



Pandemic

Gerência de Projeto e Manutenção de
Software



START **MENU**

Guilherme Vieira, Isadora Duarte, Izabel Soares, Lylian Pacheco, Mateus
Sacramento e Nayara Dornelas



Sumário

Tópicos

- [Controle de Versões](#)
- [Controle de Modificações](#)
- [Estratégia de Ramificação](#)
- [Pull Requests](#)

- [Burndown por iteração](#)
- [Análise de Valor Agregado](#)
- [Demo Parcial](#)

Controle de Versões



Commits realizados no repositório do GitHub contendo a implementação inicial de classes para o desenvolvimento do jogo, incluindo:

Exemplo:

- Classe de Baralho Infecção
- Classe de Baralho do Jogador
- Classe responsável pelo sistema de Infecção.

The screenshot shows a GitHub commit history with the following commits:

- o- Commits on May 5, 2025
 - feature: baralho de infecção**
guimvieira committed on May 5
 - feature: baralho de jogador**
guimvieira committed on May 5
 - feature: criacao de baralho**
guimvieira committed on May 5
- o- Commits on Apr 26, 2025
 - Merge pull request #2 from isadoradrt/cards_class_created** Verified 3ca2678
izabel-souza authored on Apr 26
 - feat**
izabel-souza committed on Apr 26
- o- Commits on Apr 22, 2025
 - Merge pull request #1 from isadoradrt/feature/estrutura-inicial** Verified e058d23
isadoradrt authored on Apr 22
 - feature: estrutura basica do jogo**
isadoradrt committed on Apr 22
- o- Commits on Apr 13, 2025
 - Initial commit**
isadoradrt authored on Apr 13

Controle de Modificação



Issue de documentação criada para controle de modificação.

Foi criada uma issue com o objetivo de organizar e rastrear a criação do arquivo README.md, centralizando as informações iniciais do projeto Pandemic (versão Hollow Knight), como:

- Nome e objetivo do projeto
- Regras principais
- Estado atual
- Integrantes da equipe

Criar README inicial com descrição do projeto Pandemic Hollow Knight #3

[Open](#)

 izabel-souza

Precisamos criar um arquivo README.md na raiz do repositório para descrever o projeto Pandemic na versão Hollow Knight que estamos desenvolvendo. Esse documento será a primeira fonte de informação para quem acessar o repositório e deve conter:

- Nome do projeto
- Objetivo do jogo
- Regras principais
- Estado atual do projeto (ex: "fase de design do mapa e personagens")
- Integrantes da equipe

Por exemplo:
"Este projeto é uma implementação digital do jogo de tabuleiro Pandemic na versão Hollow Knight, com foco em práticas de Gerência de Projeto e Manutenção de Software como controle de versão, issues e integração contínua."

Também é interessante deixar uma seção de "roadmap" ou "fases futuras", mesmo que ainda seja um rascunho.

[Create sub-issue](#) 

