Projeto Inscryption

Especificação de requisitos de Software

versão 1.4

06/03/2025

Versão	Autores	Data	Ação
1.0	Augusto Schweitzer, Guilherme da Silva Pierri e Rafael dos Santos Oliveira	29/09/2024	Estabelecimento dos requisitos
1.1	Augusto Schweitzer, Guilherme da Silva Pierri e Rafael dos Santos Oliveira	08/10/2024	Modificações nos requisitos não funcionais e adição de casos de uso
1.2	Augusto Schweitzer, Guilherme da Silva Pierri e Rafael dos	14/11/2024	Adição de novos requisitos funcionais

	Santos Oliveira		

1.3	Augusto Schweitzer, Guilherme da Silva Pierri e Rafael dos Santos Oliveira	26/11/2024	Mudança de requisitos funcionais
1.4	Augusto Schweitzer, Guilherme da Silva Pierri e Rafael dos Santos Oliveira	06/03/2025	Adição de requisitos funcionais

Conteúdo:

1.Introdução

1.1 Objetivo

Desenvolvimento de um programa distribuído que permita a realização de partidas do jogo Inscryption na modalidade jogador contra jogador.

1.2 Definições, abreviaturas

Regras do jogo: ver apêndice

1.3 Referências:

Apresentação das regras do jogo (regras do jogo na seção Gameplay):
https://en.wikipedia.org/wiki/Inscryption

2. Visão geral

2.1 Arquitetura do programa

Cliente-servidor distribuído.

2.2 Premissas de desenvolvimento

- O programa deve ser implementado em Python;
- O programa deve usar DOG como suporte para execução distribuída;
- Além do código, deve ser produzida especificação de projeto baseada em UML, segunda versão.

3. Requisitos de Software

3.1 Requisitos Funcionais:

Requisito funcional 1 - Start game:

Ao ser executado, o programa deve solicitar conexão com DOG Server (utilizando os recursos de DOG). O resultado da tentativa de conexão deve ser informado ao usuário. Apenas em caso de conexão bem

sucedida as funcionalidades estarão habilitadas. No caso de conexão mal sucedida, a única alternativa deve ser encerrar o programa. Após isso, apresentar na interface a tela principal do jogo, que apresentará as opções: Iniciar partida, Criar deck e Sair do jogo.

Requisito funcional 2 – Start match:

O programa deve apresentar a opção de menu “iniciar partida” para o início de uma nova partida. O procedimento de início de partida consiste em enviar uma solicitação de início a Dog Server, que retornará o resultado, que será a identificação e a ordem dos jogadores, em caso de êxito, ou a razão da impossibilidade de início de partida, caso contrário.
A interface do programa deve ser atualizada com as informações recebidas e caso o jogador local seja quem inicia a partida, a interface deve estar habilitada para seu procedimento de jogada. Esta funcionalidade só deve estar habilitada se o programa estiver em seu estado inicial, isto é, sem partida em andamento e com a mesa em seu estado inicial;

Requisito funcional 3 - Buy deck card

O programa deve possibilitar ao jogador a ação de comprar uma carta do baralho, por vez, para a sua mão, posicionando o cursor em cima do botão “Comprar carta do deck”, adicionando uma carta do seu deck à sua mão. Se a ação for realizada após o jogador já ter comprado uma carta, deve ser notificado um lance irregular.

Requisito funcional 4 - Buy squirrel card

O programa deve possibilitar ao jogador a ação de comprar uma carta esquilo, por vez, para a sua mão, posicionando o cursor em cima do botão “Comprar esquilo”, adicionando uma carta esquilo à sua mão. Se a ação for realizada após o jogador já ter comprado uma carta, deve ser notificado um lance irregular.

Requisito funcional 5 - Receive move

O programa deve poder receber uma jogada do adversário, enviada por Dog Server, quando for a vez do adversário do jogador local. A jogada recebida deve ser um lance regular e conter as informações necessárias para atualização da interface. Após isso, deve-se avaliar o encerramento de partida. No caso de encerramento de partida, deve ser notificado que houve um vencedor; no caso de não encerramento, deve ser habilitado o jogador local, para que possa proceder a seu lance;

Requisito funcional 6 - Receive start

O programa deve poder receber uma notificação de início de partida, originada em Dog Server, em função de solicitação de início de partida por parte de outro jogador conectado ao servidor. O procedimento a partir do recebimento da notificação de início é o mesmo descrito no 'Requisito funcional 2 – *Start match*, isto é, a interface do programa deve ser atualizada com as informações recebidas e caso o jogador local seja quem inicia a partida, a interface deve estar habilitada para seu procedimento de lance.

Requisito funcional 7 - Receive withdrawal notification

O programa deve poder receber uma notificação de abandono de partida por parte do adversário remoto, enviada por Dog Server. Neste caso, a partida deve ser considerada encerrada e o abandono notificado na interface.

