



Criando minha primeira API nodejs



*Pela ótica de um
Dev Java*



Hello, Devs
sou o Giovanni

Sou desenvolvedor e tenho +4 anos de experiência em engenharia de software, integração de sistemas, transformação digital e modernização de plataformas legadas para cloud AWS.

Minha Timeline

Educacional



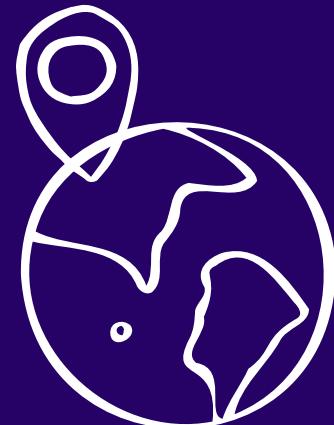
SENAI

fam

Profissional



Pessoal



Agenda de hoje

Diferenças entre Java x JS

Diferenças de ambientes de projetos

Diferenças de config de ambiente

Diferenças de criação de projetos

Diferenças de estrutura de projetos

Análise de código



Antes de começar!

Java

1

Java foi criada pela Sun Microsystems em 1995 e atualmente é mantida pela Oracle.

2

Definida como uma linguagem de programação orientada a objetos.

3

A principal função do Java é construir aplicações distribuídos em múltiplas plataformas, como mac, windows e linux.

4

Java Virtual Machine (JVM) é quem interpreta o bytecode para executar como um programa Java,

JavaScript

1

Jogada de marketing feita para chamar atenção para o JavaScript através da fama que o Java já tinha conquistado naquele momento.

2

A principal finalidade do JavaScript é adicionar interação às páginas, tornando possível a interação entre usuários e as aplicações Web e evitando que uma página seja meramente estática ao mostrar informações.

3

A principal vantagem do JavaScript é a possibilidade de uso em todas essas camadas da aplicação.

4

O JS, segundo a Pesquisa de Devs do Stack Overflow de 2022, é a mais popular no mundo.

Diferenças entre Java x JS

Linguagem interpretada ou compilada

- JavaScript é uma linguagem de programação interpretada, ou seja, pode ser lida e traduzida ao mesmo tempo que o programa está sendo executado. Enquanto isso, o Java é uma linguagem compilada: passa por uma JVM (Java Virtual Machine) de forma que ela é interpretada e traduzida.

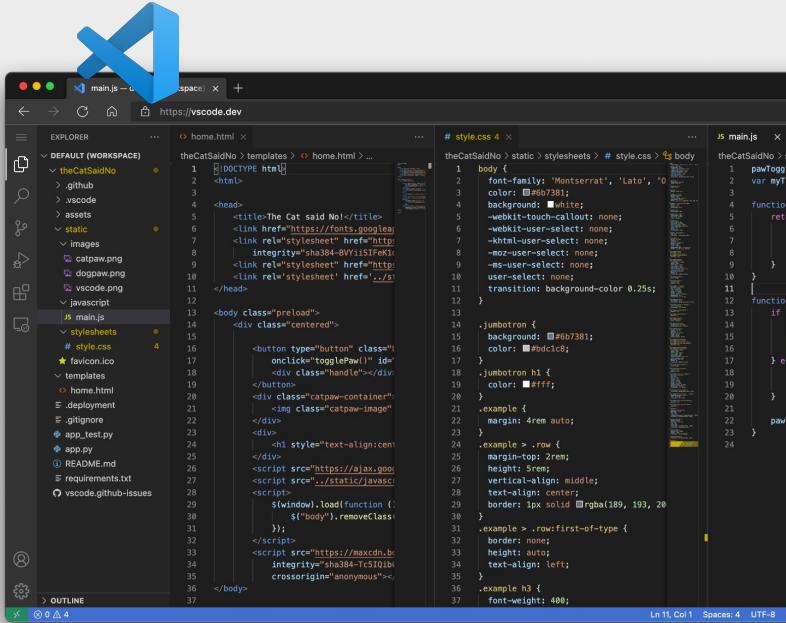
Tipagem

- Outra diferença é que JavaScript é uma linguagem weakly-typed, enquanto Java é uma linguagem strongly-typed. Estas definições estão relacionadas com quanto estreitas são as regras de escrita de ambas.