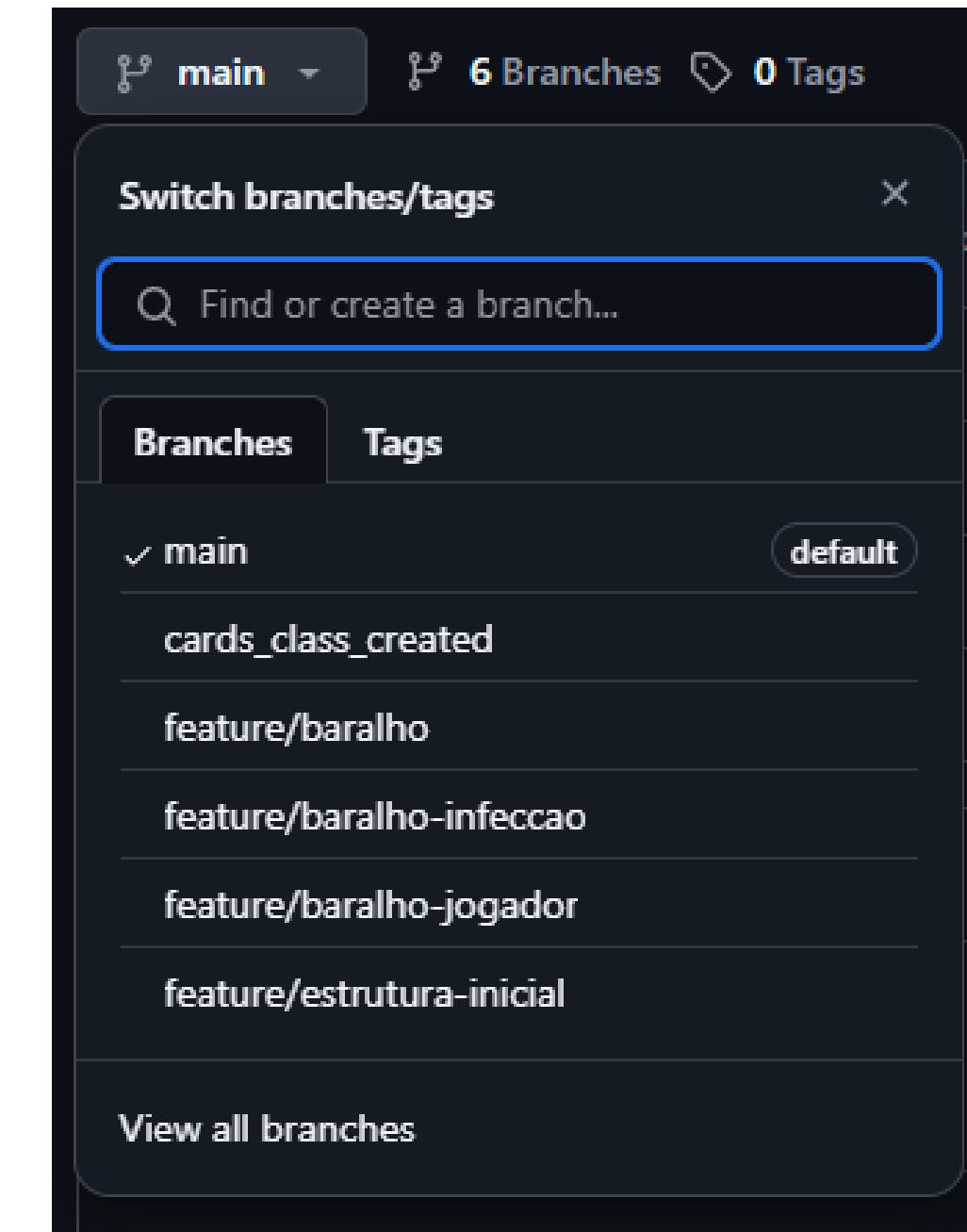
Estratégia de Ramificação



Git Flow

Adotamos a estratégia Git Flow para organizar o desenvolvimento do jogo, criando uma branch principal (main) estável e ramificações específicas (feature/*) para cada nova funcionalidade.

- facilita o controle de versões
- permite desenvolvimento paralelo
- as alterações passa por revisão via pull request antes de serem integradas



Pull Requests



Pull Requests realizados no repositório do jogo contemplam a criação e integração de componentes essenciais, como:

- A estrutura básica do jogo
- As classes responsáveis pelo baralho
- A classe que representa as cartas das cidades

<input type="checkbox"/> 0 Open	<input checked="" type="checkbox"/> 4 Closed	Author ▾	Label ▾	Projects ▾	Milestones ▾	Reviews ▾	Assignee ▾	Sort ▾
<input type="checkbox"/>	Carta de cidade	#5 by guimvieira was merged 3 minutes ago						
<input type="checkbox"/>	Feature/baralho	#4 by guimvieira was merged 2 minutes ago						
<input type="checkbox"/>	feat	#2 by izabel-souza was merged on Apr 26						
<input type="checkbox"/>	feature: estrutura basica do jogo	#1 by isadoradrt was merged on Apr 22						



