**Guia De Desenvolvimento**

**Spring MVC**

**Thymeleaf**

**Spring Boot**

**Git**

**2023**

**Parte 1**

# Introdução

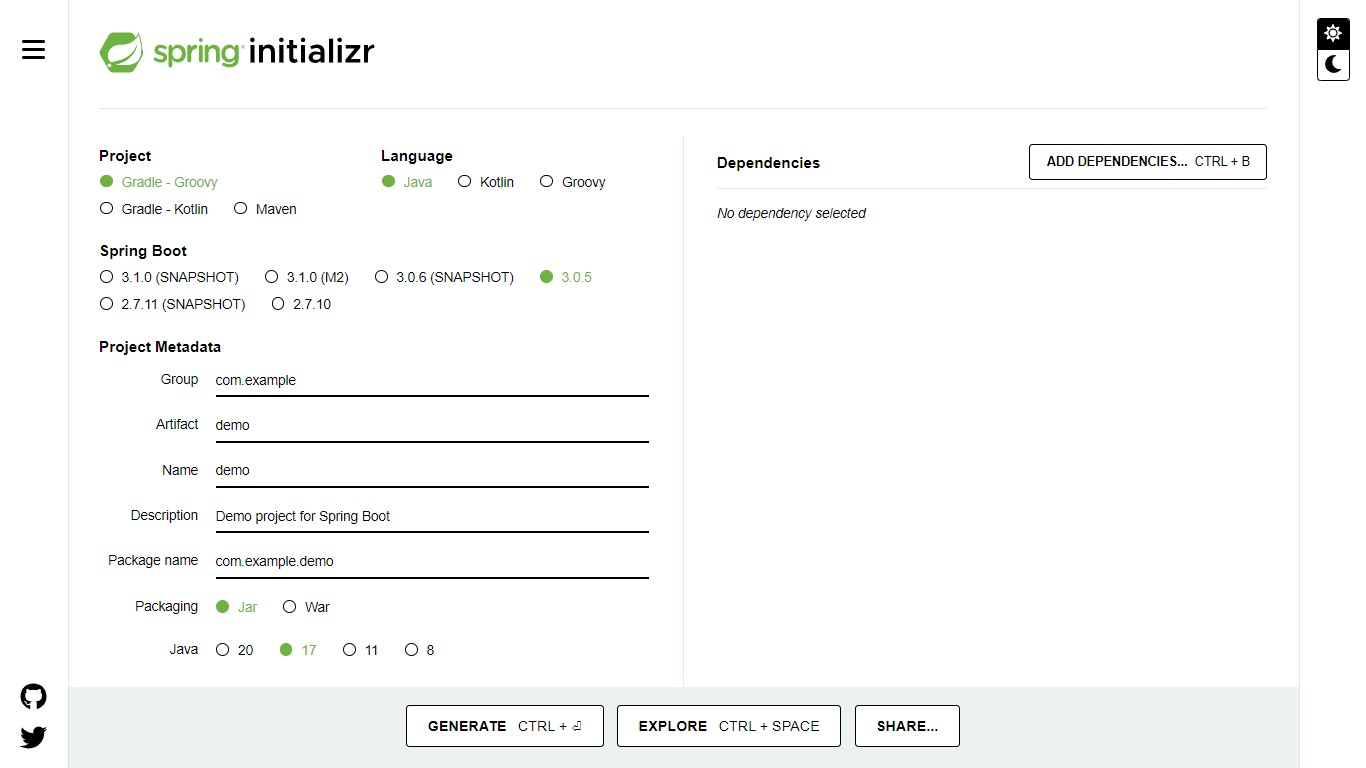
Neste documento teremos o passo-a-passo para a construção de uma aplicação web utilizando as ferramentas Spring MVC, Thymeleaf, Spring Boot e Git.

Este guia se destina a dar suporte as aulas de LIP1 do segundo módulo do curso de informática noturno do ITB Belval, portanto, algumas explicações mais detalhadas serão omitidas pois serão apresentadas pelo professor durante a aula.

