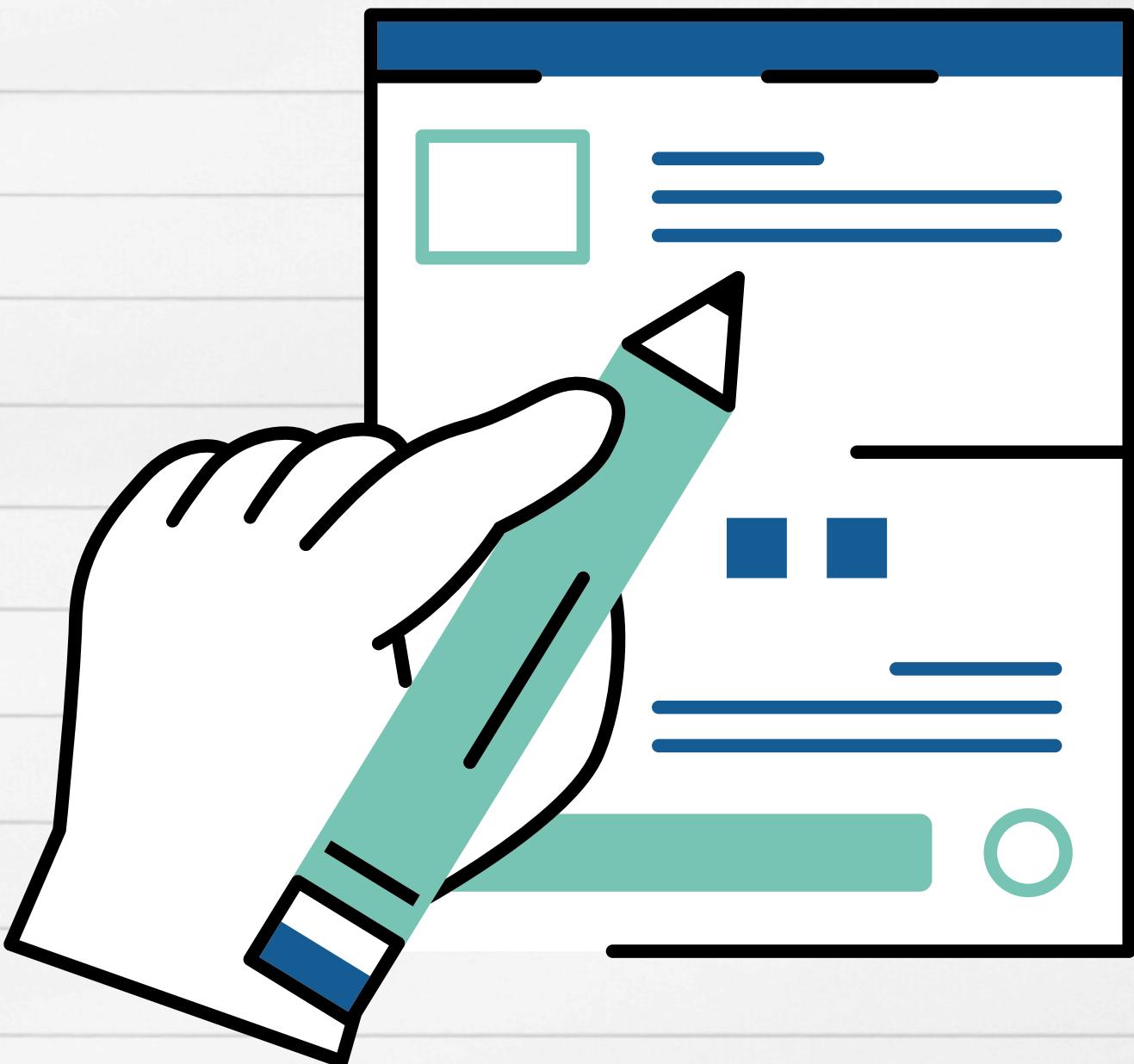


# RECOMENDAÇÃO DE CARTAS

Projeto MVC feito para realizar a recomendação das cartas do jogo. Esse projeto tem o objetivo de utilizar o MVC e fornecer endpoints para que os usuários obtenham suas cartas