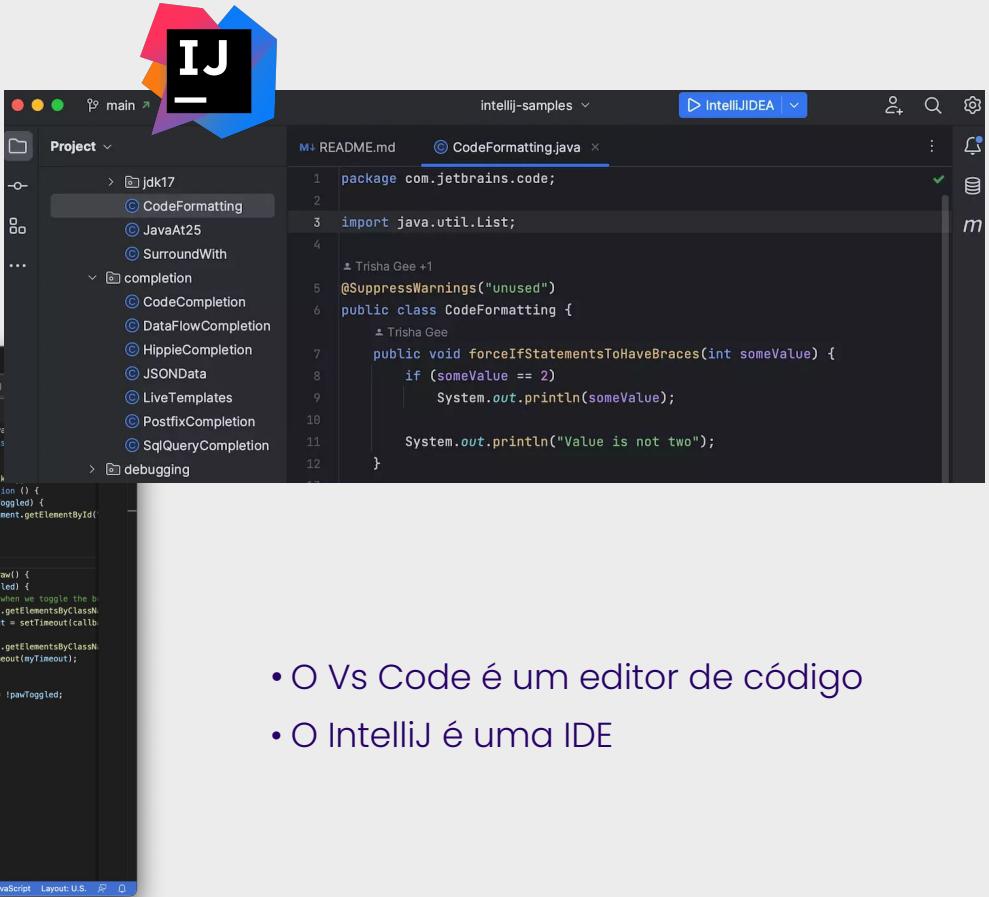
Backend ou frontend

- Outro ponto que as diferencia é o fato delas não serem utilizadas com o mesmo propósito. Enquanto Javascript é uma linguagem recomendada para apps, websites e front-end, Java é uma linguagem mais voltada para back-end e aplicações independentes de outro software e hardware.

Diferenças de ambientes de projetos



The screenshot shows the Visual Studio Code (VS Code) interface. On the left, the Explorer sidebar displays a file tree for a workspace named 'theCatSaidNo'. It includes files like .gitignore, .deployment, .github, .gitlab, app.py, README.md, requirements.txt, and several JavaScript and CSS files. The main editor area shows a multi-file view with 'main.js' and 'style.css'. The status bar at the bottom indicates the code is in 'JavaScript' mode.



The screenshot shows the IntelliJ IDEA interface. The top navigation bar shows 'IntelliJ-samples' and 'IntelliJIDEA'. The left sidebar shows a 'Project' view with a tree structure. The main editor area has tabs for 'README.md' and 'CodeFormatting.java'. The code editor shows Java code for 'CodeFormatting.java' with annotations like '@SuppressWarnings("unused")' and 'System.out.println' statements. The status bar at the bottom indicates the code is in 'Java' mode.

- O Vs Code é um editor de código
- O IntelliJ é uma IDE

Diferenças de config de ambiente

The image shows two windows from a Windows operating system's context menu properties dialog. Both windows have a title bar 'Propriedades de [Folder Name]' and three tabs: 'Versões Anteriores', 'Personalizado', and 'Geral'. The left window is for the 'Java' folder, and the right window is for the 'nodejs' folder.