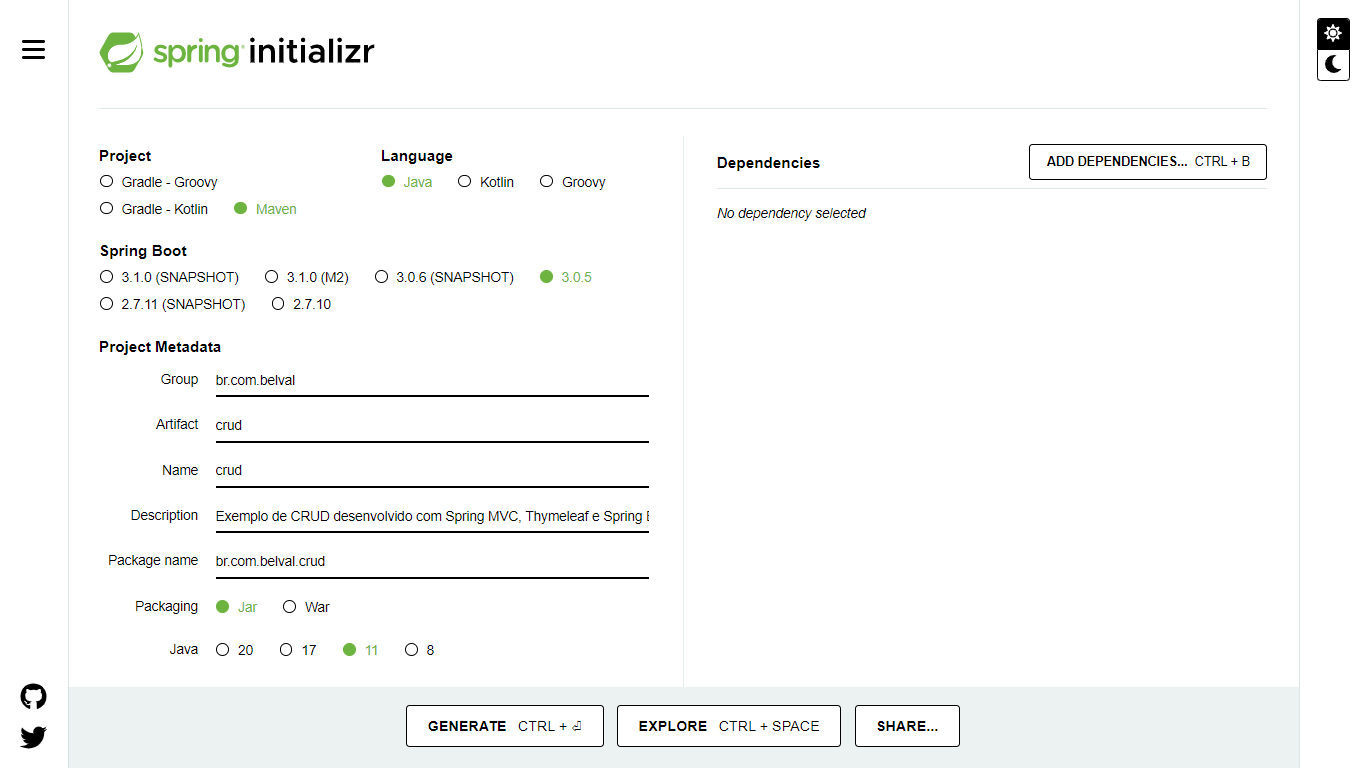
Parte 1

# Criar novo projeto

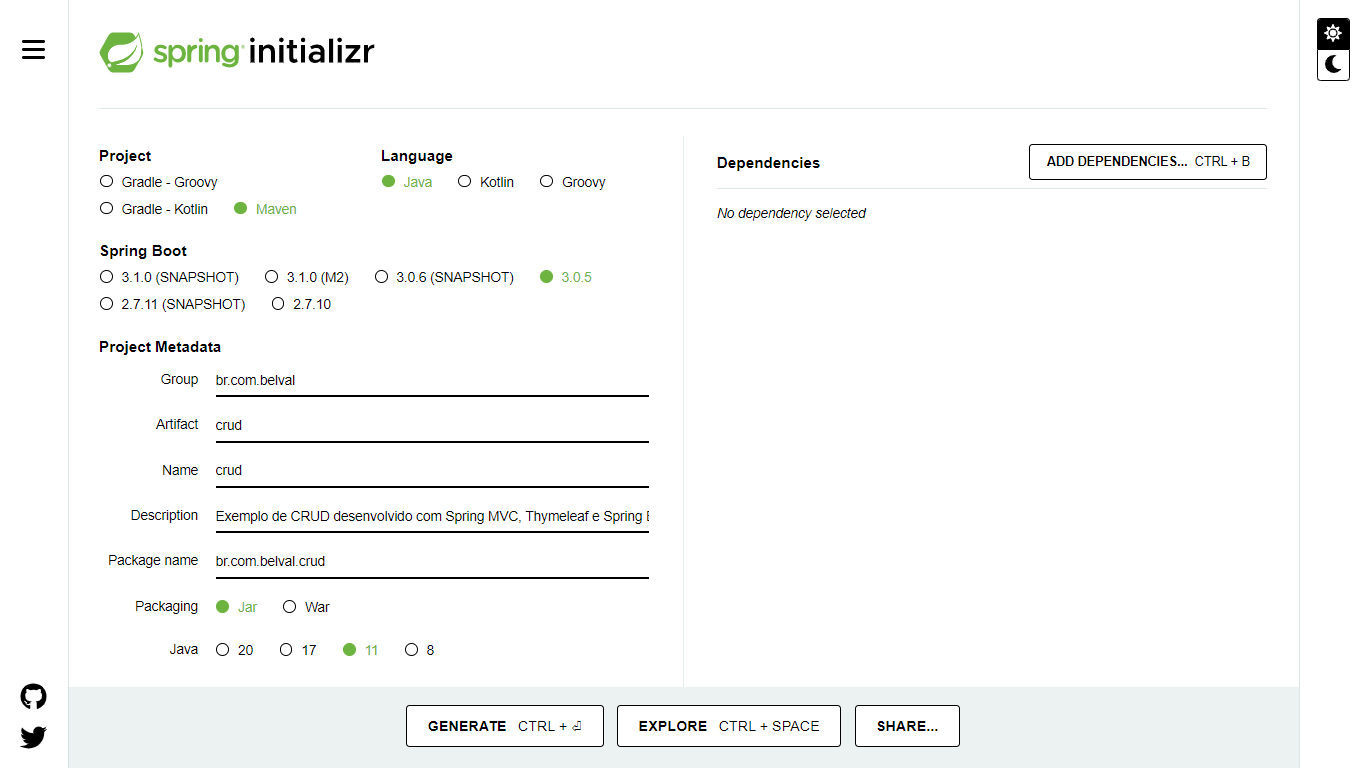
1 – Acesse a página <https://start.spring.io/>