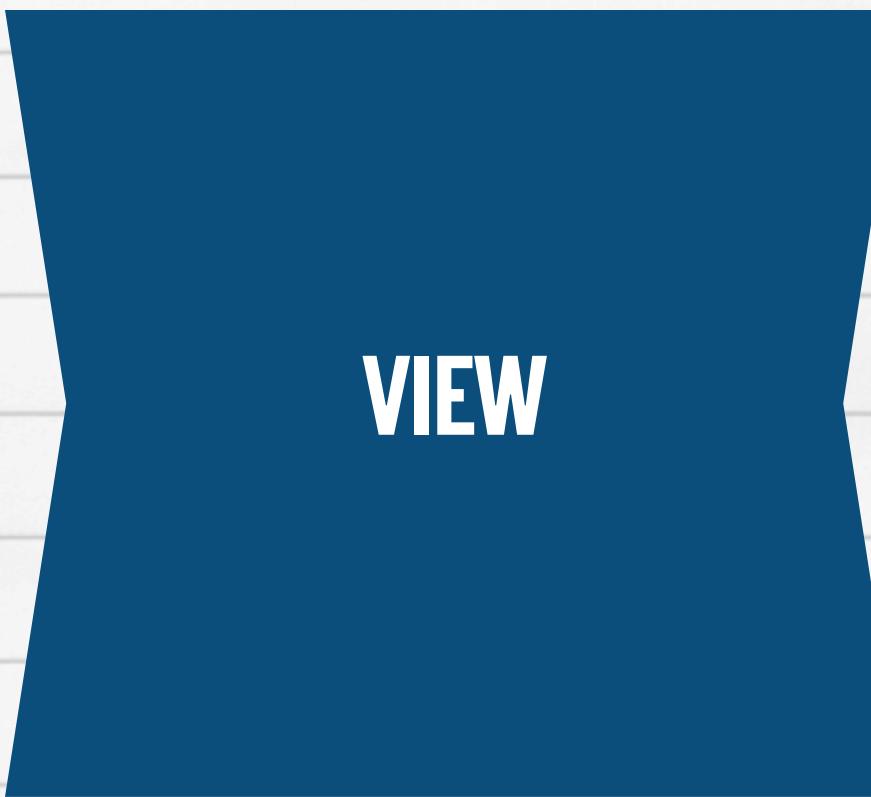


# MVC



# MVC

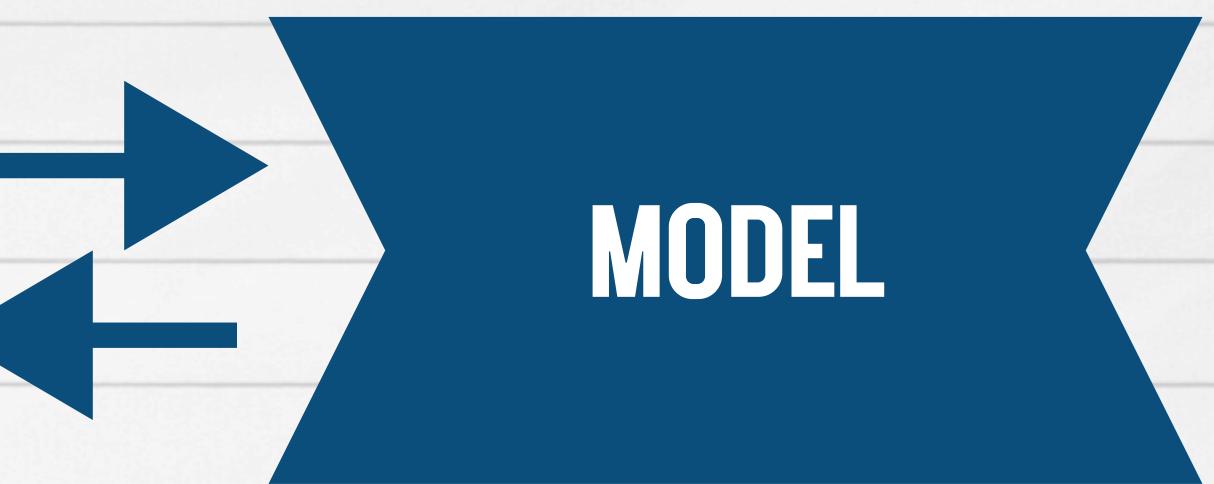
1. Usuário faz requisição