Gráfico de Burndown

Burndown Sprint 3

— HORAS PLANEJADAS — HORAS FALTANTES

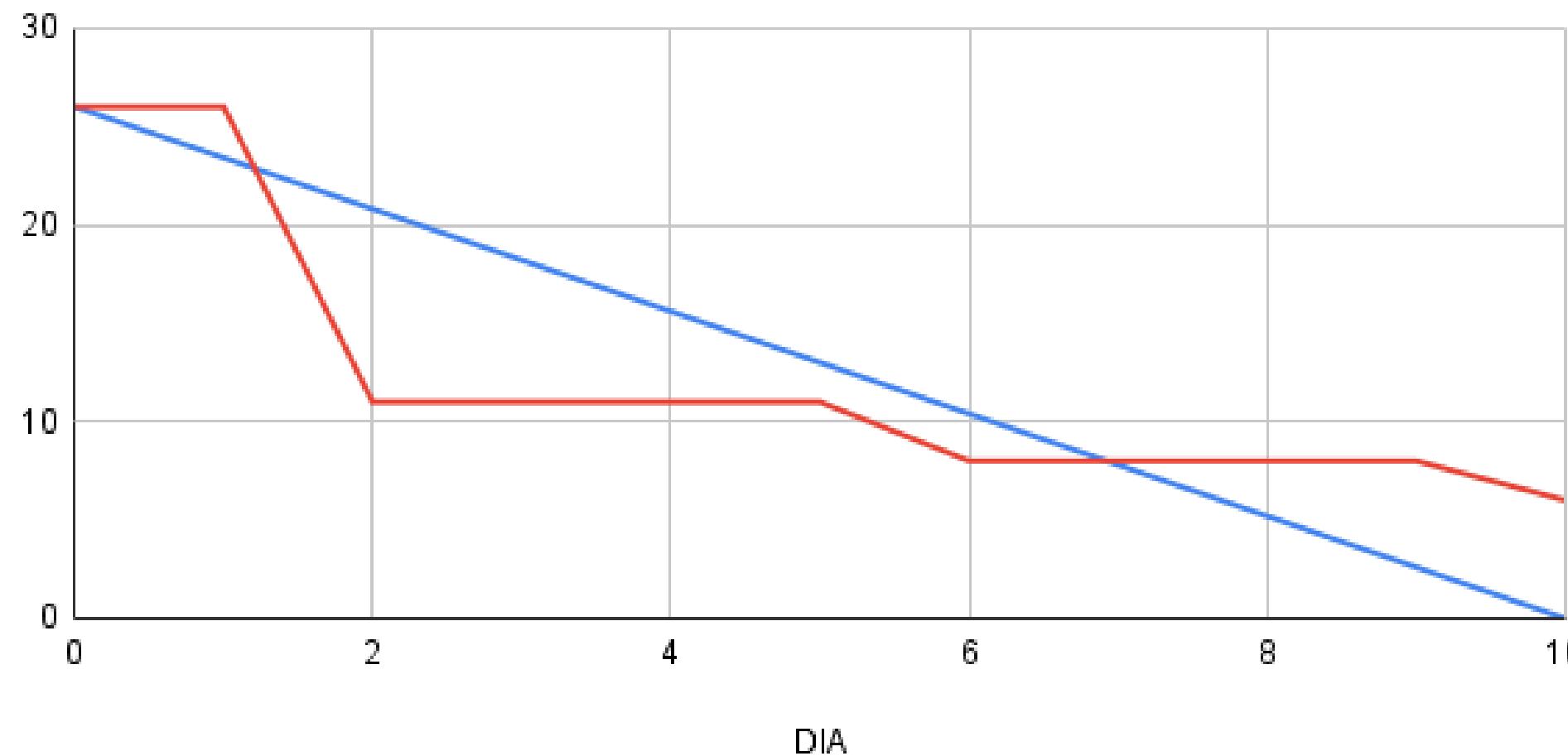
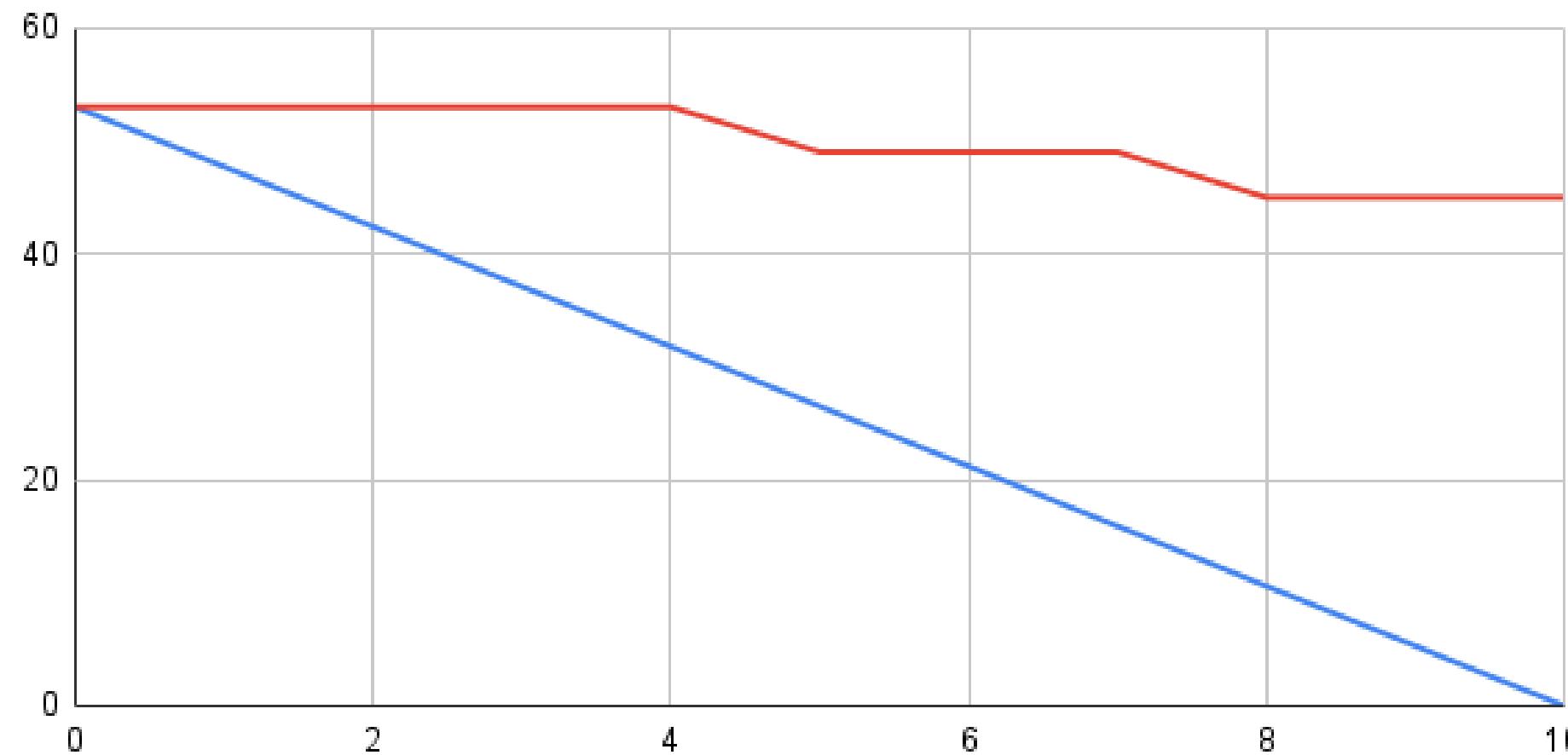