Requisito funcional 8 - Pass turn

O programa deve possibilitar ao jogador a opção de avisar ao sistema que terminou as suas ações naquela rodada, clicando no botão "Passar turno" mostrado na interface. Caso o jogador realize uma tentativa de passar o turno sem ter comprado uma carta, será notificado um lance irregular, visto que para passar o turno o jogador deve ter comprado obrigatoriamente uma carta, podendo ou não invocar uma carta na mesa.

Requisito funcional 9 - Save deck

O sistema fornece ao jogador a opção de customizar o seu deck, ou seja, escolher as cartas que ele desejar que estejam em seu baralho. Esta opção é visualizada na página principal do jogo, sendo acessada com um clique “Criar deck”. Entretanto, as cartas que o jogador iniciará na partida e como elas estarão apresentadas no baralho de compra será de forma aleatória, isto é, as cartas não estarão na ordem como o jogador quiser, pois estarão embaralhadas. Após personalizar o seu deck (utilizando os Requisitos funcionais 11 e 12) o jogador fará uso deste requisito funcional, para salvar as mudanças feitas no seu deck.

Requisito funcional 10 - Select position

O programa deve permitir ao usuário que ele escolha uma posição que seja válida para invocar suas cartas, sendo esta posição contendo uma carta (que será sacrificada) ou não, analisando detalhadamente cada ocasião. Após selecionar uma posição, o jogador pode selecionar uma carta, ou realizar a invocação de uma carta na mesa, sendo devidamente tratado para cada ocasião. Sendo assim, se o jogador escolher uma posição em há uma carta, o Requisito Funcional 13 - *Select card* - será chamado. Por outro lado, se a posição selecionada estiver vazia (sem carta) será chamado o Requisito Funcional 14 - *Invoke card*.

Requisito funcional 11 - Add card to deck

A aplicação permite ao usuário a inserção de novas cartas ao seu deck personalizado. Este requisito funcional modifica a interface do jogador e altera a lista de cartas apresentadas ao mesmo. Caso o jogador salve o deck (botão), então o requisito funcional salvar deck entra em ação.

Requisito funcional 12 - Remove card from deck

A aplicação permite ao usuário a remoção de cartas pertencentes ao seu deck personalizado. Este requisito funcional modifica a interface do jogador e altera a lista de cartas apresentadas ao mesmo. Caso o jogador salve o deck (botão), então o requisito funcional salvar deck entra em ação.

Requisito funcional 13 - Invoke card

O programa deve permitir que o usuário invoque uma carta no campo, desde que ela tenha sido previamente selecionada conforme o Requisito Funcional 10 – *Select position*. Ao realizar a invocação, a interface do jogador local será atualizada para exibir a nova carta no campo, e ela será removida da mão do jogador. Caso o jogador tente invocar uma carta sem ter comprado uma previamente, será exibida uma notificação informando que a ação é inválida, pois a compra de uma carta é obrigatória antes da invocação.

Requisito funcional 14 - Select card

O programa deve permitir que o usuário selecione uma carta para invocação, desde que a posição tenha sido previamente definida conforme o Requisito Funcional 10 – *Select position*. Ao executar *Select card*, a carta escolhida da mão do jogador será movida para o campo na posição selecionada, enquanto a carta designada como sacrifício será removida do campo.

Requisito funcional 15 - Leave game

A aplicação deve permitir que o usuário saia do jogo por meio do botão “Sair do jogo”, apresentado na página inicial do jogo. Ao executar essa ação, o jogo será fechado, e a conexão com o *Dog Server* será encerrada.

3.2 Requisitos Não Funcionais

Requisito não funcional 1 – Tecnologia de interface gráfica para usuário: A interface gráfica deve ser baseada em TKinter;

Requisito não funcional 2 – Suporte para a especificação de projeto: a especificação de projeto deve ser produzida com a ferramenta Visual Paradigm;

Requisito não funcional 3 – Interface do programa: A interface do programa será produzida conforme o esboço da imagem abaixo.

Apêndice: Regras Inscryption

O jogo Inscryption é disputado entre 2 jogadores em uma mesa de 8 posições, organizado em duas linhas com 4 posições para cada jogador. O objetivo dos jogadores é pender completamente a balança que controla a “vida” dos jogadores para o outro lado, caracterizando a condição de vitória.