Propriedade	Java	nodejs
Tipo:	Pasta de arquivos	Pasta de arquivos
Local:	C:\Program Files	C:\Program Files
Tamanho:	289 MB (303.787.747 bytes)	79,0 MB (82.917.492 bytes)
Tamanho em disco:	290 MB (304.259.072 bytes)	82,9 MB (87.023.616 bytes)
Contém:	416 Arquivos, 88 Pastas	2.176 Arquivos, 526 Pastas
Criado em:	sábado, 13 de maio de 2023, 15:27:14	domingo, 14 de maio de 2023, 01:23:56
Atributos:	<input checked="" type="checkbox"/> Somente leitura (arquivos da pasta) <input type="checkbox"/> Oculto <input type="checkbox"/> Avançados...	<input checked="" type="checkbox"/> Somente leitura (arquivos da pasta) <input type="checkbox"/> Oculto <input type="checkbox"/> Avançados...

- O Java tem 289 MB enquanto o JS tem 79 MB.
- O tamanho do disco do Java é de 290 MB já o do JS é de 83 MB.
- Java dividido em +400 arquivos e o JS em +2100 arquivos (node modules)

Diferenças de criação de projetos

```
PROBLEMAS SAÍDA CONSOLE DE DEPURAÇÃO TERMINAL
PS C:\Users\giova\OneDrive\Área de Trabalho\mentoriatech\curso\api-nodejs>
[Content da sessão restaurado de 14/05/2023 em 05:05:20]
O Windows PowerShell
Copyright (C) Microsoft Corporation. Todos os direitos reservados.

Instale o PowerShell mais recente para obter novos recursos e aprimoramentos! https://aka.ms/PSWindows

PS C:\Users\giova\OneDrive\Área de Trabalho\mentoriatech\curso\api-nodejs> npm run dev

> api-nodejs-typescript@1.0.0 dev
O Windows PowerShell
Copyright (C) Microsoft Corporation. Todos os direitos reservados.

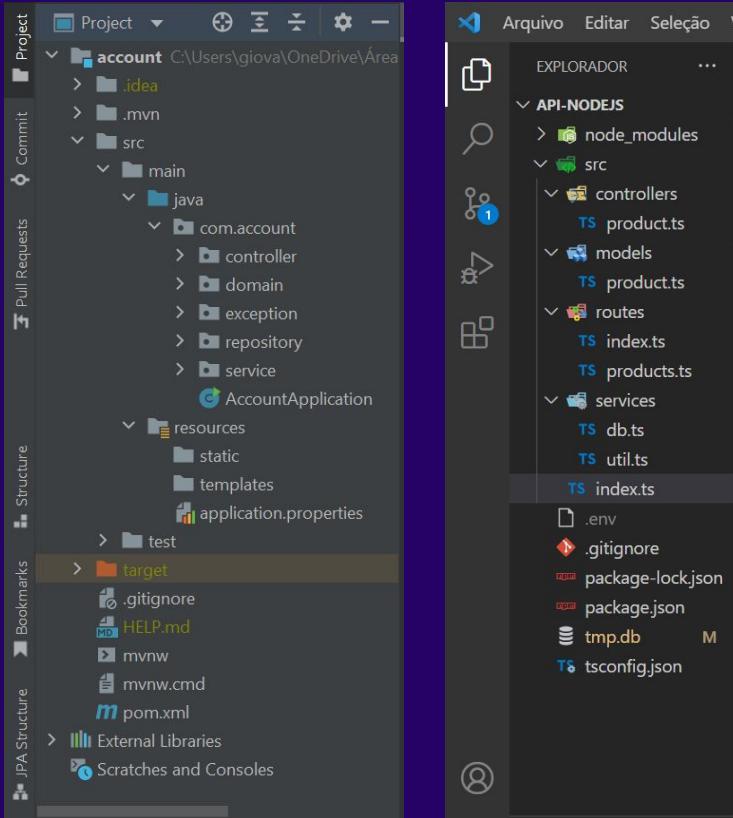
Instale o PowerShell mais recente para obter novos recursos e aprimoramentos! https://aka.ms/PSWindows

PS C:\Users\giova\OneDrive\Área de Trabalho\mentoriatech\curso\api-nodejs>
```

The screenshot shows the Spring Initializr website interface for creating a new Spring Boot project. The configuration includes:

- Project:** Maven selected over Gradle - Groovy, Gradle - Kotlin, and Groovy.
- Language:** Java selected over Kotlin and Groovy.
- Spring Boot:** Version 3.0.6 selected over 3.1.0 (SNAPSHOT), 3.1.0 (RC2), 3.1.0 (M2), 3.0.7 (SNAPSHOT), 2.7.12 (SNAPSHOT), and 2.7.11.
- Project Metadata:**
 - Group: com.example
 - Artifact: demo
 - Name: demo
 - Description: Demo project for Spring Boot
 - Package name: com.example.demo
 - Packaging: Jar selected over War
- Dependencies:**
 - Spring Boot DevTools** (DEVELOPER TOOLS): Provides fast application restarts, LiveReload, and configurations for enhanced development experience.
 - Spring Web** (WEB): Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.
 - MySQL Driver** (SQL): MySQL JDBC driver.
 - Lombok** (DEVELOPER TOOLS): Java annotation library which helps to reduce boilerplate code.