2. Controller irá “conversar” com o model necessário



3. Model irá buscar informações no banco

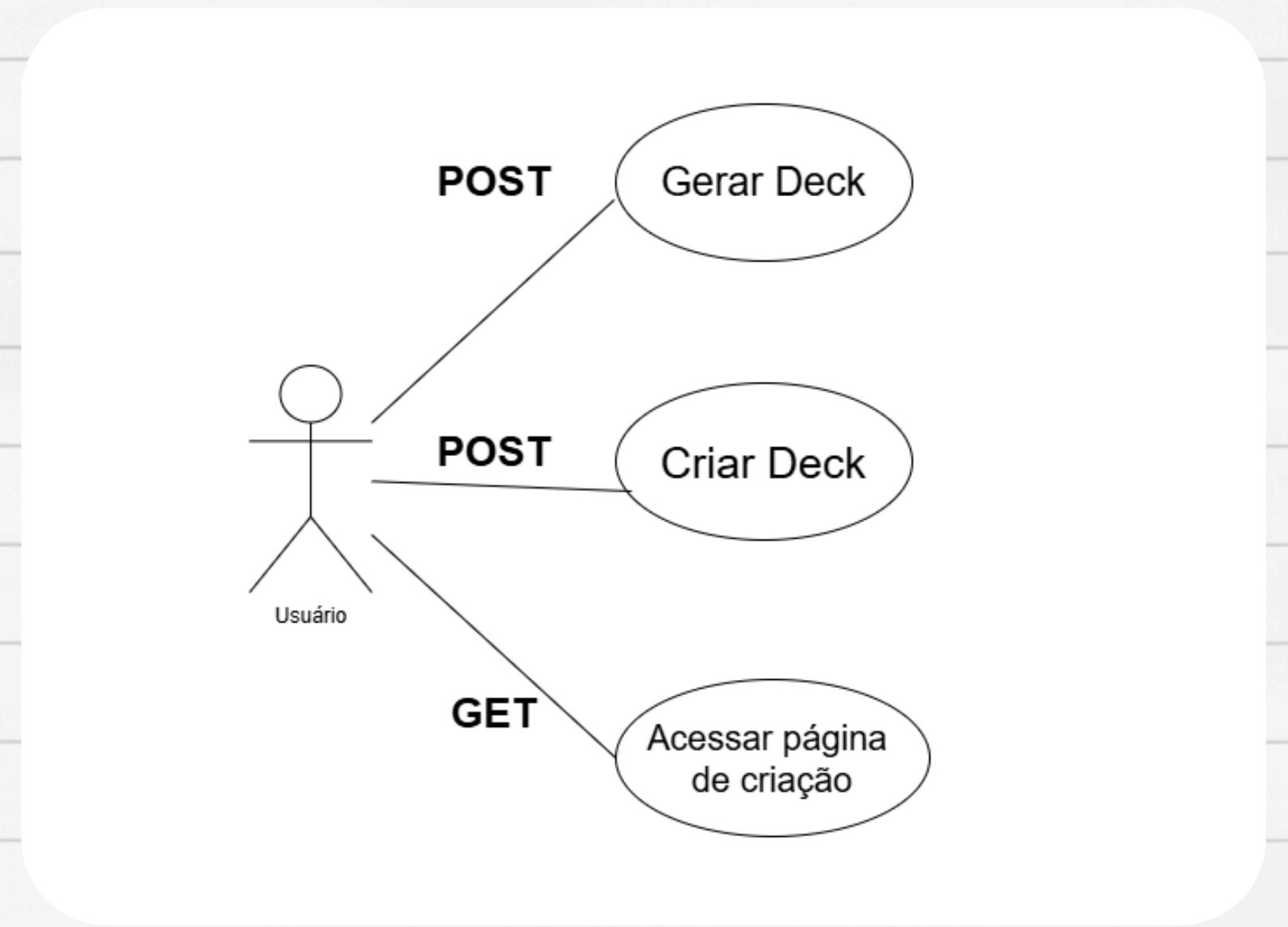


