Roteiro de Estudos

Desenvolvimento Web Full Stack

Natal, 28.02.2025

Roteiro de Estudo - Desenvolvimento Web Full Stack

Objetivo: Aprender a desenvolver aplicações web, do front-end ao back-end, com foco em praticidade e autonomia.

---

1. Introdução à Programação e ao Desenvolvimento Web

- **Objetivos**: Compreender os conceitos básicos de programação e como a web funciona.

Lógica de Programação

1. **Variáveis e Tipos de Dados**
    - Definição de variáveis.
    - Tipos primitivos: string, number, boolean, etc.
    - Operadores aritméticos e lógicos.
2. **Estruturas de Controle**
    - **if/else**, **switch**.
    - Laços: **for**, **while**.

Exercício:

- Crie um script que receba 3 números e retorne o maior deles.
- Crie um contador que incremente de 1 até 100 e imprima os múltiplos de 3 e 5.

Funções e Arrays

1. **Funções**
    - Funções simples e com parâmetros.
    - Funções que retornam valores.
2. **Arrays**
    - Declaração de arrays.
    - Manipulação de arrays: .push(), .pop(), .map(), .filter().

Exercício:

- Crie uma função que recebe um array de números e retorna a soma de todos os elementos.
- Crie uma função que recebe um nome e retorna o nome invertido.

Objetos

1. **Objetos**
    - Criação de objetos.
    - Acesso a propriedades e métodos.
2. **Prática**
    - Defina um objeto representando um usuário (nome, idade, email) e escreva funções para acessar e atualizar essas propriedades.

Exercício:

- Crie um objeto "carro" com propriedades como marca, modelo e ano. Escreva uma função que altere a cor do carro.

Como a Web Funciona

1. **O que é um servidor e cliente?**
    - Diferença entre servidor e cliente.
    - Como a web se comunica através do protocolo HTTP.
2. **Requisições HTTP**
    - Métodos: GET, POST, PUT, DELETE.
    - O que é uma API?

Exercício:

- Pesquise e entenda como fazer uma requisição simples em JavaScript para consumir uma API pública (ex: API de posts do JSONPlaceholder).

---

2. Desenvolvimento Front-End

- **Objetivo**: Aprender HTML, CSS e JavaScript básico para criar páginas web interativas.

HTML

1. **Estrutura básica de uma página HTML**
    - Tags básicas: <html>, <head>, <body>.
    - Estrutura de uma página web.
2. **Formulários e Inputs**
    - <form>, <input>, <textarea>, <select>.

Exercício:

- Crie uma página simples com um formulário de contato com campos para nome, email e mensagem.

CSS

1. **Estilos Básicos**
    - Cores, fontes e tamanhos.
    - Layout: Margens, padding, bordas.
2. **Layout Responsivo**
    - Flexbox e Grid.

Exercício:

- Crie uma página responsiva usando Flexbox que tenha um menu lateral e uma área de conteúdo.

JavaScript Básico

1. **Interação com o DOM**
    - Acessando elementos: document.getElementById(), document.querySelector().
    - Manipulando o conteúdo: .innerHTML, .textContent.
2. **Eventos**
    - Como usar eventos: addEventListener().

Exercício:

- Crie um contador simples onde o número aumenta ao clicar em um botão.
- Crie uma página com um formulário que valide os campos usando JavaScript (verificando se o campo nome não está vazio).

Projeto Inicial: Relógio Digital

1. **Desafio**
    - Crie um relógio digital que atualize a cada segundo, exibindo horas, minutos e segundos.

---

3. JavaScript Avançado e React

- **Objetivo**: Aprofundar no JavaScript, aprender React e começar a consumir APIs.

JavaScript Avançado

1. **Programação Assíncrona**
    - O que são callbacks, promises e async/await.

Exercício:

- Crie uma função que simule um delay (ex.: setTimeout) e utilize async/await para esperar por um resultado.

Manipulação de APIs

1. **Consumindo APIs**
    - Usando fetch para fazer requisições GET e POST.

Exercício:

- Consuma uma API pública e exiba as informações na página. (Exemplo: exibir informações de usuários do GitHub ou posts do JSONPlaceholder).

Introdução ao React

1. **Instalação e Configuração**
    - O que é o React? Como instalar e configurar.
2. **Componentes**
    - Criando componentes funcionais e com estado.

Exercício:

- Crie um componente App que mostre uma lista de tarefas simples e permita adicionar novas tarefas.

React Avançado

1. **Manipulando Eventos e Formulários**
    - Como trabalhar com eventos em React (ex.: onClick, onChange).
2. **Estados e Props**
    - Entendendo o estado (useState) e passando dados com props.

Exercício:

- Adicione a funcionalidade de deletar tarefas da lista que você criou no dia anterior.

---

4. Desenvolvimento Back-End (Node.js + MySQL)

- **Objetivo**: Aprender a construir APIs com Node.js, Express e conectar com bancos de dados MySQL.

Introdução ao Node.js

1. **Instalação e Configuração**
    - O que é o Node.js e como configurar o ambiente.
2. **Criação de Servidor HTTP**
    - Usando o módulo http do Node.js para criar um servidor simples.

Exercício:

- Crie um servidor que responda com "Hello, World!" ao acessar a raiz (/).

Express.js e Rotas

1. **Introdução ao Express.js**
    - Como configurar o Express e criar rotas.
2. **Middlewares**
    - Usando middlewares para manipulação de requisições.

Exercício:

- Crie uma API simples que tenha rotas para CRUD de tarefas (adicionar, listar, editar e excluir).

Banco de Dados MySQL

1. **Conectando Node.js ao MySQL**
    - Instalando o pacote mysql e configurando a conexão.
2. **Operações CRUD com MySQL**
    - Como realizar operações de inserir, ler, atualizar e deletar dados.

Exercício:

- Crie uma tabela tasks no MySQL e conecte-a com a API para salvar e recuperar as tarefas.

Autenticação com JWT

1. **JWT (JSON Web Token)**
    - O que é o JWT? Como utilizá-lo para autenticação.

2. **Login e Registro**
    - Criação de rotas para login e registro de usuários.

Exercício:

- Adicione autenticação na API usando JWT e crie um sistema de login e registro de usuários.

---

5. Integração Front-End e Back-End

- **Objetivo**: Integrar a interface do usuário (React) com a API (Node.js + MySQL).

Conectando Front-End e Back-End

1. **Consumo de APIs no React**
    - Usando fetch ou axios para consumir a API criada no back-end.
2. **Exibindo Dados na Interface**
    - Mostrando os dados recebidos da API na interface do React.

Exercício:

- Conecte o front-end de tarefas com o back-end para exibir, adicionar e remover tarefas.

Projeto Final

1. **Desafio Final**
    - Conclua o desenvolvimento de um projeto completo: um **To-Do List** com autenticação de usuários.

Exercício:

- Implemente as funcionalidades completas: criação, edição, remoção de tarefas, login/logout e persistência dos dados no banco de dados.