Diferenças de estrutura de projeto



- No java realizamos a estrutura de projetos com a extensão .class onde utilizamos a OOP tradicional.
- No Js realizamos a criação da estrutura com a extensão .TS utilizando a OOP mais flexível

Análise de código - Model



Arquivo Editar Seleção Ver Acessar Executar Terminal Ajuda product.ts - api-nodejs - Visual Studio Code

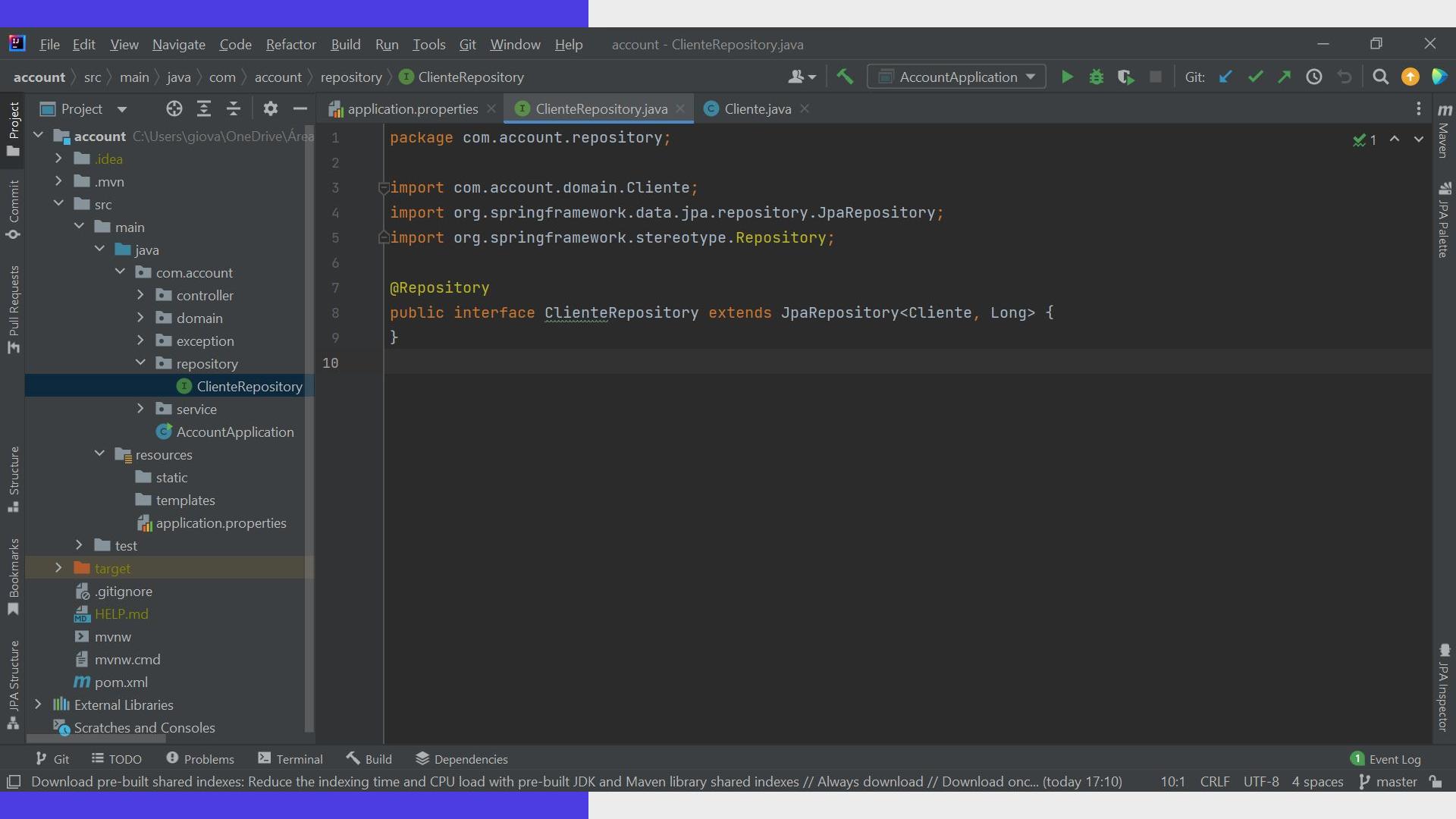
EXPLORADOR ... TS index.ts TS product.ts ...controllers TS product.ts ...models X TS products.ts