5. Retorna outra página pro usuário

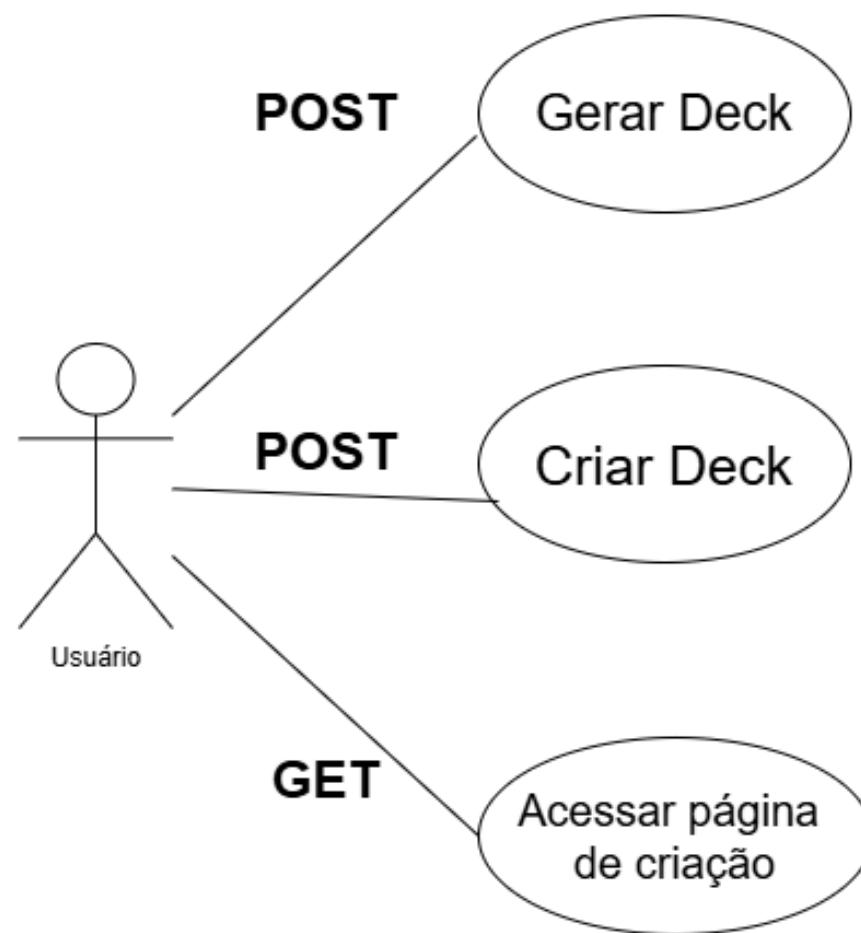
4. Controller pega essas informações do model

