

BACKEND»

REINVENTANDO

MVC MASTERY: A ESSÊNCIA MVC NO BACKEND



Felipe Martins

DESVENDANDO A ARQUITETURA MVC.

Desvendando a Arquitetura MVC no Desenvolvimento Backend

Na jornada do desenvolvimento de software, a compreensão da arquitetura MVC é fundamental. MVC, ou Model-View-Controller, é um padrão de design amplamente utilizado que organiza o código em três componentes principais: Model, View e Controller. Vamos explorar cada um desses componentes e como eles trabalham juntos para criar aplicativos robustos e escaláveis.



01

O MODEL

O MODEL

O Modelo representa os dados e a lógica de negócios do aplicativo. Ele lida com a manipulação e persistência dos dados, bem como as regras de negócios. Por exemplo, em uma aplicação de lista de tarefas, o Modelo pode consistir em uma classe Task com métodos para adicionar, remover e atualizar tarefas.

```
// Exemplo de Modelo em uma aplicação de lista de tarefas
class Task {
  constructor(title, description) {
    this.title = title;
    this.description = description;
    this.completed = false;
  }

  markAsCompleted() {
    this.completed = true;
  }
}
```

snappify.com

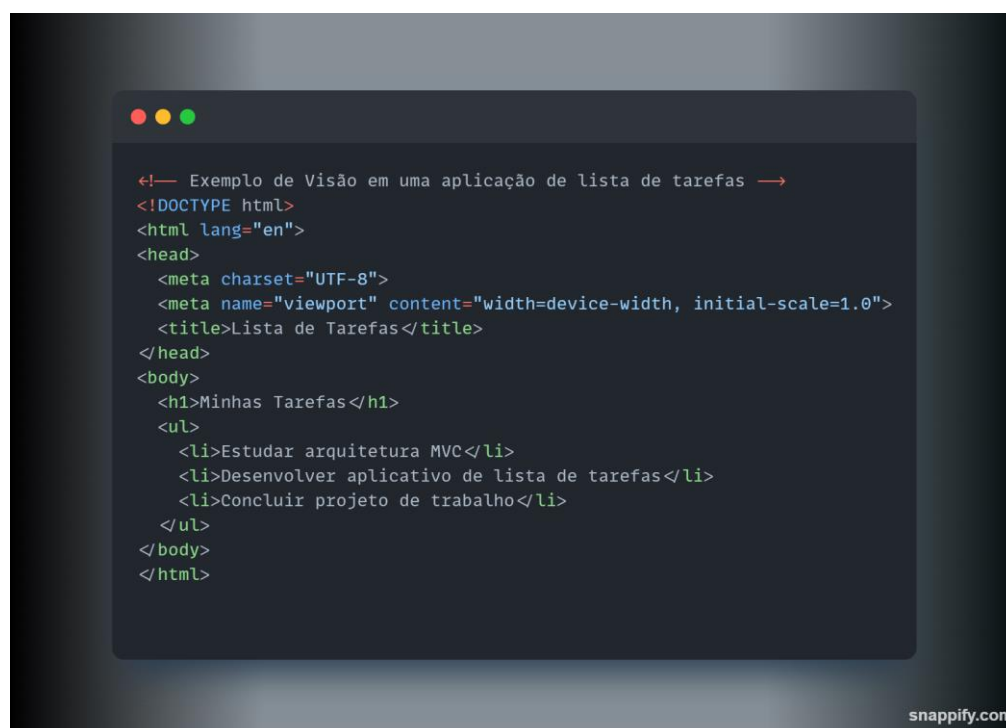


02

O VIEW

O VIEW

A Visão é responsável por exibir os dados ao usuário de uma maneira compreensível. Ela é separada do Modelo e do Controller, garantindo uma clara separação de preocupações. Por exemplo, em uma aplicação de lista de tarefas, a Visão pode ser uma página da web que exibe todas as tarefas disponíveis em uma lista formatada.



```
<!-- Exemplo de Visão em uma aplicação de lista de tarefas -->
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Lista de Tarefas</title>
</head>
<body>
  <h1>Minhas Tarefas</h1>
  <ul>
    <li>Estudar arquitetura MVC</li>
    <li>Desenvolver aplicativo de lista de tarefas</li>
    <li>Concluir projeto de trabalho</li>
  </ul>
</body>
</html>
```