src > models > TS product.ts > [?] deleteProducts

```
1 import { dbQuery, dbQueryFirst } from "../services/db";
2
3 export type Product = {
4     id: number;
5     name: string;
6     price: number;
7 }
8
9 export const insertProduct = async (product: Product) => {
10     await dbQuery(`INSERT INTO product (name, price) VALUES(?, ?)`, [product.name, product.price]);
11     let retorno = await dbQuery(`SELECT seq AS Id From sqlite_sequence WHERE name= 'product'`);
12     return retorno[0].Id as number | undefined;
13 }
14
15 const listProducts = async () => {
16     const retorno = await dbQuery(`SELECT * FROM product`);
17     return retorno as Product[];
18 }
19
20 const getProducts = async (id: number) => {
21     const retorno = await dbQueryFirst(`SELECT * FROM product WHERE id=?`, [id]);
22     return retorno as Product | undefined;
23 }
24
25 const deleteProducts = async (id: number) => {
26     await dbQueryFirst(`DELETE FROM product WHERE id=?`, [id]);
27 }
```

The screenshot shows a Java development environment with the following details:

- Project Structure:** The project is named "account" and is located at `C:\Users\giova\OneDrive\Área de Trabalho\account`. It contains a `src` folder with `main`, `java`, and `resources` subfolders. The `java` folder contains packages `com.account`, `controller`, and `domain`. The `domain` package contains a class named `Cliente`. Other files in the `src/main/java` directory include `exception`, `repository`, `service`, and `AccountApplication`.
- Editor:** The `Cliente.java` file is open in the editor. The code defines a class `Cliente` that implements `Serializable`. It includes annotations `@NoArgsConstructor`, `@Id`, `@GeneratedValue(strategy = GenerationType.IDENTITY)`, and `@CreationTimestamp`. The class has private fields `id`, `tipoTransacao`, `valor`, `tipo`, and `data`.
- Toolbars and Status Bar:** The top bar includes standard menu items like File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, Git, Window, Help, and a tab for "account - Cliente.java". The bottom bar shows tabs for "application.properties" and "Cliente.java". The status bar at the bottom provides build information: "Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK and Maven library shared indexes // Always download // Download onc... (today 17:10)" and "38:1 CRLF UTF-8".



Análise de código - Conexão DB



A screenshot of the Visual Studio Code interface. The top bar shows the menu: Arquivo, Editar, Seleção, Ver, Acessar, Executar, Terminal, Ajuda, followed by the file path db.ts - api-nodejs - Visual Studio Code. The left sidebar has icons for Explorer, Search, Problems (with 1), and others. The Explorer pane shows a project structure under 'API-NO...': node_modules, src (controllers, models, routes, services), and files like index.ts, product.ts, products.ts, db.ts, util.ts, and tsconfig.json. The db.ts file is selected in the Explorer. The main editor pane displays the code for db.ts:

```
src > services > TS db.ts > dbQueryFirst
1 import sqlite3 from "sqlite3";
2
3 const DATABASE_FILE = process.env.DATABASE_FILE;
4 if(!DATABASE_FILE)
5     throw new Error("DATABASE_FILE não informado");
6
7 export const openConnection = () => {
8     let db = new sqlite3.Database(DATABASE_FILE);
9     return db;
10}
11
12 export const dbQueryFirst = async (query: string, params?: any[]) => {
13     const retorno = await dbQuery(query, params);
14     return retorno[0];
15}
16
17 export const dbQuery = (query: string, params?: any[]) => {
18     let db = openConnection();
19     return new Promise<any[]>((resolve, reject) => {
20         db.all(query, params, (err, rows) => {
21             if(err)
22                 reject(err);
23             else
24                 resolve(rows);
25         })
26     })
27     .finally(() => {
28         db.close();
29     })
30}
```