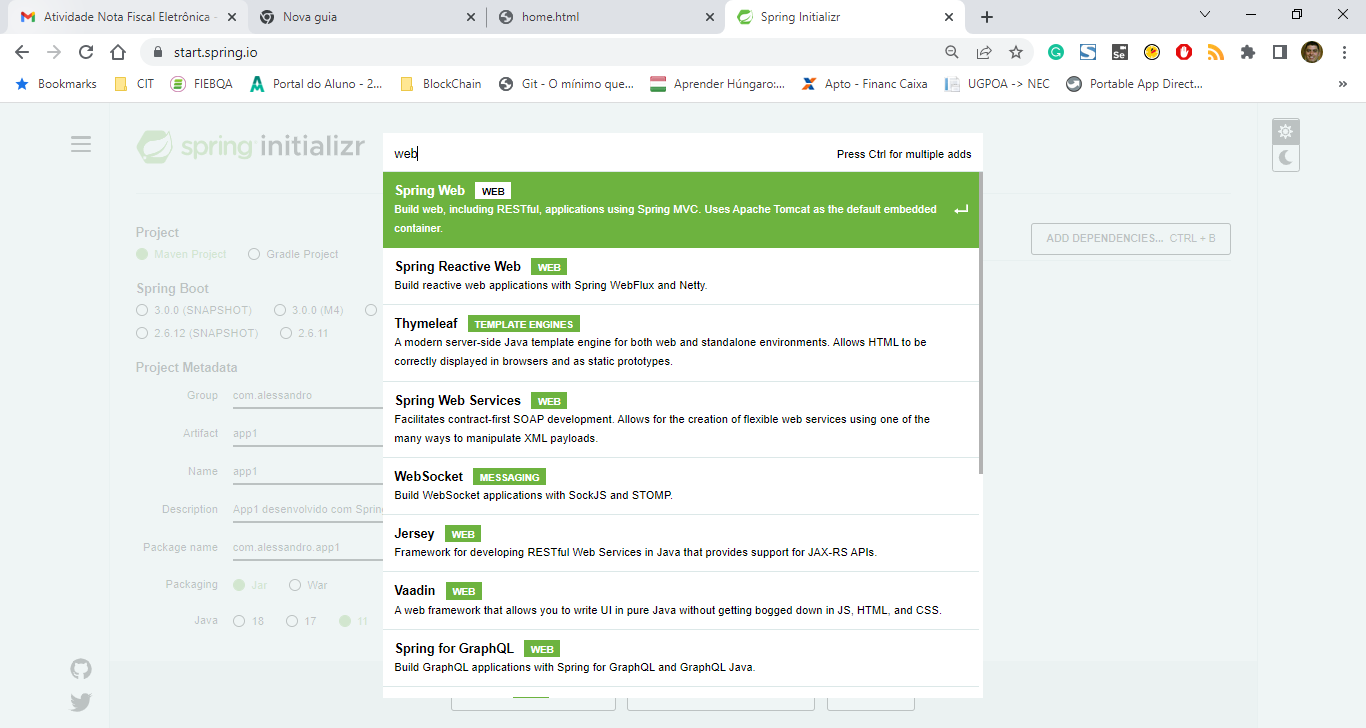
2 – No campo “Project” selecione a opção “Maven”, no campo “Group” preencha com “br.com.belval”, em “Artifact” preencha com “crud”, em “Description” preencha com “Exemplo de CRUD desenvolvido com Spring MVC, Thymeleaf e Spring Boot”. Marque a opção “11” para a versão do java.



3 – Clique no botão “ADD DEPENDENCIES... CTRL + B”

