**IFTM - INSTITUTO FEDERAL DO TRIÂNGULO MINEIRO**
**ENGENHARIA DE COMPUTAÇÃO**

Rafael de Oliveira Évora
Yuri David Silva Duarte

# SEGUNDA ETAPA - TUTORIAL
# BACK EXPRESS.JS E FRONT REACT

**Uberaba**
**2024**

**1) Como criar as classes das entidades (Doador, Doacao) e enumerações (TipoSanguineo, RH) do sistema usando as tecnologias do seu grupo.**

Caso você ainda não tenha instalado o Joda, nós o instalaremos para lidar com datas, horários, períodos, durações, formatação e parsing de datas, entre outras funcionalidades relacionadas ao tempo.

npm install @js-joda/core

TUTORIAL

```
import TipoSanguineo from "../Enums/TipoSanguineo";
import FatorRH from "../Enums/FatorRH";

class Doador {

    constructor(

        private codigo: number,
        private nome: string,
        private cpf: string,
        private contato: string,
        private tipoSanguineo: TipoSanguineo,
        private fatorRH: FatorRH,
        private tipoRhCorretos: boolean

    ) { }

    public getCodigo() {

        return this.codigo;

    }

    public setCodigo(codigo: number) {

        this.codigo = codigo;

    }

    public getNome() {

        return this.nome;

    }
```

```
public setNome(nome: string) {

    this.nome = nome;

}

public getCpf() {

    return this.cpf;

}

public setCpf(cpf: string) {

    this.cpf = cpf

}

public getContato() {

    return this.contato;

}

public setContato(contato: string) {

    this.contato = contato;

}

public getTipoSanguine() {

    return this.tipoSanguineo;

}

public setTipoSanguineo(tipoSanguineo: TipoSanguineo) {

    this.tipoSanguineo = tipoSanguineo;

}
```

```
    public getFatorRH() {

        return this.fatorRH;

    }

    public setFatorRH(fatorRH: FatorRH) {

        this.fatorRH = fatorRH;

    }

    public getTipoRhCorretos() {

        return this.tipoRhCorretos;

    }

    public setTipoRhCorretos(tipoRhCorretos: boolean) {

        this.tipoRhCorretos = tipoRhCorretos

    }

    public static fromJson(json: Doador): Doador {
        return new Doador(
            json.codigo,
            json.nome,
            json.cpf,
            json.contato,
            json.tipoSanguineo,
            json.fatorRH,
            json.tipoRhCorretos
        )
    }

}

export default Doador;
```

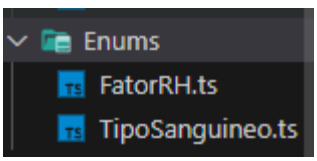
```
import { LocalDate, LocalTime } from '@js-joda/core';
```

```
class Doacao {

    constructor(
        private codigo: number,
        private data: LocalDate,
        private hora: LocalTime,
        private volume: number
    ) { }


    public getCodigo() {
        return this.codigo;
    }

    public setCodigo(codigo: number) {
        this.codigo = codigo;
    }

    public getDate(){
        return this.data
    }

    public setDate(data: LocalDate){
        this.data = data
    }

    public getHora(){
        return this.hora
    }

    public setHora(hora: LocalTime){
        this.hora = hora
    }

    public getVolume(){
        return this.volume;
    }

    public setVolume(volume: number){
        this.volume = volume;
    }

    public static fromJson(json: Doacao): Doacao {
```

```
        return new Doacao(
            json.codigo,
            json.data,
            json.hora,
            json.volume
        )
    }
}

export default Doacao;
```

```
class Dados {

    constructor(
        private texto: string,
        private inteiro: number,
        private booleano: boolean,
        private opcaoSelect: string,
        private opcaoRadio: string
    ) { }

    public getTexto() {
        return this.texto;
    }

    public setTexto(texto: string) {
        this.texto = texto;
    }

    public getInteiro() {
        return this.inteiro;
    }

    public setInteiro(inteiro: number) {
        this.inteiro = inteiro;
    }

    public getBooleano() {
        return this.booleano;
    }

    public setBooleano(booleano: boolean) {
```

```
        this.booleano = booleano;
    }

    public getOpcaoSelect() {
        return this.opcaoSelect;
    }

    public setOpcaoSelect(opcaoSelect: string) {
        this.opcaoSelect = opcaoSelect;
    }

    public getOpcaoRadio() {
        return this.opcaoRadio;
    }

    public setOpcaoRadio(opcaoRadio: string) {
        this.opcaoRadio = opcaoRadio;
    }

    public static fromJson(json: Dados): Dados {
        return new Dados(
            json.texto,
            json.inteiro,
            json.booleano,
            json.opcaoSelect,
            json.opcaoRadio
        )
    }
}

export default Dados;
```




```
enum FatorRH {

    positivo = "+",
    negativo = "-",

}
```

```
export default FatorRH;
```

```
enum TipoSanguineo {

    A = "A",
    B = "B",
    AB = "AB",
    O = "O",

}

export default TipoSanguineo;
```

**2) Como criar o BD do seu sistema para armazenar os dados das entidades do sistema (Doador, Doacao) usando as tecnologias do seu grupo. Caso as tecnologias do seu trabalho já venham com algum esquema de controle de migração de bancos de dados como o Flyway do Java use-o, mesmo que o sistema final fique com apenas uma migração.**

Utilizaremos o TypeORM que permite mapear objetos em bancos de dados relacionais. Ao definir entidades no TypeORM, você está criando representações das tabelas do banco de dados como classes no seu código.

