema: Visitante, Usuário e Administrador.

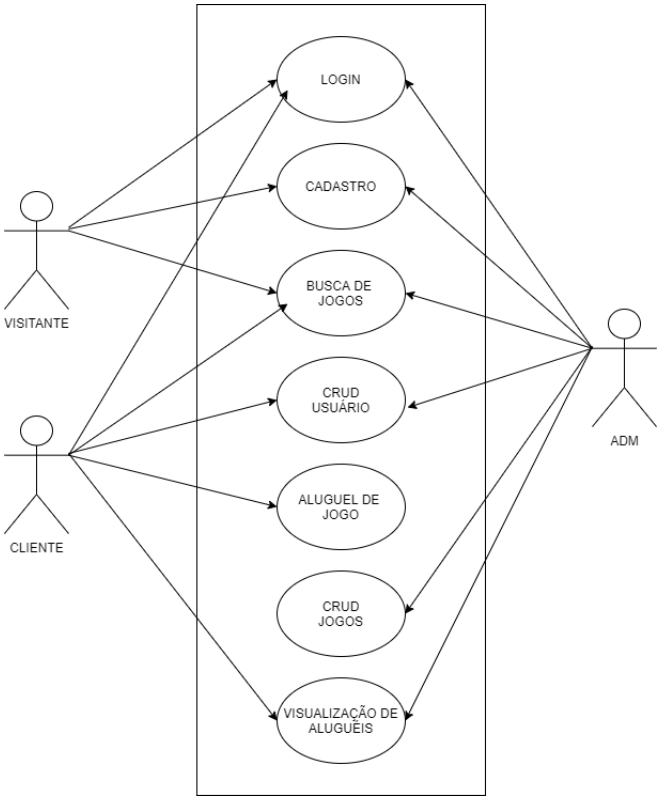

Figura 1: Diagrama de Casos de Uso

- Visitante: Caso o usuário não esteja devidamente logado ele terá permissão para realizar o login, se cadastrar e buscar por jogos no catálogo da loja;
- Usuário: Caso o usuário esteja devidamente logado, ele poderá realizar um novo login, buscar por jogos, fazer um CRUD em seu perfil, efetuar o aluguel de jogos e visualizar seu histórico de aluguéis;

- Administrador: Caso o usuário esteja devidamente logado e seja um administrador ele terá permissão para cadastrar, pesquisar, editar e deletar de jogos e outros usuários, sendo possível decidir se o novo usuário será um cliente ou admnistrador. Além disso, é permitida a visualização do histórico de aluguéis a partir do mais recente ao mais antigo;

6 Plano de VVT

6.1 Requisitos a serem testados

Esta seção descreve em linhas gerais o conjunto de requisitos a serem testados no projeto a ser desenvolvido, comunicando o que deve ser verificado. Exemplos de requisitos a serem testados são: desempenho, segurança, interface de usuário, controle de acesso, funcionalidades.

Os requisitos do software que que serão testados são:

1. Segurança:
    - Validação de acesso via JWT: A validação apenas ocorre em sessões ou locais que necessitam de um controle de acesso, em outros casos essa validação é desconsiderada.
2. Interface de Usuário:
    - Usabilidade: A medição da qualidade da interface será dada através do feedback de usuários selecionados para checarem o fluxo de navegação, affordance do layout e quão intuitivas e claras estão as informações disponíveis na plataforma.
3. Controle de acesso:
    - Exclusividade no acesso à telas especificas: A validação ocorre através da verificação de acesso de telas exclusivas para um tipo de usuário especifico (usuário logado e administrador).
4. Funcionalidades:
    - Banco de dados: A validação das operações de CRUD (Create, Read, Update, Delete) do banco de usuários, de jogos e aluguéis/históricos ocorrerá através de testes unitários.

- Endpoints da API: Testes unitários seram realizados sobre cada um dos endpoints presentes na API (Application Programming Interface), para cada um dos métodos de requisição HTTP, sejam eles GET, POST, PUT ou DELETE.

6.2 Estratégias e ferramentas de teste

Testes de segurança, interface, controle de acesso, funcionalidades do banco de dados e endpoints da API serão realizados. No que tange a interface de usuário, um questionário será aplicado para um grupo específico de pessoas após uma série de tarefas a serem efetuadas na plataforma, visando avaliar a usabilidade. A conclusão deste estudo será feita ao obter resultados considerados aceitáveis pela medição de qualidade empregada.