Gráfico de Burndown

Burndown Sprint 4

■ HORAS PLANEJADAS ■ HORAS FALTANTES





Análise de Valor Agregado

Até a Sprint 4

Indicador	Valor
PV (Planejado)	146h
EV (Executado)	83h
Variação de Cronograma (SV = EV – PV)	-63
Índice de Desempenho de Prazo (SPI = EV / PV)	0,5685



Demo Parcial



Carta

```
from enum import Enum

class TipoCarta(Enum):
    CIDADE = "Cidade"
    EVENTO = "Evento"
    EPIDEMIA = "Epidemia"

class Carta:
    def __init__(self, nome: str, tipo: TipoCarta, cor: str = None, descricao: str = None):
        self.nome = nome
        self.tipo = tipo
        self.cor = cor # Só faz sentido para cartas de cidade
        self.descricao = descricao # Só faz sentido para eventos ou epidemias

    def __str__(self):
        return f"{self.tipo.value}: {self.nome}" + (f" ({self.cor})" if self.cor else "")
```



Demo Parcial



Carta Cidade

```
from cartas import Carta, TipoCarta

class CartaCidade(Carta):
    def __init__(self, nome, cor):
        super().__init__(nome, TipoCarta.CIDADE, cor)

    def __str__(self):
        return f"Cidade - {self.nome} ({self.cor})"
```



Demo Parcial



Baralho

```
import random

# Classe de Baralho
class Baralho:

    def __init__(self, cartas=None):
        self.cartas = cartas or []

    def embaralhar(self):
        # Reembaralha o baralho
        random.shuffle(self.cartas)

    def tirar_carta(self):
        # Tira uma carta do baralho
        if self.cartas:
            return self.cartas.pop()
        return None

    def adicionar_carta(self, carta):
        # Adiciona uma carta ao baralho
        self.cartas.append(carta)
```



Demo Parcial



Baralho Jogador

```
from baralho import Baralho

class BaralhoJogador(Baralho):
    def __init__(self, cartas=None):
        super().__init__(cartas)

    def tirar_carta(self):
        if not self.cartas:
            print("O baralho de jogador está vazio!")
            return None

        carta = super().tirar_carta()
        if carta:
            print(f"Carta do baralho de jogador tirada!")
        return carta
```



Demo Parcial



Baralho Infecção

```
from baralho import Baralho

class BaralhoInfeccao(Baralho):
    def __init__(self, cartas=None):
        super().__init__(cartas)

    def tirar_carta(self):
        if not self.cartas:
            print("O baralho de infecção está vazio!")
            return None

        carta = super().tirar_carta()
        if carta:
            print(f"Carta de infecção tirada!")
        return carta
```

Demo Parcial



~*~*~*~*~

ALERTA!

SURTO DE INFECÇÃO

~*~*~*~*~

~*~*~*~*~

CURA

DESCOBERTA

~*~*~*~*~

~*~*~*~*~

ALERTA!

SURTO DE ESPOROS
FÚNGICOS

~*~*~*~*~

~*~*~*~*~

ALERTA!

SURTO DE VAZIO

~*~*~*~*~

Demo Parcial



MENSAGEM DE VITÓRIA/DERROTA

~*~*~*~*~

VITÓRIA!

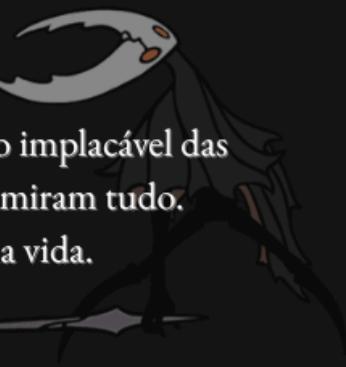


Graças aos esforços incansáveis dos especialistas, as trevas foram dissipadas. As doenças que assolavam Hallownest foram erradicadas, e a esperança floresce mais uma vez entre os ecos silenciosos do reino.

~*~*~*~*

~*~*~*~*

DERROTA!



Infelizmente, Hallownest sucumbiu à propagação implacável das doenças. O vazio, a infecção e os esporos consumiram tudo. Restam apenas sombras onde antes havia vida.