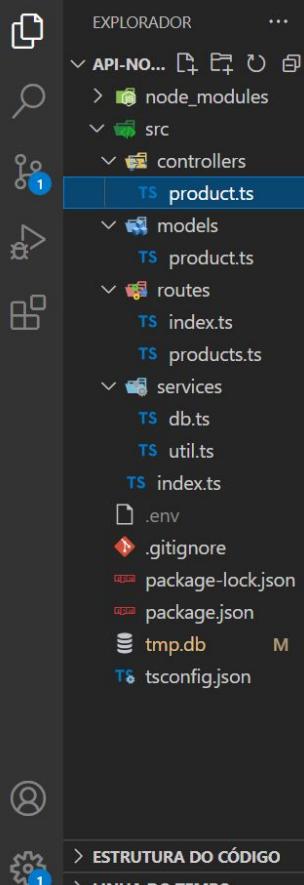
The screenshot shows a Java development environment with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, Git, Window, Help.
- Project Bar:** account - application.properties
- Toolbars:** Standard toolbar with icons for New, Open, Save, Print, etc.
- Left Sidebar:** Project, Commit, Pull Requests, Structure, Bookmarks, JPA Structure.
- Central Area:** Shows the `application.properties` file content:

```
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect
spring.datasource.url=jdbc:mysql://localhost/teste?serverTimezone=GMT-6
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.datasource.username=root
spring.datasource.password=
```
- Right Sidebar:** Maven panel with 7 errors and 1 warning.
- Bottom Navigation:** Git, TODO, Problems, Terminal, Build, Dependencies.
- Bottom Status:** Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK and Maven library shared indexes // Always download // Download o... (today 17:10), 6:25, LF, ISO-8859-1, 4 spaces, master, Event Log.

Análise de código Controller & Rotas







EXPLORADOR

...

TS index.ts

TS product.ts

TS products.ts X

API-NO...

...



> node_modules

src

controllers

TS product.ts

models

TS product.ts

routes

TS index.ts

TS products.ts

services

TS db.ts

TS util.ts

TS index.ts

.env

.gitignore

package-lock.json

package.json

tmp.db M

tsconfig.json



> ESTRUTURA DO CÓDIGO

> LINHA DO TEMPO

master* 0 0 0 0

```
src > routes > TS products.ts > ...
1  import { Router } from "express";
2  import { productController } from "../controllers/product";
3
4  const productRouter = Router();
5  productRouter.post('/', productController.insertProduct);
6  productRouter.get('/', productController.listProducts);
7  productRouter.get('/:id', productController.getProducts);
8  productRouter.delete('/:id', productController.deleteProducts);
9  productRouter.put('/:id', productController.updateProducts);
10
11 export{
12   productRouter
13 }
```

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help APIClient-master - ClientController.java

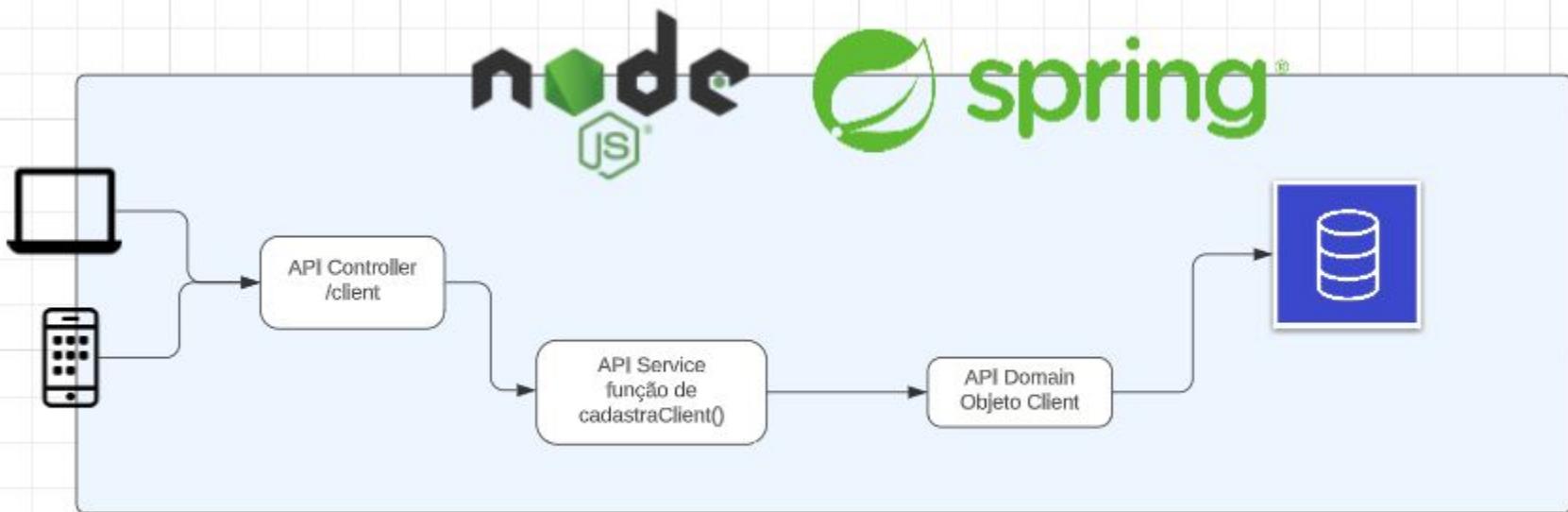
APIClient-master > src > main > java > com > cliente > controller > ClientController > update ClientApplication