03

O CONTROLLER



O CONTROLLER

O Controlador atua como intermediário entre o Modelo e a Visão. Ele recebe as entradas do usuário, processa essas entradas e atualiza o Modelo conforme necessário. Por exemplo, em uma aplicação de lista de tarefas, o Controlador pode receber um pedido para adicionar uma nova tarefa, criar uma instância do Modelo Task correspondente e atualizar a Visão para refletir a mudança.

```
// Exemplo de Controlador em uma aplicação de lista de tarefas
function addTask(title, description) {
  const newTask = new Task(title, description);
  // Lógica para adicionar a nova tarefa ao Modelo e atualizar a Visão
}
```

snappify.com



04

AS ROTAS

AS ROTAS

As Rotas são responsáveis por mapear URLs para funções específicas do Controlador. Elas direcionam as solicitações do cliente para os controladores apropriados com base no caminho da URL e nos métodos HTTP utilizados. Por exemplo, em uma aplicação web usando Node.js e Express, podemos definir rotas para lidar com solicitações de diferentes páginas ou recursos.

```
// Exemplo de definição de rota em uma aplicação Express.js
const express = require('express');
const app = express();

// Rota para a página inicial
app.get('/', (req, res) => {
  res.send('Bem-vindo à página inicial!');
});

// Rota para exibir uma lista de usuários
app.get('/users', (req, res) => {
  // Lógica para recuperar e exibir a lista de usuários
});
```

snappify.com



05

OS MIDDLEWARES

OS MIDDLEWARES

Os Middlewares são funções que são executadas entre a recepção da solicitação do cliente e o envio da resposta pelo servidor. Eles permitem executar tarefas comuns, como autenticação, validação de entrada e manipulação de erros, de forma modular e reutilizável. Por exemplo, em uma aplicação Express.js, podemos usar middlewares para verificar se o usuário está autenticado antes de permitir o acesso a determinadas rotas.

```
// Exemplo de middleware para autenticação em uma aplicação Express.js
function authenticate(req, res, next) {
  if (req.isAuthenticated()) {
    return next();
  }
  res.redirect('/login');
}

// Rota protegida que requer autenticação
app.get('/profile', authenticate, (req, res) => {
  res.send('Página do perfil do usuário');
});
```

snappify.com



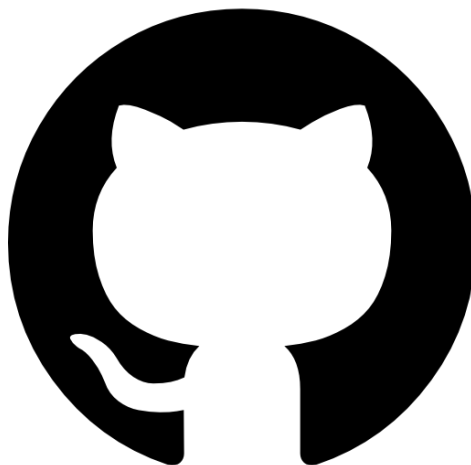
AGRADECIMENTOS



OBRIGADO POR LER ATÉ AQUI

Este Ebook foi gerado por IA e diagramado por humano.
O passo a passo se encontra no meu Github.

Esse conteúdo foi gerado com fins didáticos de construção, não foi realizado uma validação cuidadosa humana no conteúdo e pode conter erros gerados por uma IA.



<https://github.com/felmartins1985/prompts-recipe-to-create-a-ebook>