# INTRODUÇÃO

Objetivo: realizar a recomendação de cartas aleatórias para um usuário final

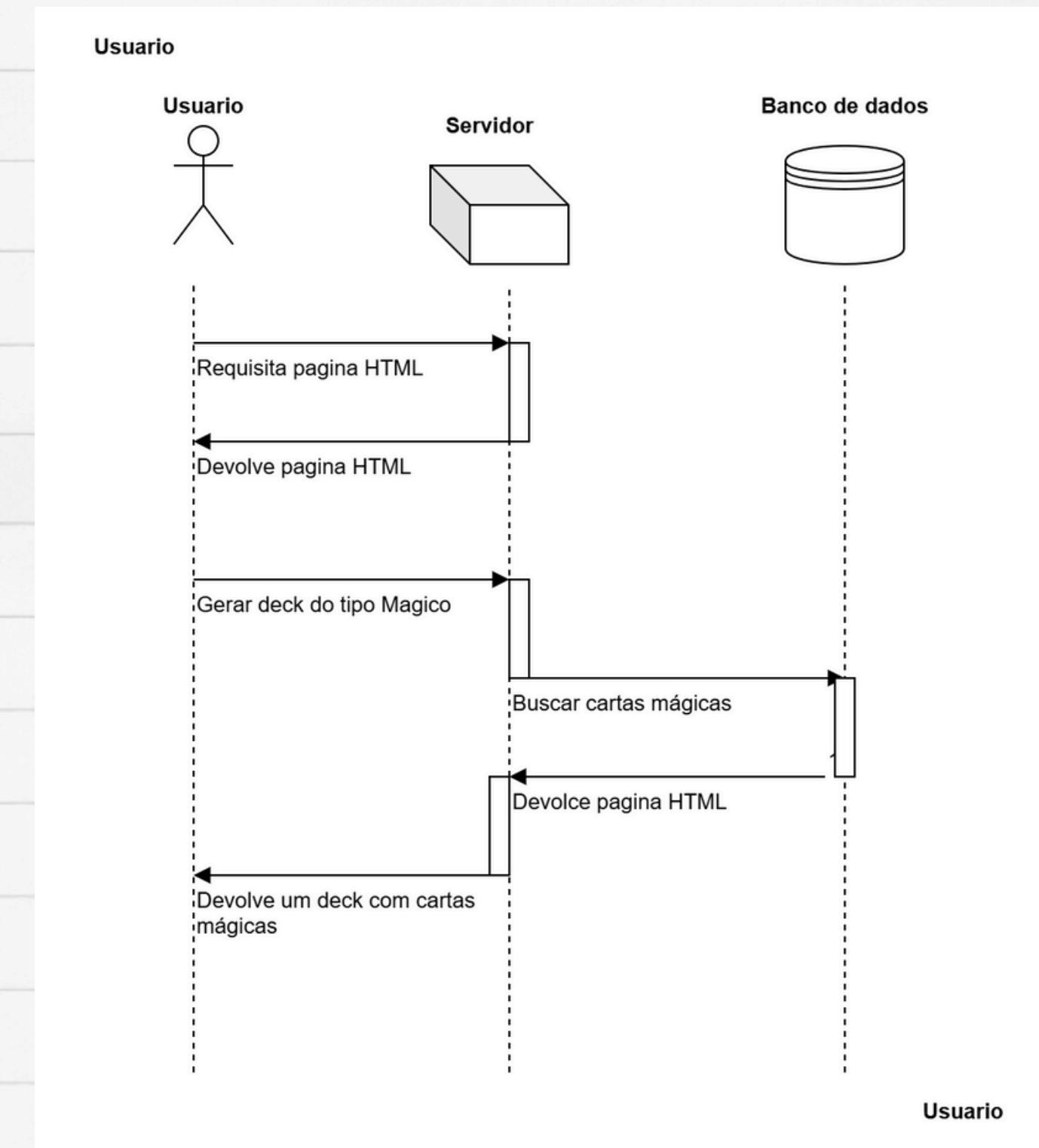


# FUNCIONALIDADES



- 1. Gerar um deck de cartas de um tipo específico**
  - a. Exemplo: gerar um deck de cartas mágicas
- 2. Criar um deck com cartas específicas**
  - a. Exemplo: criar um deck com as cartas x, y e z
- 3. Acessar o formulário de geração de decks**