Para instalá-lo vá na pasta do projeto e abra com o terminal. Execute o comando:
`npm install typeorm --save`

Também precisaremos do reflect-metadata:
`npm install reflect-metadata --save`

Agora precisaremos importar os tipos:
`npm install @types/node --save-dev`

Por fim instalaremos o driver correspondente ao banco de dados que utlizaremos (PostgreSQL):
`npm install pg --save`

```
import { Entity, Column, PrimaryColumn } from 'typeorm'
import { isTipoSanguineo, TipoSanguineo } from
"../Enums/TipoSanguineo";
import { isFatorRH, FatorRH } from "../Enums/FatorRH";

@Entity()
```

```
class Doador {

    @PrimaryColumn()
    private codigo: number;
    @Column()
    private nome: string;
    @Column()
    private cpf: string;
    @Column()
    private contato: string;
    @Column()
    private tipoSanguineo: TipoSanguineo;
    @Column()
    private fatorRH: FatorRH;
    @Column()
    private tipoRhCorretos: boolean;

    constructor(

        codigo: number,
        nome: string,
        cpf: string,
        contato: string,
        tipoSanguineo: TipoSanguineo,
        fatorRH: FatorRH,
        tipoRhCorretos: boolean

    ) {
        this.codigo = codigo;
        this.nome = nome;
        this.cpf = cpf;
        this.contato = contato;
        this.tipoSanguineo = tipoSanguineo;
        this.fatorRH = fatorRH;
        this.tipoRhCorretos = tipoRhCorretos;
    }

    public getCodigo() {

        return this.codigo;

    }
```

```
    public setCodigo(codigo: number) {
        this.codigo = codigo;
    }
    public getNome() {
        return this.nome;
    }
    public setNome(nome: string) {
        this.nome = nome;
    }
    public getCpf() {
        return this.cpf;
    }
    public setCpf(cpf: string) {
        this.cpf = cpf
    }
    public getContato() {
        return this.contato;
    }
    public setContato(contato: string) {
        this.contato = contato;
    }
    public getTipoSanguine() {
```

```
        return this.tipoSanguineo;

    }

    public setTipoSanguineo(tipoSanguineo: TipoSanguineo) {

        if (isTipoSanguineo(tipoSanguineo)) {
            this.tipoSanguineo = tipoSanguineo;
        }
    }

    public getFatorRH() {

        return this.fatorRH;
    }

    public setFatorRH(fatorRH: FatorRH) {
        if (isFatorRH(fatorRH)) {
            this.fatorRH = fatorRH;
        }
    }

    public getTipoRhCorretos() {

        return this.tipoRhCorretos;
    }

    public setTipoRhCorretos(tipoRhCorretos: boolean) {

        this.tipoRhCorretos = tipoRhCorretos
    }

    public static fromJson(json: Doador): Doador {
        return new Doador(
            json.codigo,
            json.nome,
            json.cpf,
            json.contato,
            json.tipoSanguineo,
            json.fatorRH,
            json.tipoRhCorretos
        )
```

```
    }

}

export default Doador;
```

```
import { Entity, Column, PrimaryColumn } from 'typeorm';
import { LocalDate, LocalTime } from '@js-joda/core';


@Entity()
class Doacao {

    @PrimaryColumn()
    private codigo: number;
    @Column()
    private data: LocalDate;
    @Column()
    private hora: LocalTime;
    @Column()
    private volume: number;

    constructor(
        codigo: number,
        data: LocalDate,
        hora: LocalTime,
        volume: number
    ) {
        this.codigo = codigo;
        this.data = data;
        this.hora = hora;
        this.volume = volume;
    }


    public getCodigo() {
        return this.codigo;
    }

    public setCodigo(codigo: number) {
        this.codigo = codigo;
    }
```

```
    public getDate() {
        return this.data
    }

    public setDate(data: LocalDate) {
        this.data = data
    }

    public getHora() {
        return this.hora
    }

    public setHora(hora: LocalTime) {
        this.hora = hora
    }

    public getVolume() {
        return this.volume;
    }

    public setVolume(volume: number) {
        this.volume = volume;
    }

    public static fromJson(json: Doacao): Doacao {
        return new Doacao(
            json.codigo,
            json.data,
            json.hora,
            json.volume
        )
    }
}

export default Doacao;
```