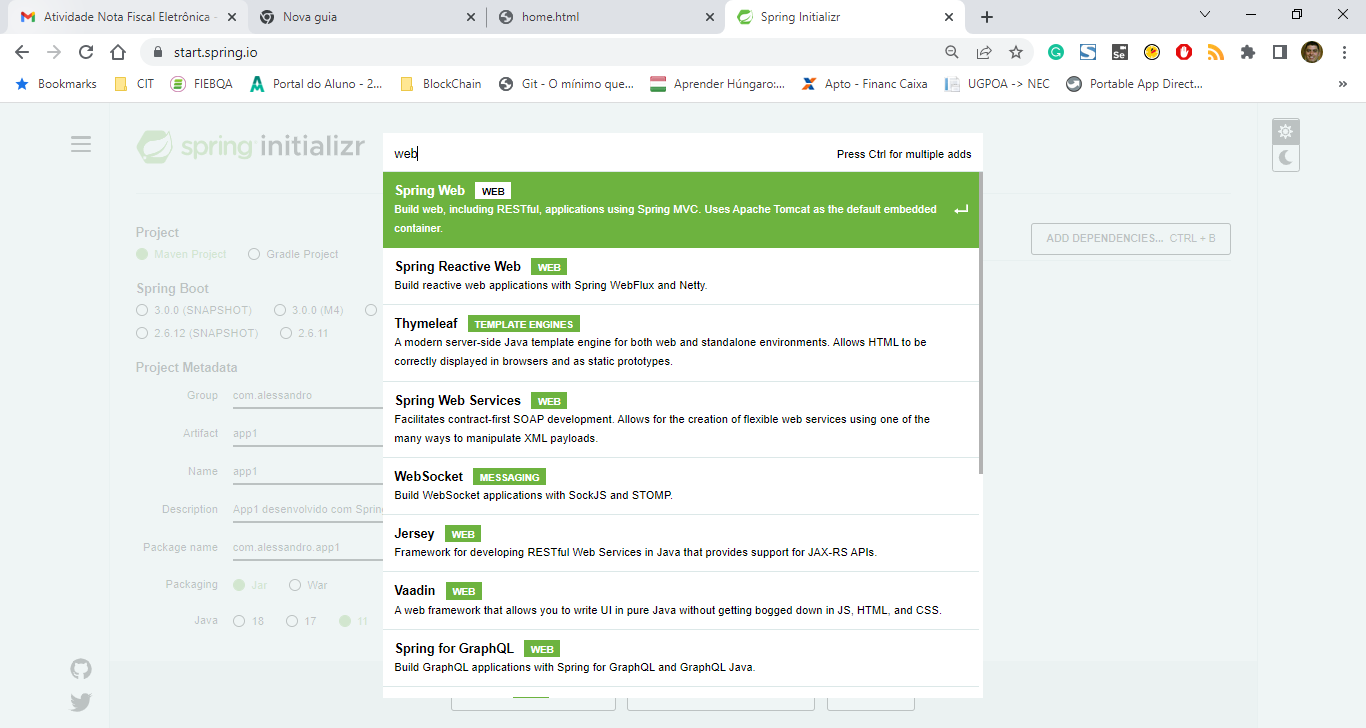


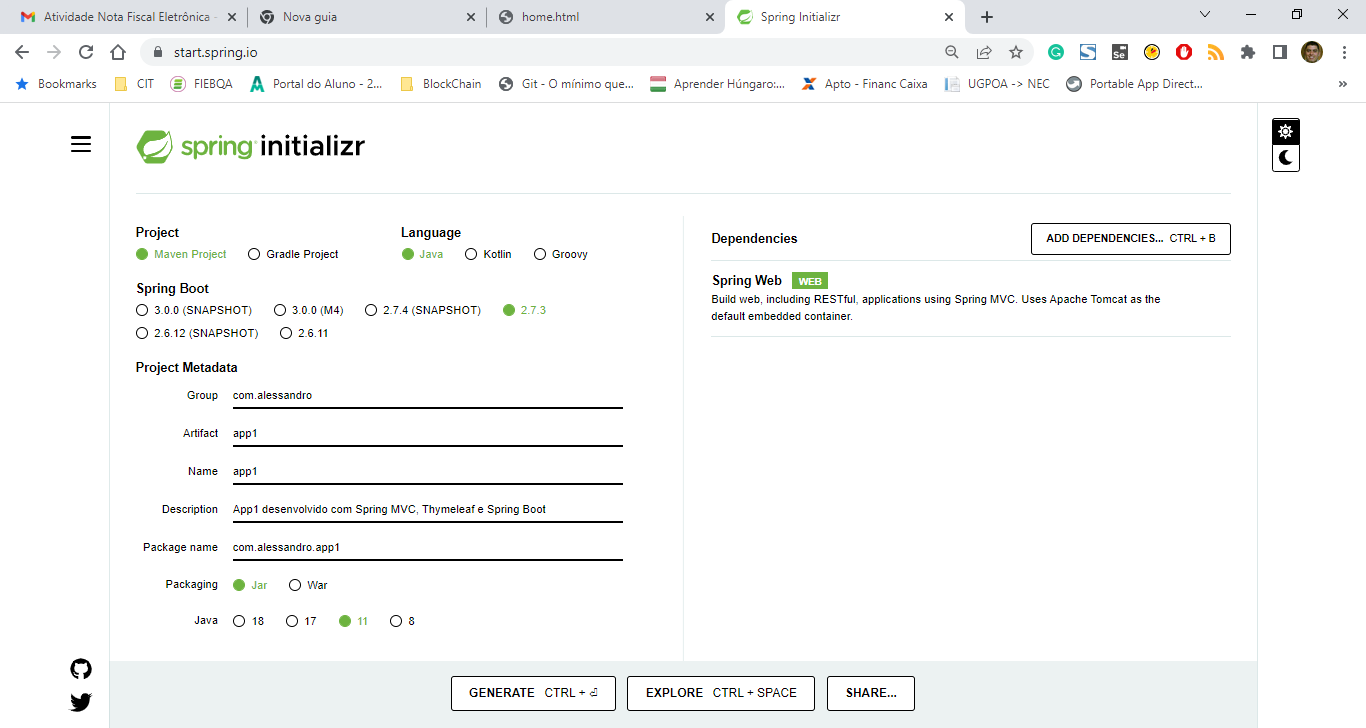
4 – Na próxima tela, preencha o campo de pesquisa com o termo “web”. Provavelmente surgirá uma opção “Spring Web” na primeira opção da lista. Clique nesse item para adicionar o framework Spring MVC ao projeto.