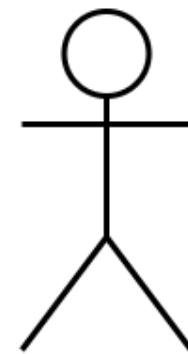
# EXEMPLO DE USO GERAL



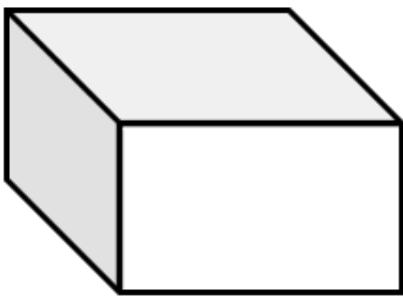
# EXEMPLO DE USO 1

**Usuario**

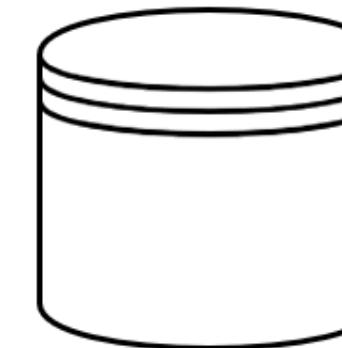
**Usuario**



**Servidor**

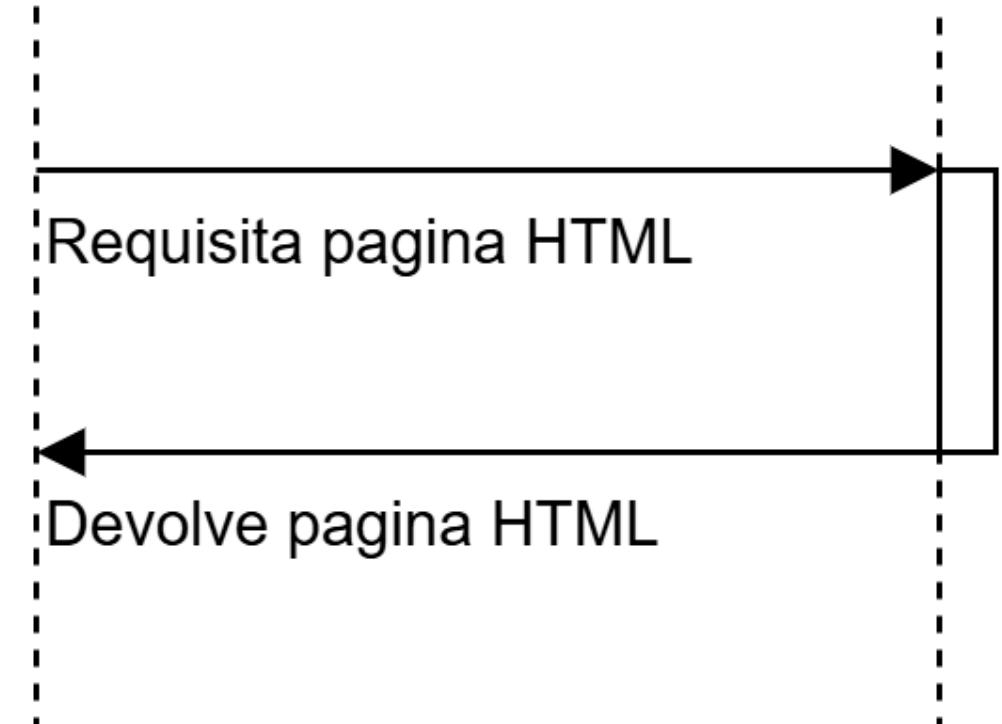


**Banco de dados**

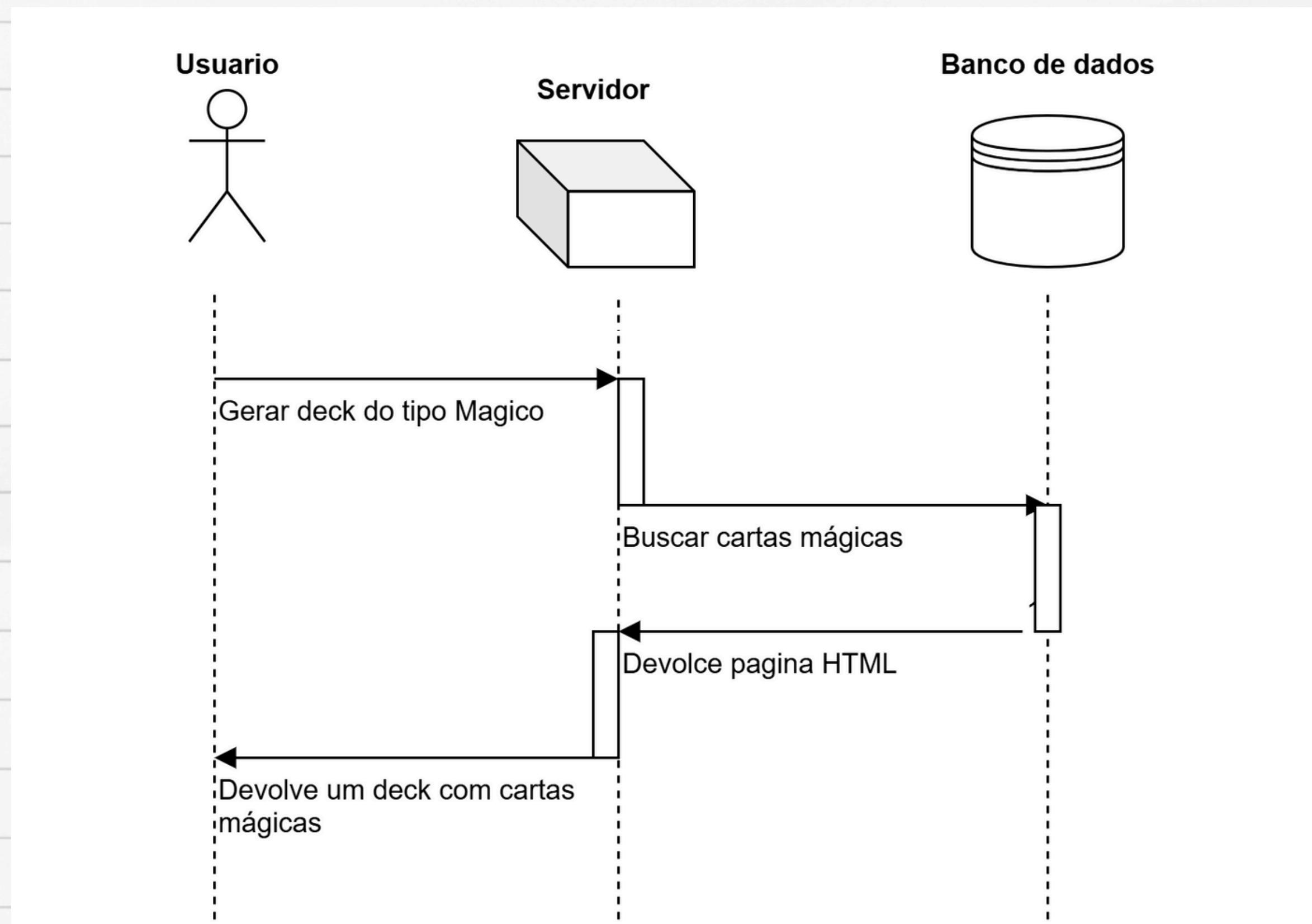


Requisita pagina HTML

Devolve pagina HTML



# EXEMPLO DE USO 2



# DIAGRAMA DE CLASSES

User

- id: int
- name: string
- email: string

Card

- id: int
- name: string
- attack: int
- defense: int
- type: string
- cost: int
- speed: int

Deck

- userId: int
- cards: Card[]

# PRINCÍPIOS

## Aberto-Fechado

Novos tipos de cartas podem ser adicionados (extensão). Porém, os tipos atuais de cartas devem permanecer os mesmos

```
if (type === "distancia") {  
    result = await RangeStrategy.build(Card);  
} else if (type === "corpo") {  
    result = await MeleeStrategy.build(Card);  
} else if (type === "mágico") {  
    result = await MagicStrategy.build(Card);  
} else {  
    return res.status(400).json({ message: "Tipo de carta inválido" });  
}
```

# PRINCÍPIOS

## Princípio de Liskov

Cada uma das classes de estratégias podem ser substituídas pela classe generalista “Strategy” sem modificar o funcionamento

```
const { MeleeStrategy } = require("../strategies/MeleeStrategy");
const { MagicStrategy } = require("../strategies/MagicStrategy");
const { RangeStrategy } = require("../strategies/RangeStrategy");
```

# PRINCÍPIOS

## Inversão de dependências

Dependência de classes abstratas, e não de implementações. Aqui fazemos uma configuração geral do banco de dados, podendo mudá-lo a qualquer momento

```
const db = require("./src/models/db");
```

# PRINCÍPIOS

## SRP

Cada rota possui sua responsabilidade única:  
renderizar a pagina de deck ou gerar o deck

```
renderDeckPage: (_, res) => { ...  
},
```

```
generateDeck: async (req, res) => { ...  
},
```