Loja Virtual de Jogos de Computador

Fernando Martins, Luigi Boscatto, Pedro Vargas Tannuri

[1]Departamento de Ciência da Computação
Universidade do Estado de Santa Catarina (UDESC) – Joinville, SC – Brasil

1. Descrição do Problema

Considerando o aumento na popularidade de jogos eletrônicos, sobretudo para computadores, muitos jogadores anseiam por uma plataforma de compra e venda de jogos, bem como um local para organizar comunidades com interesses em comum.

Por outra perspectiva, podemos ainda mencionar o interesse que os desenvolvedores têm em publicar seus produtos de forma rápida e barata, que funcione como uma vitrine virtual.

1.1. Escopo do Software

- Catálogo de jogos: Uma lista completa de todos os jogos disponíveis para compra, com informações relevantes, como título, descrição, preço, desenvolvedor e nota (de acordo com as avaliações dos usuários);

- Conta do usuário: Os usuários poderão se cadastrar com um email e uma senha. Suas contas deverão conter também um nome (nickname), telefone, endereço, lista de amigos, e uma lista de grupos da comunidade que os usuários podem formar. A fim de adquirir jogos, a conta deve possuir um carrinho de compras, que será populado por jogos. Após a compra, os jogos deverão ser listados numa biblioteca, vinculada à conta;

- Área da comunidade: Dentro da plataforma, os usuários terão acesso a um espaço dedicado a troca de experiências e interesses entre jogadores. Nele, será possível criar e participar de inúmeros grupos, postar resenhas sobre os jogos, abrir discussões e publicar modificações dos jogos (mods).

1.2. Listagem de Stakeholders

- Desenvolvedores de jogos: São as empresas ou indivíduos que criam os jogos disponíveis na loja. Eles têm interesse em uma plataforma que promova a visibilidade e venda de seus jogos, além de fornecer informações e suporte aos usuários;

- Proprietários da plataforma: São os donos ou gestores da loja de jogos online. Eles têm interesse em ter um software eficiente que promova a venda e distribuição dos jogos digitais de forma lucrativa, bem como disseminar a cultura gamer;

- Equipe de desenvolvimento: São os profissionais responsáveis pelo desenvolvimento, manutenção e atualização do software da loja. Eles devem entender as necessidades e requisitos dos proprietários da loja, desenvolvedores de jogos e usuários finais para construir um software que atenda essas expectativas;

- Usuários finais: São os clientes da loja de jogos online. Eles desejam uma experiência simples, conveniente e segura de compra e acesso aos jogos, além de criar um ciclo de interação social com outras pessoas que compartilham seu interesse por jogos e desenvolvimento de mods;

2. Requisitos funcionais e não-funcionais do projeto

O sistema deverá seguir os requisitos e apresentar as funcionalidades listadas abaixo.

2.1. Requisitos funcionais

- Cadastro de usuário: Permitir que os usuários criem contas na plataforma;
- Navegação no Catálogo: Fornecer aos usuários a capacidade de explorar o catálogo de jogos, filtrar por categorias, gêneros, avaliações e desenvolvedores;
- Visualização de Informações de Jogos: Exibir detalhes dos jogos, incluindo título, descrição, imagens, requisitos mínimos do sistema, classificação etária e preço;
- Adição ao Carrinho de Compras: Permitir que os usuários adicionem jogos ao carrinho de compras enquanto navegam no catálogo;
- Finalização de Compra: Oferecer um processo de checkout seguro, onde os usuários possam revisar o carrinho de compras, selecionar opções de pagamento e concluir a compra;
- Download de Jogos: Permitir que os usuários baixem os jogos adquiridos de forma segura e confiável;
- Avaliações e Comentários: Permitir que os usuários avaliem e comentem sobre os jogos adquiridos, fornecendo feedback para os desenvolvedores.

2.2. Requisitos Não Funcionais

- Linguagem: O sistema deverá ser implementado na linguagem Java;
- Segurança: Garantir a segurança dos dados pessoais e financeiros dos usuários, com medidas de proteção e criptografia adequadas;
- Desempenho: Garantir um desempenho rápido e eficiente do sistema, permitindo que os usuários naveguem pelo catálogo, façam compras e baixem jogos sem atrasos significativos;
- Confiabilidade: Assegurar que o sistema esteja disponível de forma confiável e que as transações monetárias sejam processadas corretamente, evitando erros;
- Usabilidade: Criar uma interface amigável e intuitiva, facilitando a navegação, busca e compra de jogos para qualquer usuário da plataforma;
- Escalabilidade: Projetar o sistema para lidar com um grande número de usuários simultâneos e suportar o crescimento futuro do catálogo de jogos e das comunidades;

- Internacionalização: Permitir a localização do sistema em diferentes idiomas e moedas, para atender a usuários de diferentes regiões e culturas;

- Integração com Sistemas de Pagamento: Integrar o sistema com provedores de pagamento confiáveis para permitir transações seguras e diversas opções de pagamento;

- Backup e Recuperação: Implementar rotinas regulares de backup dos dados do sistema e garantir a capacidade de recuperar o sistema em caso de falhas ou desastres;

- Acessibilidade: Garantir que o sistema seja acessível para usuários com necessidades especiais, cumprindo os padrões de acessibilidade e oferecendo recursos de suporte adequados.