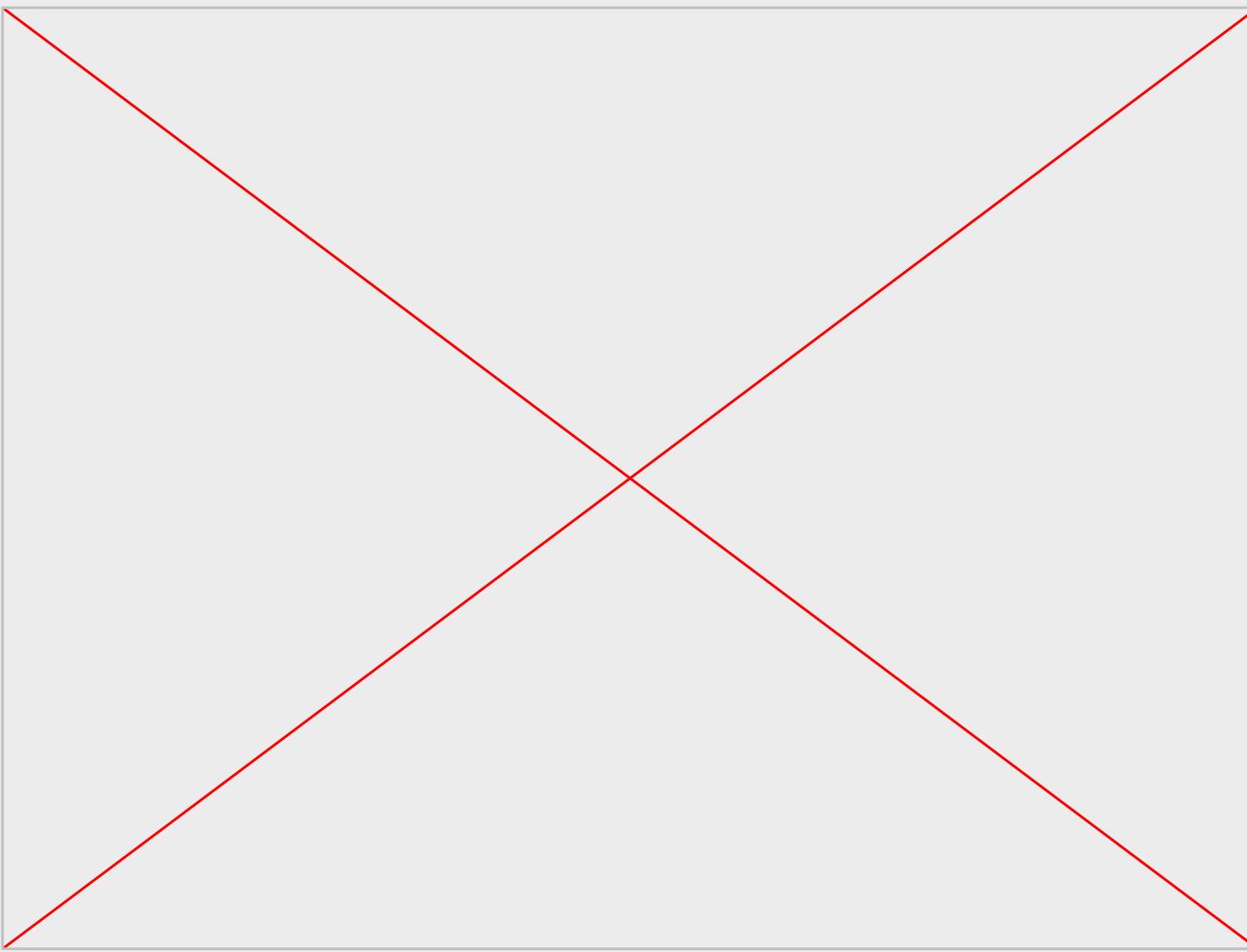
Project .idea .mvn src main java com.cliente controller ClientController.java