~*~*~*~*

Demo Parcial

ESPECIALISTAS



Demo Parcial

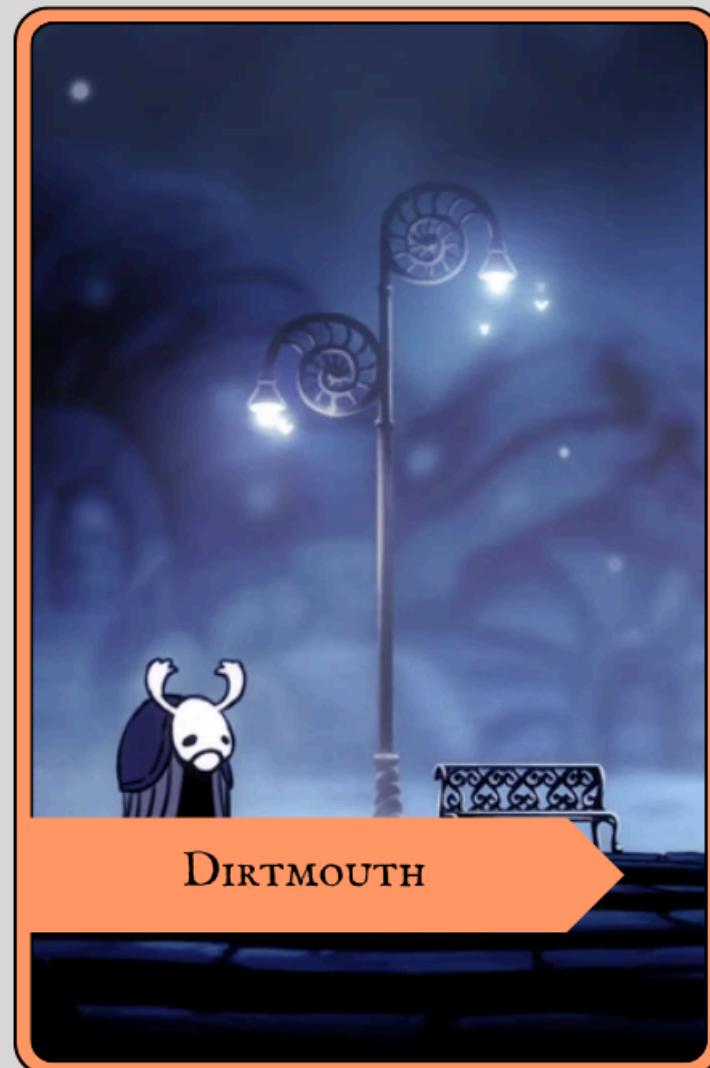
CIDADES



VILA DOS LOUVA-A-DEUS



ERMOS FÚNGICOS



DIRLMOUUTH



PICO DE CRISTAL

Demo Parcial

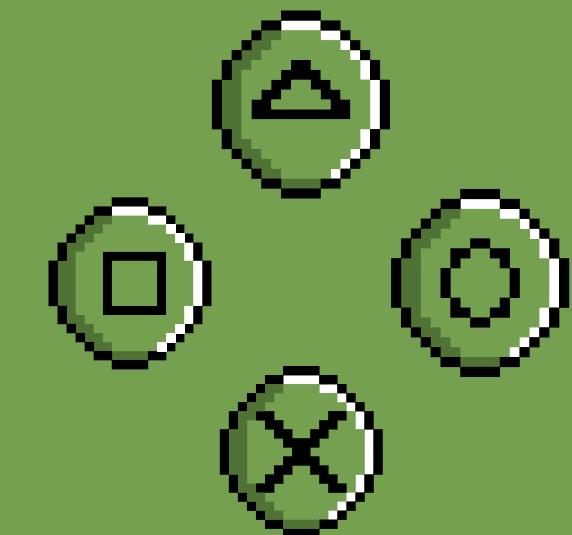
CIDADES





GitHub

[https://github.com/isadoradrt/
Trabalho-GPMS](https://github.com/isadoradrt/Trabalho-GPMS)





Obrigado!

