Atividades da Semana Oministack 11.

**Aula 02**

**Conceitos aprendidos**

**Rotas e Recursos**

Rotas : É o endereço completo da url inclusive o recurso.

Recurso: É o final da URL representando provalvemente uma entidade da aplicação. Ex: /users

**Métodos HTTP**

Método GET: Buscar uma informação no Back-End

Método POST: Criar uma informação no Back-End

Método PUT: Alterar uma informação no Back-End

Método DELETE: Deletar uma informação no Back-End

**Tipos de Parâmetros**

Query Params: Parâmetros nomeados enviados na rota, após o simbolo de "?" e geramente servem para filtro, paginação.

Route Params: Parâmetros utilizados para identificar recursos.

Request Body: Corpo da requisição utilizado para criar ou alterar dados em recursos.

**Bancos de Dados**

Banco de Dados Relacional : MySQL, SQLite, PostgreSQL, Oracle, Microsoft SQL Server, estes prezam a estrutura e a organização dos dados. Usam a linguagem SQL.

Banco de Dados Não Relacional : MongoDB, CouchDB, etc... Servem mais como forma de armazenamento sem se preocupar muito com organização e relacionamentos. Linguagem unica para cada DB.

Formas de comunicação com banco de dados: Por Driver ou por Query Builder.

Ferramenta para comunicação com o banco de dados utilizada: Knex, web site de ferência [Knex](http://knexjs.org/)

Entidades da nossa aplicação : ONG, Caso (Incident)

Funcionalidades : Login de ONG, Logout ONG, Cadastro de ONG, Cadastrar novo Caso, Deletar Casos. Listar Casos especificos de uma ONG, Listar todos dos Casos, Entrar em contato com a ONG.

**Realizei uma correção de Logica na minha opnião necessária.**

Não existe uma checagem da geração automática do Id Randômico das Ongs, o que quer dizer que apesar da chance ser baixa, pode acontecer de uma mesma Ong ter um id cadastrado de uma ong que já existe, e ai na seleção de profile ele vai retornar sempre a ultima ong que esse id, por isso implementei essa lógica aqui, para impedir esse cenário.

OngControler.js

const connection = require('../database/connection');

const crypto = require('crypto');

/\*\*

\* Função para gerar um id Randomico.

\*

\*/

function generateRandomicId() {

return crypto.randomBytes(4).toString('HEX');

};

/\*\*

\* Função responsável por obter uma ONG pelo Id.

\*

\* @param {\*} id

\*/

async function getOngById(id) {

return await connection('ongs')

.select('\*')

.where({ 'id': id })

.first();

};

module.exports = {

/\*\*

\* Controlador para listagem de todas as ongs cadastradas.

\*

\* @param {\*} request

\* @param {\*} response

\*/

async index(request, response) {

const ongs = await connection('ongs').select('\*');

return response.json(ongs);

},

/\*\*

\* Controlador para criação de uma nova Ong.

\*

\* @param {\*} request

\* @param {\*} response

\*/

async create(request, response) {

const { name, email, whatsapp, city, uf } = request.body;

//Gerando Primeiro Id Randomico

let id = generateRandomicId();

//Verificando se o id gerado já não existe no banco de dados

let ongWithIdExist = await getOngById(id);

//Loop de segurança que vai checar Equanto a Ong existir com aquele id de seleção,

//Ele vai ficar gerando novs Id's e checando novamente, dessa forma garantimos que

//O id será unico no Banco de dados.

while (ongWithIdExist) {

console.log('ONG com esse ID já existe. Solicitando novo ID');

id = generateRandomicId();

ongWithIdExist = await getOngById(id);

}

await connection('ongs').insert({

"id": id,

"name": name,

"email": email,

"whatsapp": whatsapp,

"city": city,

"uf": uf

})

return response.json({ "id": id });

},

}

**Aula 03**

**Conceitos aprendidos**

Componente : Um Componente é uma função que retorna HTML.

JSX: É o nome dado para a mistura de JavaScript com HTML.

Propriedades no React : São "atributos" passados para componentes dentro do React.

Estado no React: Informação que será mantida pelo componente, gerenciada pelo componente e essas informações são refletidas na interface.

Imutabilidade: Por uma questão de performace, o valor de estados não podem ser alterados de forma direta, eles precisam ser sobrepostos.