Campo de pesquisa

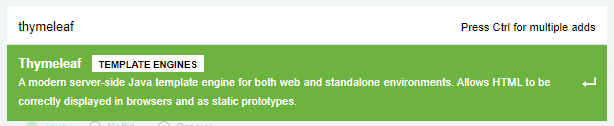
Spring MVC

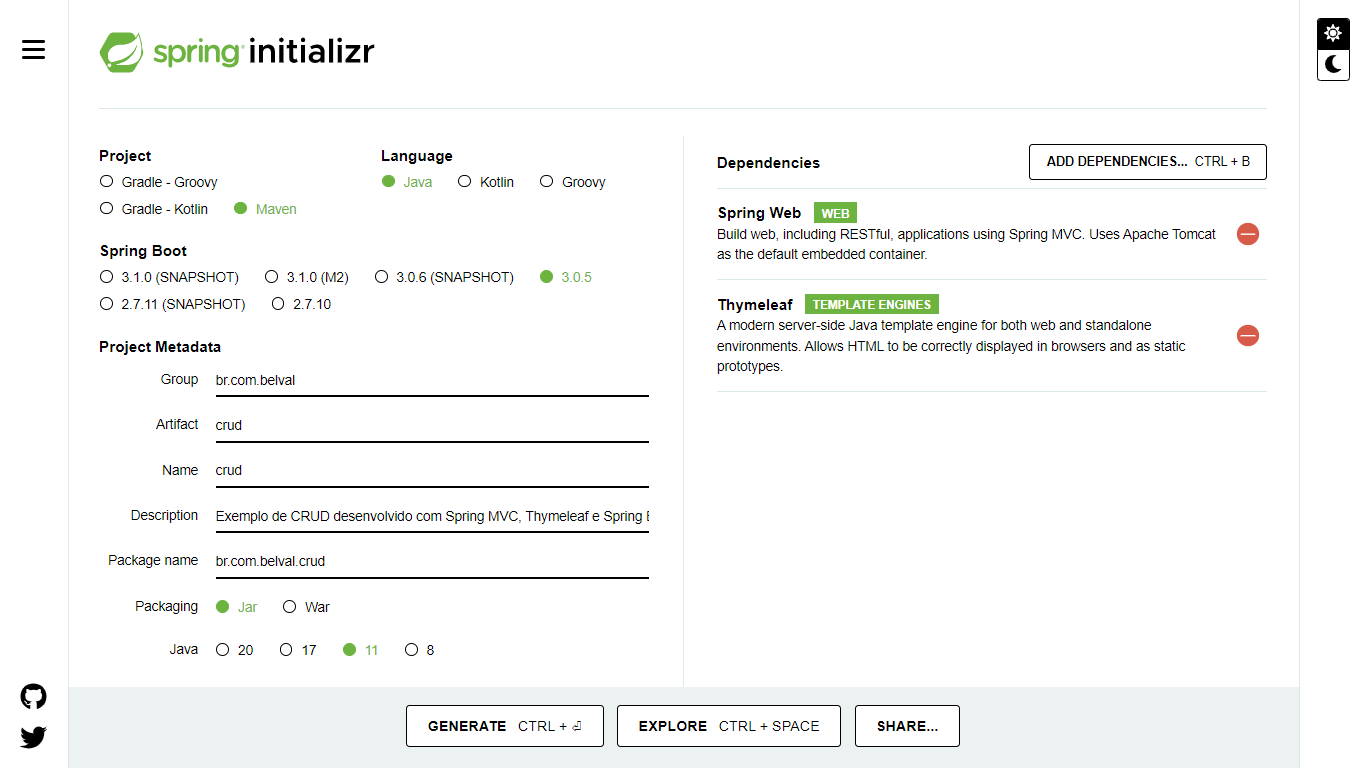




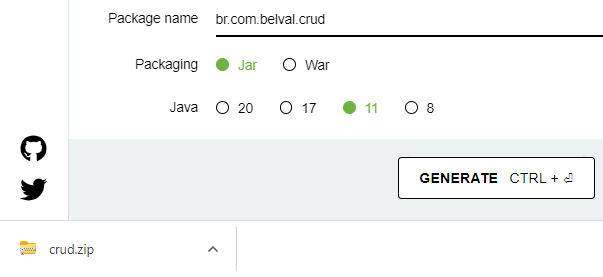
Dependência do Spring MVC adicionada

5 – Em seguida, clique novamente em “ADD DEPENDENCIES...CTRL + B” e preencha o campo de pesquisa com o termo “thymeleaf”. Provavelmente surgirá uma opção “Thymeleaf” como única opção da lista. Clique nesse item para adicionar o motor de “templates”(modelos).

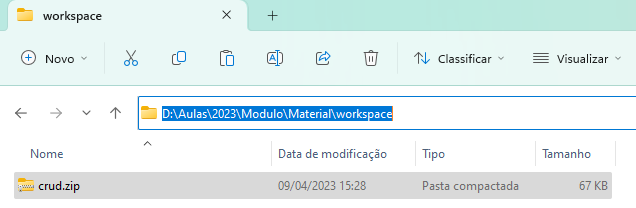




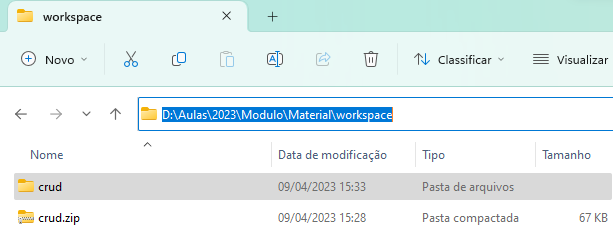
6 – Clique em “Generate CTRL + <ENTER>” para baixar o projeto.



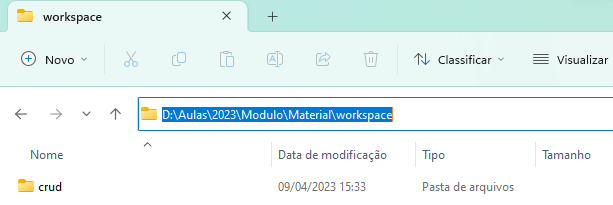
7 – Copie o \*.zip baixado para sua pasta “.../workspace”

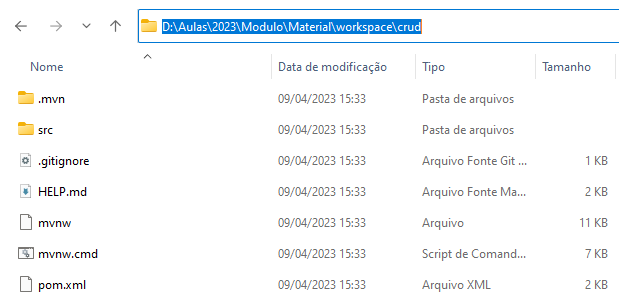


8 – Dentro do arquivo \*.zip há uma pasta “crud”, extraia esta pasta para dentro do diretório workspace.



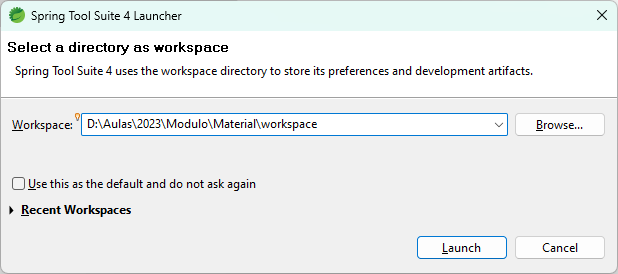
9 – Como efeito, teremos uma pasta e um arquivo \*.zip “crud” dentro de workspace. Apague o arquivo \*.zip ficando apenas com a pasta “crud” dentro do diretório workspace.



