

# Revisão P2 – TP2 – 2025-2

## 1) Schema Mongoose

```
import mongoose from "mongoose";
const { Schema } = mongoose;

const UserSchema = new Schema({
  mail: { type: String,
    maxLength: [50, "O tamanho máximo do campo é 50 caracteres"],
    required: true }, // not null
  password: { type: String,
    minlength: 6, // menor quantidade de caracteres possível
    maxlength: 10, // maior quantidade de caracteres possível
    select: false, // não é listado em um select
    required: true } // not null
});

const SpentSchema = new Schema({
  user: { type: mongoose.Schema.Types.ObjectId, // chave estrangeira
    ref: 'User', // coleção de referência
    required: true }, // not null
  description: { type: String,
    maxlength: 30,
    required: true },
  value: { type: Number,
    required: true }
});
```

## 2) Anotação de tipos

- **string**: sequências de caracteres;
- **number**: números inteiros ou reais;
- **boolean**: somente os valores true e false;
- **null**: representa um valor nulo. Na prática é usado para referenciar a ausência de endereço de memória.
- **undefined**: representa um valor indefinido (não definido);
- **bignumber**: representa números inteiros maiores do que -(2<sup>53</sup> - 1) e menores do que (2<sup>53</sup> - 1).
- **any**: qualquer um dos tipos acima.

### Exemplos:

```
let nome:string = "Ana";
let idade:number = 25; //número inteiro
let peso:number = 59.9; //número real
let doador:boolean = true; //booleano
let fone:null = null; //endereço de memória
let cel:undefined = undefined; //não definido
```

```
let distancia:bigint = 20n; //o literal n é usado para indicar que o número  
é bigint  
let qualquer:any = 10; // qualquer valor
```

### **3) Coesão e Acoplamento**

- Coesão é ligada ao princípio da responsabilidade única, introduzido por Robert C. Martin.
- Uma classe deve ter apenas uma responsabilidade específica.
- Um código coeso tem classes e/ou métodos com uma única responsabilidade.
- Métodos devem fazer apenas uma tarefa específica, como imprimeSoma() deve apenas imprimir a soma e não calculá-la.
- Separar responsabilidades facilita a manutenção e alteração do código.
- Coesão é fundamental para a manutenção e evolução dos softwares.
  
- Acoplamento refere-se à dependência entre classes.
- Quanto maior a dependência, mais fortemente acopladas estão as classes.
- Classes fortemente acopladas dificultam o gerenciamento do sistema.
- Alterações em uma classe podem exigir mudanças em várias outras classes.

### **4) Validações Mongoose**

### **5) Funções**

### **6) MVC - Definições**