3. Estimativa de duração do projeto completo

A fim de calcular uma estimativa para o tempo de duração do projeto em questão, deve-se analisar o tamanho do código, os tipos de entradas e saídas, complexidade das atividades propostas, entre outros. Para isso, utilizaremos o método COCOMO.

Iniciaremos contando e categorizando as funcionalidades do software. Em seguida, deve-se definir a complexidade de implantação de cada uma.

- Número de Entradas Externas (EE):
    1. Cadastro de novos jogos (baixo);
    2. Cadastro de empresas (baixo);
    3. Cadastro de usuários (baixo);
    4. Publicações de posts nas comunidades (média);
    5. Publicação de avaliações dos jogos (média);
    6. Alterações nos dados de usuários (média);
    7. Alterações nos dados de empresas (média);
    8. Alterações nos dados dos jogos (média).
- Número de Saídas Externas (SE):
    1. Relatório de vendas dos jogos (média);
    2. Relatório de avaliação dos jogos pelos usuários (média).
- Número de Consultas Externas (CE):
    1. Busca de jogos por categorias (fácil);
    2. Busca de jogos por nome (fácil);
    3. Busca de empresas desenvolvedoras de jogos (fácil);
    4. Busca de perfis de usuários (fácil);
    5. Busca de comunidades de interesse (fácil).
- Número de Arquivos Lógicos Internos (ALI):
    1. Arquivo para tabela de usuários (média);
    2. Arquivo para tabela de empresas (média);
    3. Arquivo para tabela de jogos (média);

4. Arquivo para tabela de avaliações de jogos (média);
5. Arquivo para tabela de comunidades (média);
6. Mensagem de erro na compra de jogos (fácil);
7. Mensagem de erro no download ou instalação (fácil);
8. Mensagem de login inválido (fácil);
9. Mensagem de erro de cadastro de novo usuário (fácil);
10. Mensagem de confirmação de login (fácil);
11. Mensagem de confirmação de avaliação enviada (fácil);
12. Mensagem de confirmação de compra (fácil);
13. Mensagem de confirmação de cadastro (fácil).

- Número de Arquivos de Interface Externos (AIE):

1. Interface externa para sistema de compra dos jogos (média).

Na sequência, encontraremos o valor de Pontos de Função não Ajustados (PFNA). Para isso, utilizamos uma tabela de pesos (Figura 1) para cada elemento conforme sua complexidade, e calculamos PFNA através da fórmula:

$$PFNA = \sum elemento \cdot Peso$$

Elemento\Complexidade	Baixa	Média	Alta
Entradas Externas (EE)	3	4	6
Saídas Externas (SE)	4	5	7
Consultas Externas (CE)	3	4	6
Arquivos Lógicos Internos (ALI)	7	10	15
Arquivos de Interface Externos (AIE)	5	7	10

Figure 1. Tabela de pesos conforme complexidade da funcionalidade

Fazendo os cálculos, de acordo com as funcionalidades definidas e seus pesos:

$$EE = 3 \cdot 3 + 5 \cdot 4 = 29$$

$$SE = 2 \cdot 5 = 10$$

$$CE = 5 \cdot 3 = 15$$

$$ALI = 8 \cdot 7 + 5 \cdot 10 = 106$$

$$AIE = 1 \cdot 7 = 7$$

A soma de todos os pesos resulta em:

$$PFNA = \sum EE + \sum SE + \sum CE + \sum ALI + \sum AIE = 169$$

Com PFNA definido, podemos obter o valor de LOC (Lines of Code). Com o programa escrito em Java, cada PFNA equivale a 53 linhas de código, então obtemos o seguinte:

$$LOC = 169 \cdot 53 = 8957$$

Convertendo isso para KLOCS, temos 8,957 KLOCS. A partir desse valor, podemos, em fim, calcular o Esforço - medido em pessoas-mês - e a Duração - medida em

meses - estimados do projeto. Vamos considerar que esse é um projeto de complexidade média.

$$Esforço = 3,0 \cdot KLOC^{1,12}$$

$$Esforço = 3,0 \cdot 8,957^{1,12}$$

$$Esforço = 35pm$$

$$Duraçao = 2,5 \cdot Esforço^{0,35}$$

$$Duraçao = 2,5 \cdot 35^{0,35}$$

$$Duraçao = 9meses$$

Através da Estimativa Paramétrica do método COCOMO, chegamos a conclusão que esse projeto teria uma Duração de 9 meses, com um Esforço de 35 pessoas-mês.