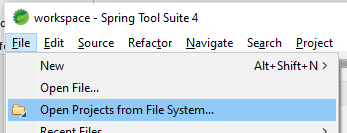


# Abrir projeto no Spring Tool Suite

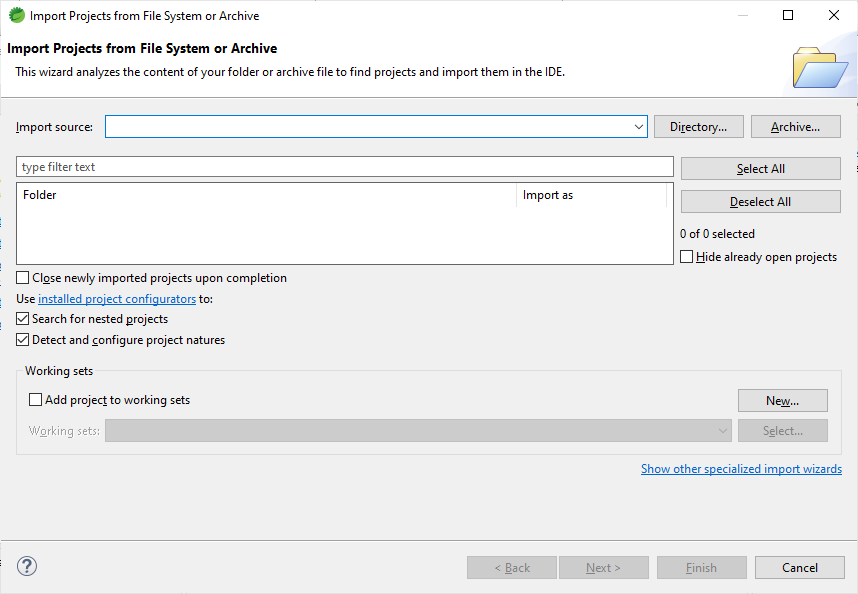
1 – Abra o Spring Tool Suite (STS) clicando no atalho na área de trabalho, preencha o campo “Workspace” com o caminho de seu diretório workspace e, por fim, clique em “Launch”.



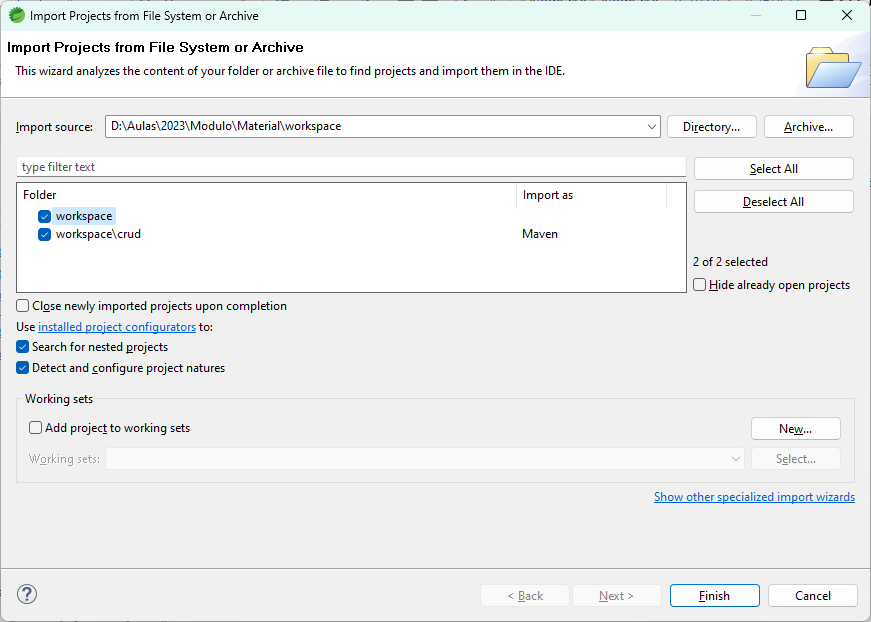
2 – Clique File > Open Projects from File System...



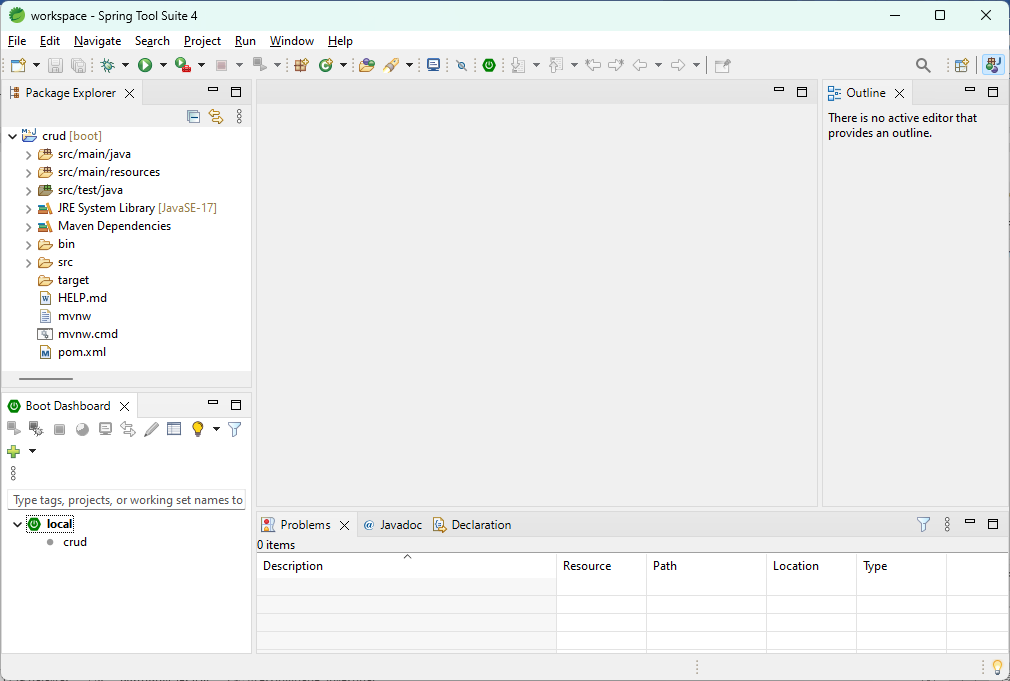
3 – Na tela seguinte clique em “Directory...” e selecione a pasta “.../workspace”



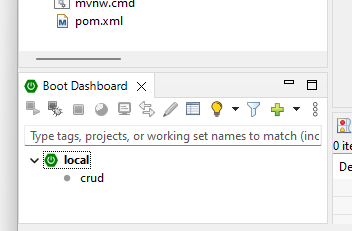
4 – Uma vez que o projeto crud esteja selecionado, clique em “Finish”.



