**Back-End**

- Regras de negócio;

- Conexão Banco de dados;

- Envio de E-mail - (configuração conta, senha;)

- Comunicação webservices;

- Autenticação do usuário;

- Criptografia e segurança;

**comunicação com Front-End - JSON**

- Front-End Web

- Front-End Mobile

- Serviços externos

--------------------

**React**

**Abordagem de SPA**

As requisições trazem apenas dados como respostas e com esses dados, o front-end pode preencher as informações em tela.

A página nunca recarrega, otimizando a performance e dando vida ao conceito de SPA. Retornando apenas JSON podemos ter quatros front-end quisermos.

-------------------

**React Native**

Todo código feito é em JavaScript, esse código não é convertido em código nativo, melhor do que isso, o dispositivo passa a entender o código Javascript e a interface gerada é totalmente nativa.

**Expo**

Ele usa todos os sensores do celula(mapas, canmeras e geolocalização)

**2º dia**

**ROTA E RECURSOS**

/USERS - O CAMINHO DO RECURSO QUE VOCE QUER USAR - COMO UMA TAELA.

**METODOS HTTP:**

**.GET**: quando queremos buscar/lista informação do back-end

**.POST**: quando queremos criar uma informação do beck-end

**.PUT:** Alterar uma informação o back-end

**.DELETE**: Deletar um informação do back-end

**TIPOS DE PARAMETOS**

**Query PARAMS**: paramentos nomeados enviados na rota apos o "?" (filtros, paginação)

ex. http://localhost:3333/users?nome=Diego (onde ele traz o usuario diego) ou http://localhost:3333/users?page==2&nome=Diego&idade=25 (ele traz todos os Diego que tem idade 25)

* Para usarmos params temos que usar o request.query ele busca todas que tem uma deteminda nome

**Route Params**: Parâmetros utilizados para identificar recursos

Ex. no codigo colocamos no recurso /users completamos /:id que vai localiza usuario pelo id (app.get('/users/:id', (request, response) => { ) - no bouse para realizar a pesquisa na pagina vem http://localhost:3333/users/1 - qie /1 siguinifica id=1 codigo do usuario.

* Para usarmos Route Paramos, e buscar somente 1 codigo usamos const params = request.params e no recurso acrescentaremos users/:id

**Request Bory** : Corpo da requisição, utilizando para criar ou alterar recursos

* Para usar o bory temos que mudar nossa requisição para app.post('/users', (request, response) => { const body = resquest.body

**Receber JSON -** para o seu código receber um comando temos que escrever o codigo (app.use(express.json());) antes de comando, assim ele recebe o comando JDSON e converto para o código objeto do JS.

**Request** - é responsável por todos os dados da nossa requisição

**Response** - é responsável por retornar uma resposta para o usuário.

**INSOMNIA.REST**

Serve para testar o Back-end para saber se as rotas e métodos estão ok sem o Front-end

Instalando Nodemon

Monitoramento para desenvolvedor

Instalar o pacote npm install nodemon -D dento da raiz do back-end.

Depois de instalado vamos em pachage.json e altera o scripts “test”: para “start”:”nodemon index.js” e executa o comando npm start(para executar/istatar seu back-end e não precisar ficar reiniciando para testar a cada mudança no código.

**DIFERENÇA DE BANCO DE DADOS**

VAMOS ISNTALAR SQLITE NA APLICAÇÃO USANDO

- NPM INSTALL KNEX – INTALA KNEX

- NPM INSTALL SQLITE3 – INTALA SQLINTE

- NPX KNEX INIT – CRIS A PASTA KNEXLILE.JS NA RAIZ

**Conexão com o DB.**

CRIAR A PASTA SRC E JOGAR O INDEX DENTRO DELA, NESSA PASTA VAMOS JOCAR TUDO QUE DESENVOLVERMOS.

<https://github.com/HegerValesin/Rocketset.git>