Em relação aos demais pontos, o framework Jest será utilizado para efetuar testes unitários, com o intuito de assegurar a qualidade e bom funcionamento do software em geral. Nesse sentido, a finalização destes testes será feita de forma individual de acordo com os resultados obtidos.

6.3 Equipe e infra-estrutura

No contexto deste projeto, a equipe responsável pelo desenvolvimento contém um interesse *fullstack*, visando participar em todas as vertentes do projeto inclusive na projeção do teste de interface e na confecção dos testes unitários. Já os responsáveis pelos testes de interface, serão um grupo de usuários previamente selecionados para realizar uma série de atividades na plataforma e por fim avaliá-la através de um questionário. Os equipamentos envolvidos durante esta etapa serão os próprios hardwares dos desenvolvedores e convidados para testes. O material de apoio utilizado será a documentação presente nos sites dos frameworks *React, Node* e *Jest*.

6.4 Execução do Plano de Teste

A seguir são mostradas as tarefas relacionadas ao teste do protótipo da plataforma da locadora de jogos ***Loka Games***:

- Planejar Teste:
    1. Identificar Requisitos para o Teste;
    2. Desenvolver Estratégia de Teste;
    3. Criar Planejamento;
    4. Gerar Plano de Teste.

- Projetar Teste:
    1. Desenvolver Conjunto de Teste;
    2. Identificar e Descrever Casos de Teste;
    3. Identificar e Estruturar Scripts de Teste;
    4. Revisar e Acessar Cobertura de Teste.
- Implementar Teste:
    1. Configurar Ambiente de Teste;
    2. Registrar ou Programar Scripts de Teste;
- Executar Teste:
    1. Executar Script de Teste;
    2. Avaliar Execução do Teste;
    3. Verificar os Resultados;
    4. Investigar Resultados Inesperados;
    5. Registrar Defeitos.
- Avaliar Teste:
    1. Avaliar a Cobertura dos Casos de Teste;
    2. Avaliar Cobertura do Código;
    3. Analisar Defeitos;
    4. Determinar se os Critérios de Conclusão do Teste e os Critérios de Êxito foram alcançados;
    5. Criar Relatório de Avaliação do Teste.

7 Medição e Qualidade de Software

No que tange a medição e qualidade do software desenvolvido neste trabalho, a ferramenta que mais demonstrou completude para atender a demanda necessária na *API* (Application Programming Interface) e no *Client* foi a JSNose, uma vez que ela fornece uma ánalise de code smells, acoplamento para JS, HTML, CSS dentre outros utilitários.

Portanto, faz-se necessário a instalação desta dependência nos respectivos repositórios das frentes do projeto para que a mesma seja executada. Devido

ao curto tempo hábil para desenvolvimento, até o momento a ferramenta ainda não foi executada, porém alguns *code smells* foram detectados pelo time, são eles:

- **Comment:** Falta de comentários chaves ao longo do código, tanto na API como na aplicação, dificultando a compreenção da aplicação em iterações futuras;
- **Shotgun Surgery & Duplicated Code:** Falta de componentização de partes da aplicação que resulta em vários trechos idênticos replicados;
- **Duplicated Code:** Presença de códigos desnecessáriamente repetidos resultantes na sobreposição de papéis.
- **Data Class:** Presença de objetos apenas para armazenar dados estáticos a serem consumidos pela aplicação.

8 Observações

Ao longo do curso houveram dificuldades quanto ao balanceamento entre o desenvolvimento da aplicação idealizada e a documentação exigida, dado que o volume de tópicos a serem escritos em conjunto com a necessidade de planejar, desenvolver e avaliar o código para a aplicação, tornaram a conclusão do trabalho prático custosa em termos de tempo e funcionalidades. Mesmo não havendo a necessidade de concluir a aplicação, o esforço empreendido ao longo do período foi com o intuito de finalizar o projeto proposto, ocasionando na situação mencionada anteriormente.

9 Referências

[1] Chapman, S.J. – Electric Machinery Fundamentals, 4th Edition;
[2] Fitzgerald, A. E. – Máquinas Elétricas, 2da Edição;