Aguarde o Maven (gerenciador de dependências) buscar as dependências do projeto e o projeto deverá aparecer como a figura abaixo, sem nenhuma indicação de erro.

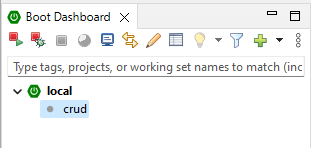


Observe que, caso a importação das dependências tenha sido bem sucedida, nosso projeto crud, deverá aparecer “dentro” de “local” na View “Boot Dashboard”.

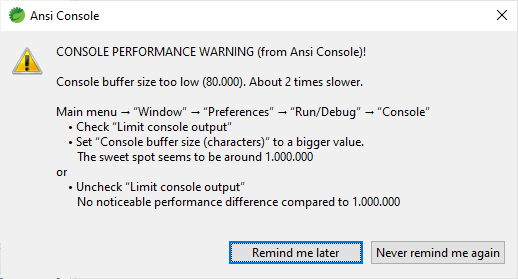


# Executar o projeto

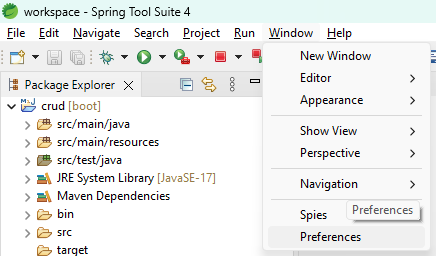
1 – Uma das formas para executar um projeto baseado no Spring Boot é, através da IDE STS, na View “Boot Dashboard”, selecionar o projeto e clicar no botão “Start or Restart”.

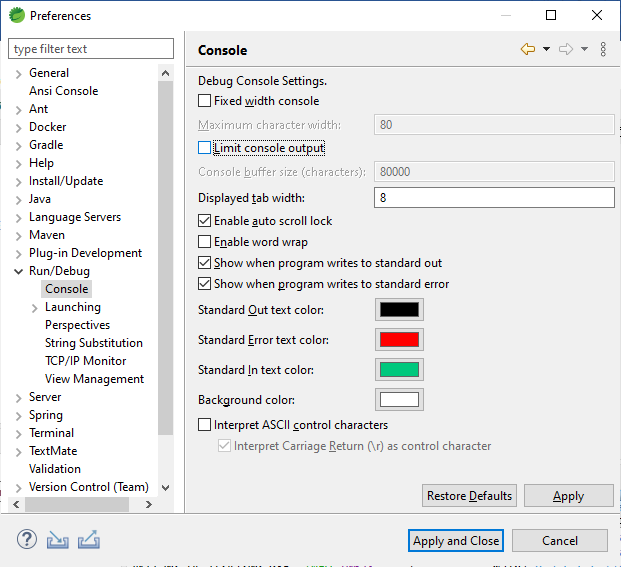


2 - Na primeira vez que você tentar executar seu projeto pode aparecer a tela a seguir, que alerta sobre problemas de performance em função do espaço reduzido do console. Clique em “Never remind me again”.

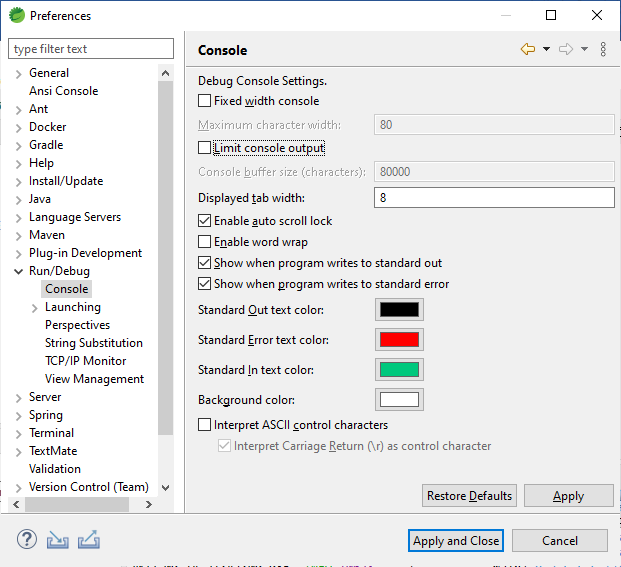


3 – Caso seu ambiente esteja muito lento você pode seguir a recomendação dessa tela e ir no menu Window > Preferences > Run/Debug > Console e desmarcar a opção “Limit console output”.