4. Diagrama de Classes do Projeto UML

A seguir, temos o Diagrama de Classes (Figura 2) do projeto em questão. Caso haja dificuldade de visualização neste documento, a imagem também está disponível no repositório do Github do projeto.

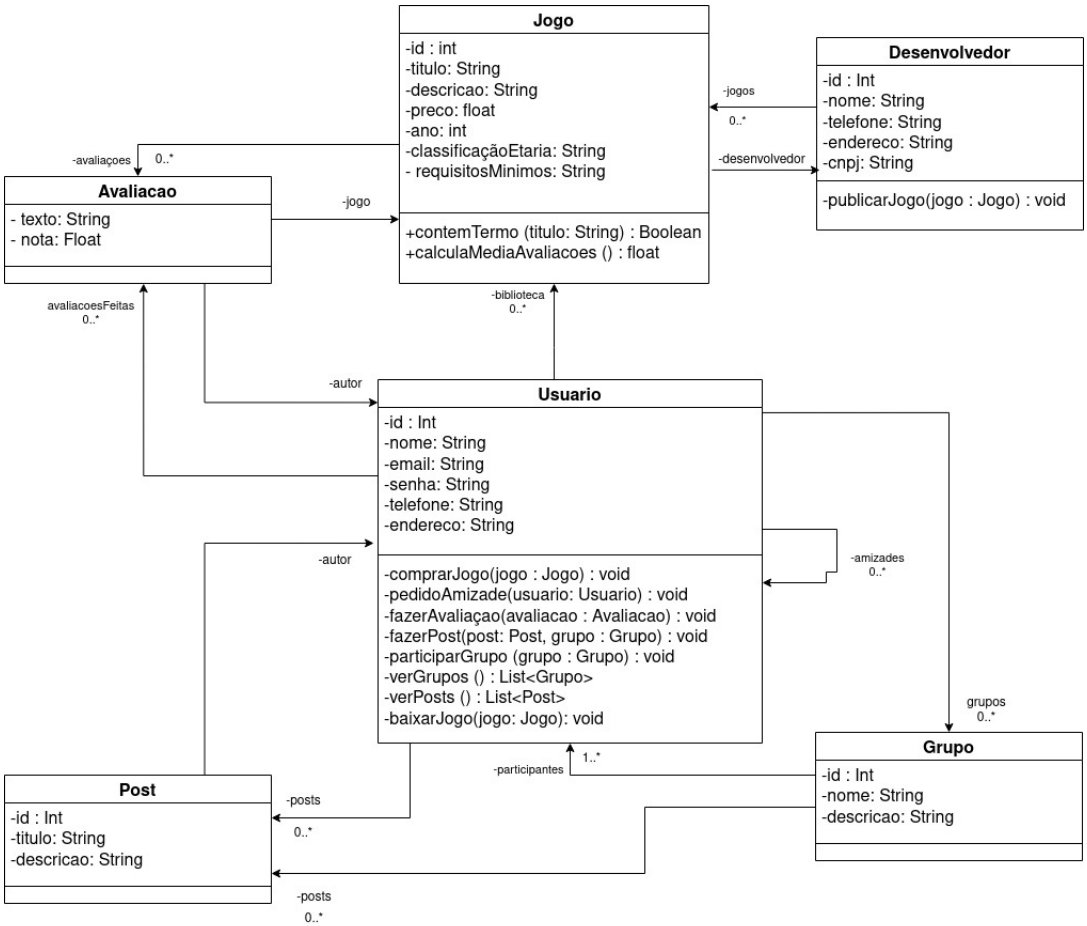

Figure 2. Diagrama de classes UML representando estrutura do sistema

5. Testes Unitários

Para validar o funcionamento correto do código e das funcionalidades implantadas, fizemos o uso de Testes Unitários, que estão disponíveis na pasta Test do repositório do Github do projeto.

Visto que o código que criamos é apenas um ”esqueleto” do que seria o projeto concluído, os testes cobrem as partes mais críticas de cada classe. Como as classes Avaliacao e Post não tem métodos crítico, não há arquivo de testes unitários para elas.

Na classe Jogo, precisamos garantir que suas Avaliaçoes estão sendo registradas, e que a nota média é calculada corretamente. Na Classe Grupo, os testes garantem que os usuários e os posts estão sendo registrados corretamente no grupo. A classe Desenvolvedor, por sua vez, garante que os seus jogos estão sendo publicados.

Por fim, a classe mais crítica é a de Usuario, pois seus métodos interagem com todas as outras do código. Visto isso, criamos um teste para cada funcionalidade implantada.