

Projeto Mau-Mau

Especificação de Requisitos de Software

Req_PR_JogoMauMauDistribuido_2023agu24

Versão 1.0

6/9/2023

Versão	Autor(es)	Data	Ação
1.0	Eduardo Ribeiro Heitor Junior Ismael Coral Hoepers Heinzelmann Lucas de Lacerda Tinoco	6/9/2023	Estabelecimento de Requisitos

Sumário

<u>1.Introdução</u>	2
1.1 Objetivos de Desenvolvimento	2
1.2 Definições.....	2
<u>2. Visão Geral</u>	3
2.1 Arquitetura de software	3
2.2 Premissas de desenvolvimento	3
<u>3.0 Requisitos de Software</u>	3
3.1 Requisitos funcionais	3
3.2 Requisitos não funcionais	5

1. Introdução

1.1 Objetivo:

Desenvolvimento de um programa que suporte a disputa de partidas de jogo Mau-Mau na modalidade usuário contra usuário.

1.2 Definições:

Ideia geral do jogo: Mau-Mau é um jogo disputado entre 2 a 4 jogadores, utilizando um baralho convencional(baralho francês) cujo objetivo é esvaziar a mão primeiro.

Como Jogar: No início da partida um jogador é sorteado para começar, e o jogo seguirá no sentido horário. Cada jogador compra 5 cartas do baralho, e uma é colocada sobre a mesa(cartá referênciá). No turno do jogador, caso ele tenha uma carta de mesmo valor ou naipe da carta referênciá ele deve jogar-lá e passar o turno, em caso contrário ele compra uma carta do monte, caso seja possível jogar a carta ele joga, caso contrário ele passa o turno. A última carta jogada se torna a nova carta referênciá para o próximo jogador. Isso se repete até que um dos jogadores esvazie a mão.

Observações: Algumas cartas possuem habilidades, descritas em “Cartas especiais”.

Caso o baralho esvazie antes de um jogador vencer embaralha-se o monte que contém todas as cartas referências(monte de cartas em cima da mesa que define o valor e naipe das cartas que tem que ser jogada) menos aquela carta que atualmente é a referênciá, para ser o baralho.

Cartas especiais:

- Ás: O próximo jogador é pulado
- Rei: O sentido do jogo é invertido
- Valeté: Escolhe um naipe
- 7: O próximo jogador compra 2 cartas

2. Visão Geral

2.1 Arquitetura de software:

Cliente-servidor distribuído

2.2 Premissas:

- O Programa deve ser implementado em python
- Deve ser feita uma modelagem do programa em UML
- A modelagem do projeto deve ser produzido com Visual Paradigm
- O programa deve usar Pynetgames como suporte a execução distribuída

3. Requisitos da aplicação

3.1 Requisitos funcionais

Requisito funcional 1: Initialize -> Ao executar, o programa deve apresentar sua interface inicial (A tela do menu com os botões de “Jogar”, “Regras” e “Sair”), e solicitar o nome do jogador. Em seguida é solicitada a conexão com o Pynetgames server, retornando o resultado ao usuário

Requisito funcional 2: Receive connection -> O programa deve tratar da confirmação de sucesso de uma conexão. Além disso ao clicar no botão “Iniciar” no menu, e depois escolher o número de jogadores desejados, deve enviar uma solicitação de partida para o Pynetgames, que em caso de sucesso retornará a identificação e a ordem dos jogadores, ou em caso de falha a razão da impossibilidade de início da partida.

Requisito funcional 3: View rules -> O programa deve, ao clicar no botão “Regras” no menu, atualizar a sua interface, mostrando ao jogador as regras do jogo.

Requisito funcional 4: Receive match -> O programa deve receber uma notificação de início de partida, originada em Pynetgames server, em função da solicitação de início de partida por outro jogador conectado ao servidor. A interface deve ser atualizada com as informações recebidas dos jogadores, e caso o jogador local seja que inicia, deve habilitar seu procedimento de jogada.

Requisito funcional 5: Play card -> O programa deve permitir a um jogador habilitado selecionar uma carta da sua mão e jogar-lá na mesa. A carta selecionada deve ser visualmente destacada. Caso a carta selecionada não cumpra os requisitos para ser jogada (olhar as regras em “Definições”) deve ser notificada uma jogada irregular e o programa deve novamente aguardar a primeira ação (selecionar carta), caso contrário o jogador descarta a carta de sua mão e coloca ela sobre a pilha de descarte na mesa. O programa deve verificar o encerramento da partida (mão do jogador vazia), no caso de encerramento deve ser notificado o nome do jogador vencedor, em caso de não encerramento deve ser habilitado o próximo jogador.

Requisito funcional 6: Draw card -> O programa deve permitir que o jogador habilitado selecione o deck em cima da mesa para comprar uma carta, isto é, remover a carta do topo do deck e adicionar ele na mão do jogador. O jogador só poderá comprar uma carta caso nenhuma das cartas de sua mão cumpra os requisitos para ser jogada(Requisito funcional 5), em caso contrário o programa deve informar uma jogada irregular e o programa aguarda novamente a primeira ação (selecionar carta)

Requisito funcional 7: Receive move -> O programa deve poder receber uma jogada dos adversários, enviada por Pynetgames server, quando for a vez de algum dos adversários do jogador local. A jogada recebida deve ser um lance regular. O programa deve, em caso de possibilidade de jogar alguma carta(Requisito funcional 5), remover a carta da origem definida e colocá-la sobre o monte de descarte, ou em caso de compra(Requisito funcional

6) remover a carta do topo do baralho e adicioná-la na mão do jogador habilitado. Após isso, deve avaliar o encerramento da partida, no caso de encerramento deve ser notificado o nome do jogador vencedor, em caso de não encerramento deve ser habilitado o próximo jogador.

Requisito funcional 8: Receive disconnect -> O programa deve tratar do recebimento de desconexões por parte de algum jogador, utilizando ferramentas do Pynetgames. Em caso de desconexão o jogo deve ser resetado para seu estado inicial.

Requisito funcional 9: Receive error -> O programa deve tratar do recebimento de eventuais erros que ocorram no framework, utilizando ferramentas do Pynetgames. Em caso de erro jogo deve ser resetado para seu estado inicial.

3.2 Requisitos não funcionais

Requisito não funcional 1 : Interface gráfica -> Utilizar a biblioteca Tkinter do python para fazer os elementos gráficos

Requisito não funcional 2: A interface gráfica deve ter a seguinte aparência:

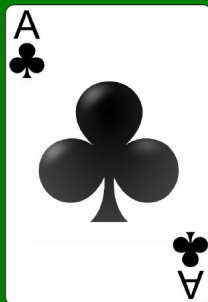


Mau-Mau

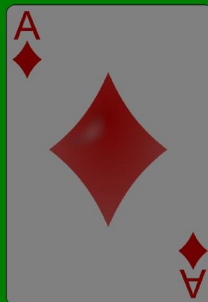
Jogar

Regras

Sair



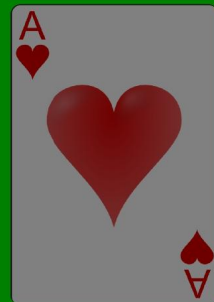
Jogador 1



Jogador 2



Jogador 3



Jogador 4

Iniciar



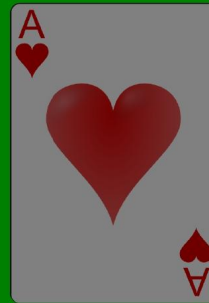
Jogador 1



Jogador 2



Jogador 3



Jogador 4

Iniciar