Elementos do jogo:

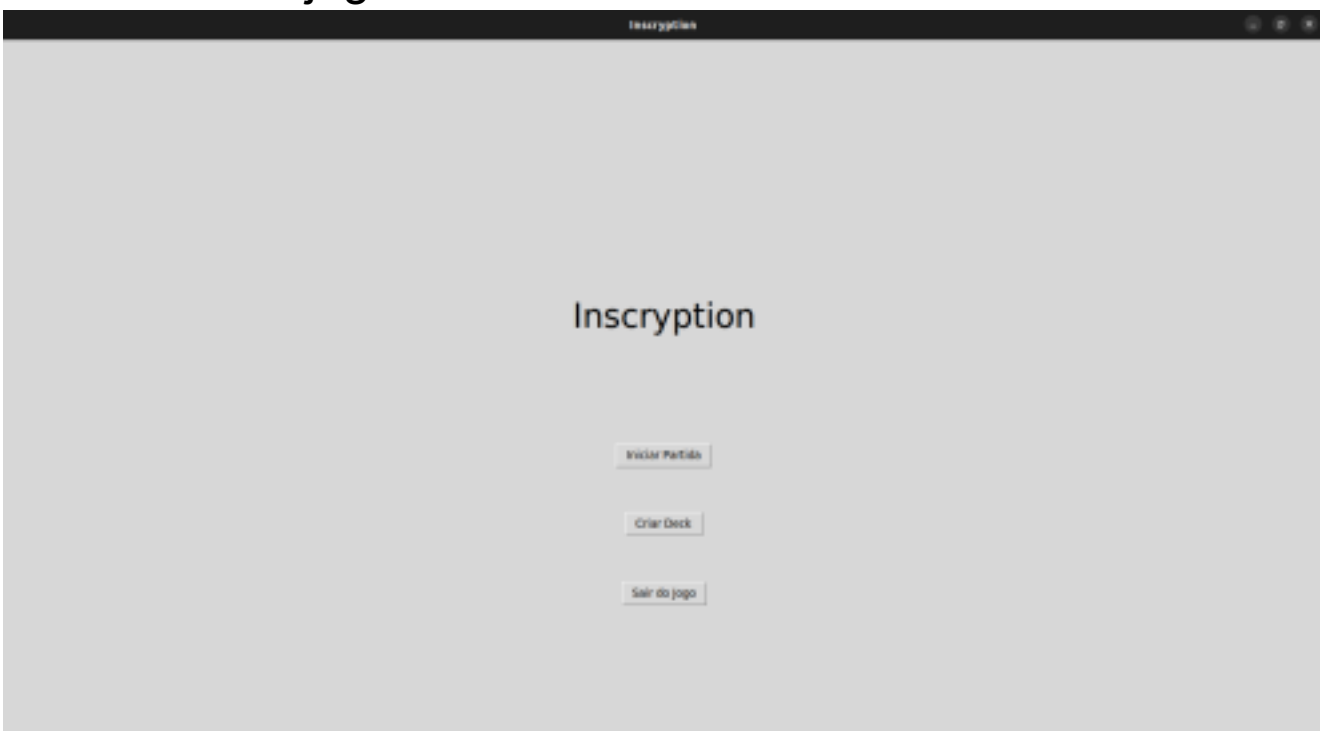

Figura 1 - Menu principal do jogo

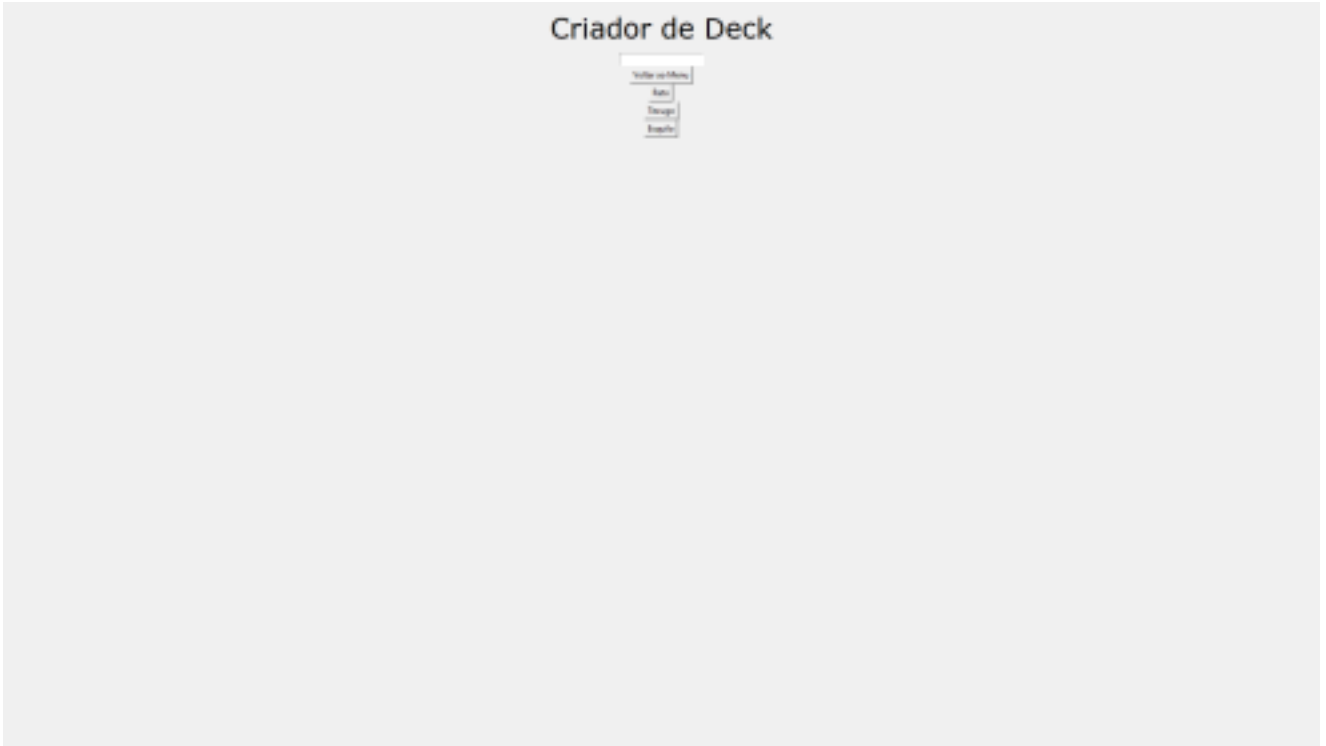

Figura 2 - Tela de criação do deck

Figura 3 - elementos do jogo Inscryption

Além da mesa, o jogo conta com uma balança, que controla quanto de dano cada jogador infligiu ao outro. O objetivo é fazer a balança pender completamente para o lado do adversário, marcando 7 pontos na diferença de dano entre os jogadores. Se isso acontecer, o adversário perde a rodada. Da mesma forma, se a balança pender para o lado do jogador, este será derrotado.

Baralhos

No início de cada partida, cada jogador tem dois baralhos:

Um baralho de criaturas, que contém diversas cartas com atributos diferentes, como: Wolf, Bear e Vulture.

Um baralho de Esquilos, usado para ser sacrificado para invocar cartas mais poderosas, visto que Esquilos não causam dano, apenas possuem vida.

Atributos das Cartas

Cada carta de criatura tem atributos que afetam seu desempenho na batalha:

Dano: O quanto a carta pode causar de dano ao oponente ou a uma carta adversária.

Vida: Quantos pontos de vida a carta tem antes de ser destruída.

Custo de Invocação: O que é necessário para colocar a carta na mesa. Algumas cartas requerem sacrifícios de outras criaturas, enquanto outras cartas necessitam de ossos para serem sacrificadas, sendo estes ossos obtidos a partir do sacrifício de cartas.

Selo: Algumas cartas possuem selos (habilidades especiais), que concedem efeitos únicos, como aumentar a quantidade de dano ou vida da carta a cada rodada.

Custo de Invocação e Sacrifícios

Muitas criaturas requerem sacrifícios para serem invocadas. Um Esquilo ou qualquer outra criatura já colocada na mesa pode ser sacrificada para pagar o custo de uma carta mais poderosa, como um Lobo (que exige dois sacrifícios). Já outras cartas podem ser invocadas a partir de Ossos, que são obtidos pelo sacrifício de outras cartas.