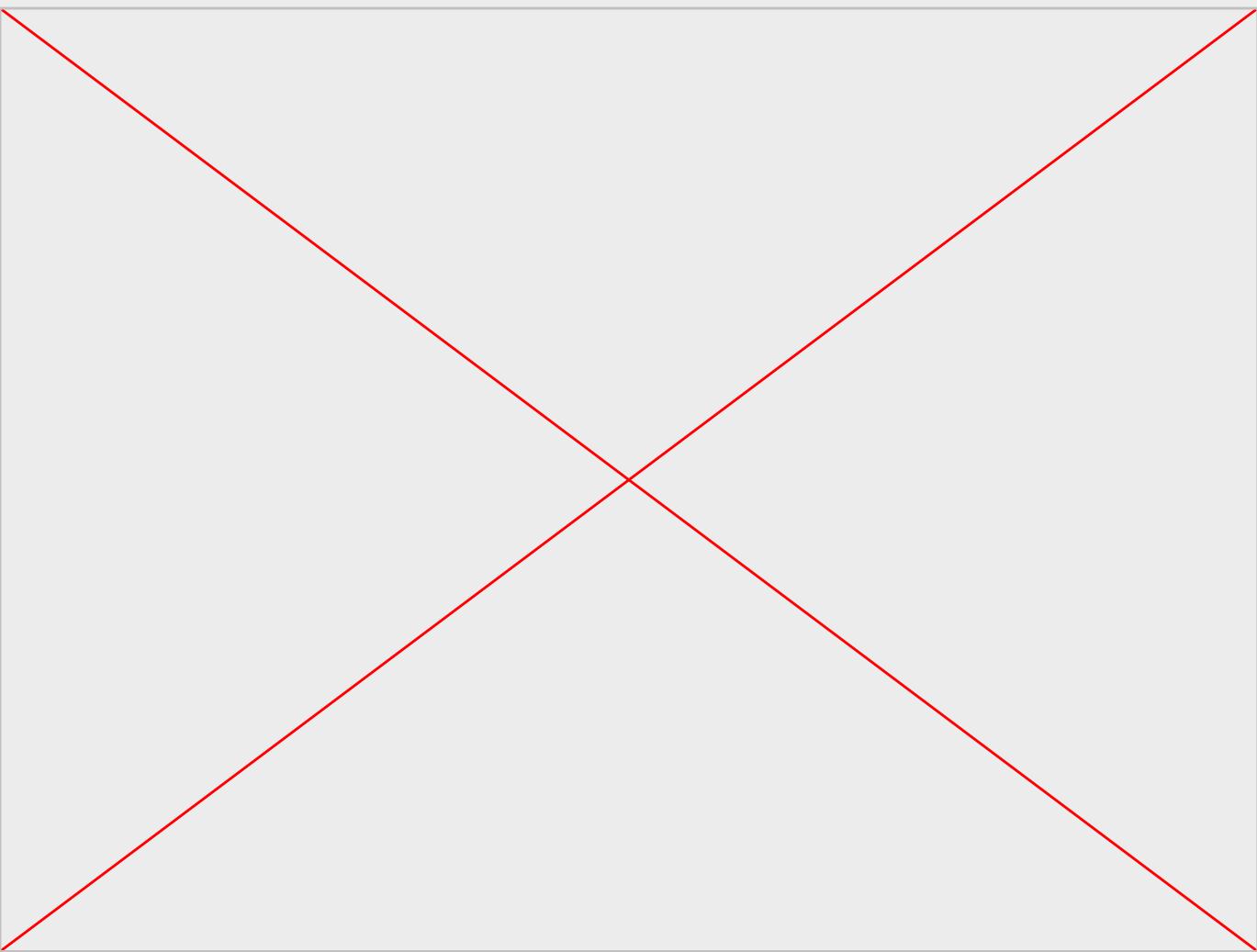
```
package com.cliente.controller;
import ...
@RestController
@RequestMapping("/client")
public class ClientController {
    @Autowired
    private ClientService clientService;
    @GetMapping("/{id}")
    public ResponseEntity<Client> findById(@PathVariable Long id){
        Client client = this.clientService.findById(id);
        return ResponseEntity.ok().body(client);
    }
}
```

Demo

Arquitetura simples da API







Considerações finais!

JAVA

Robusto e
tradicional

JS

Flexível e
moderno

JS

Melhor performance
Curva de aprendizado

JAVA

Mais seguro
Menos personalizável

Obrigadx!

Dúvidas?



Redes Sociais

Instagram



Linkedin