Sistema de Combate

No início de cada turno, o jogador compra uma carta de um dos dois baralhos (de criaturas ou de Esquilos). Em seguida, o jogador pode colocar uma carta na sua linha, se houver espaço, e começar a batalha:

Se uma criatura atacar uma posição vazia na linha do oponente, o dano vai diretamente para a balança, somando pontos na

balança adversária.

Se houver uma criatura na frente da carta atacante, os dois valores de ataque e vida se comparam, com o ataque reduzindo diretamente a vida da criatura oponente. Quando a vida chega a zero, a carta é destruída e removida do tabuleiro.

Habilidades Especiais (Selos)

Algumas cartas têm habilidades especiais, chamadas de selos, que modificam os seus atributos existentes. Os selos são ativados após o jogador passar o turno, ou seja, após realizar a sua jogada. Entre as habilidades presentes estão:

Amplificador (Amplifier): Este selo aumenta o atributo dano da carta. Este aumento irá variar de acordo com o custo que a carta exige para ser invocada.

Guardião (Guardian): Este selo incrementa o atributo vida da carta, tornando-a mais resistente. Este aumento de vida irá variar de acordo com o custo que a carta exige para ser invocada.

Encerramento da partida

A partida é decretada como finalizada ao estar na situação em que a diferença de pontos na balança de ambos os jogadores for maior ou igual a 7. Exemplo: Jogador local com 6 pontos na sua balança e Jogador remoto com 13 pontos na balança. Nesta ocasião, o vencedor é o jogador local, visto que somou uma pontuação >= 7 na balança do seu adversário. Sendo assim, a partida é encerrada.