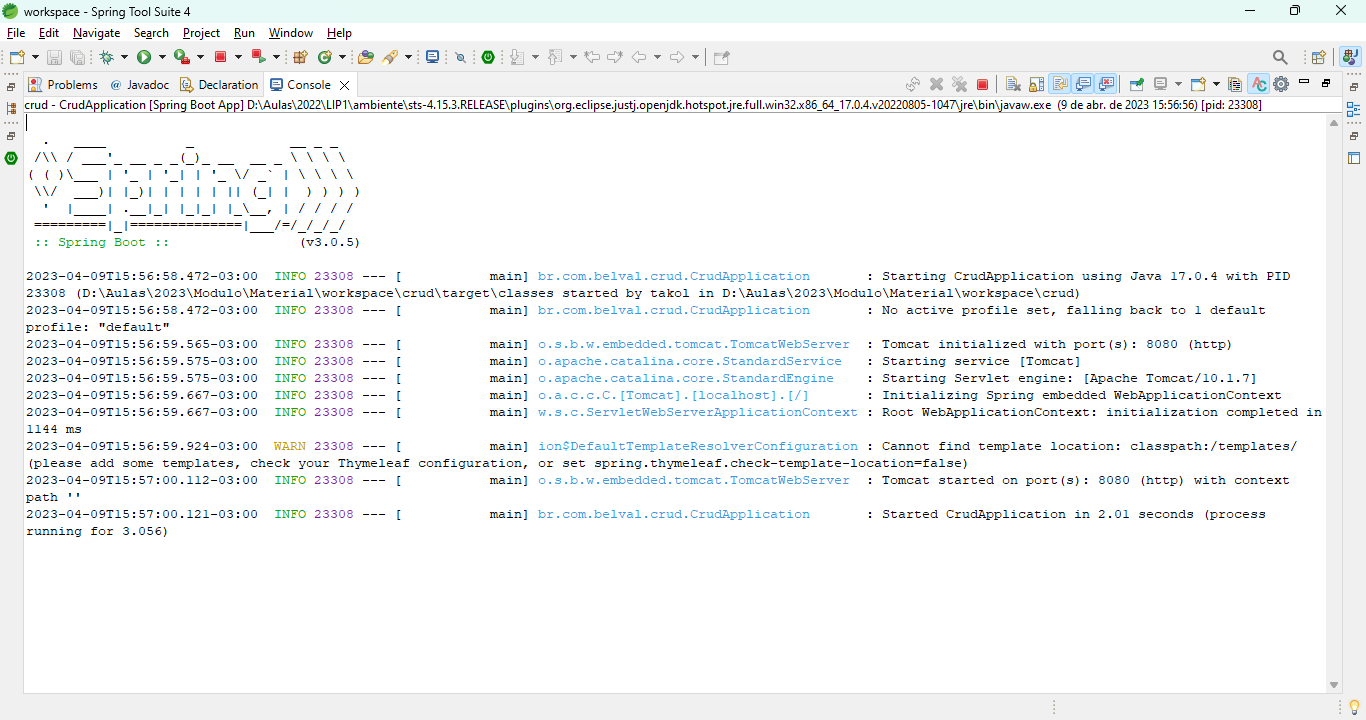


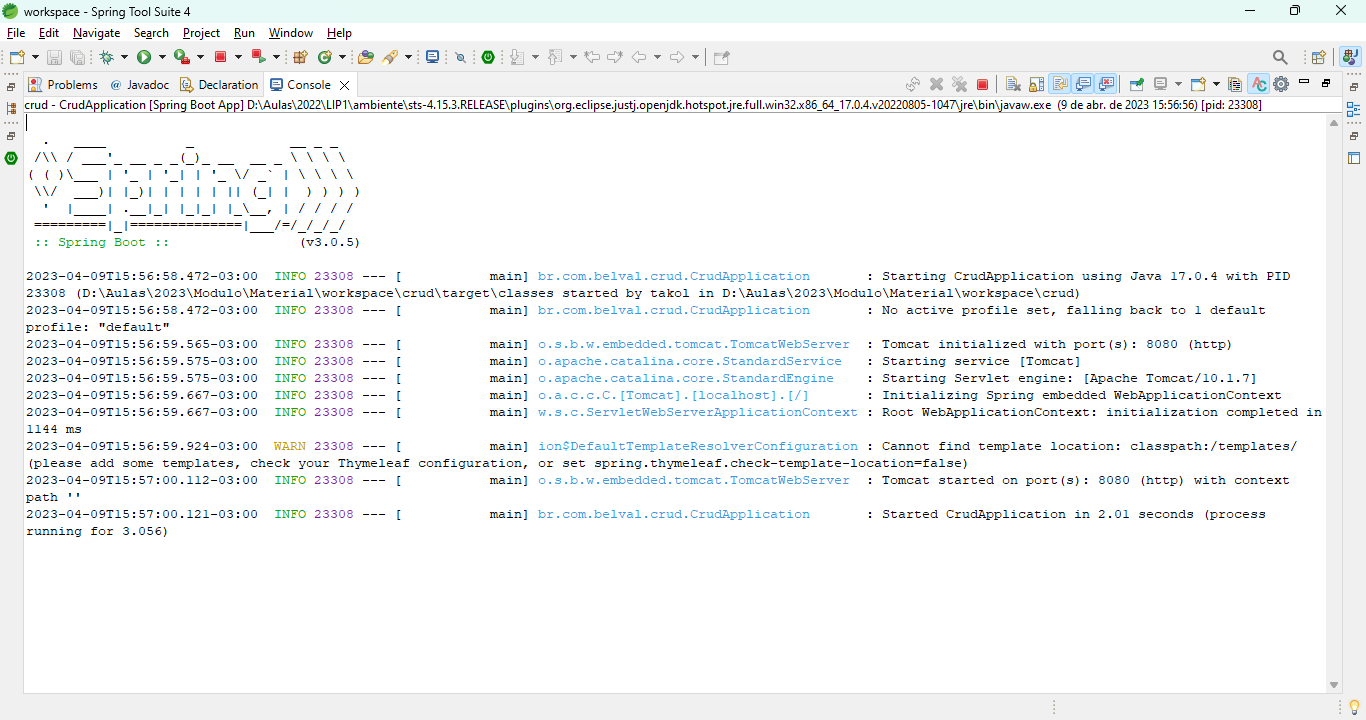


4 – Então é só clicar em “Apply and Close”

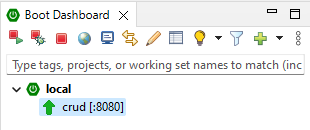


Após executar o projeto, aparecerá, no console, o registro dessa execução com algumas informações, entre elas, que o servidor web tomcat foi executado e que ele e, por extensão, a nossa aplicação também, pois esta roda dentro do tamcat, estão “ouvindo” a porta 8080 da máquina. É através dessa porta que acessaremos nossa aplicação.

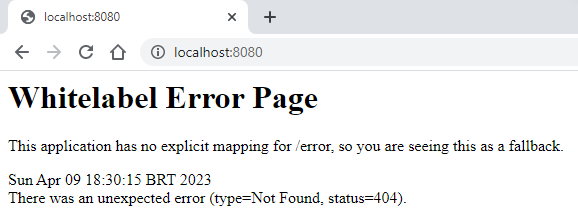




No “Boot Dashboard”, aparecerá



5 – Abra o browser e digite na barra de endereço <http://localhost:8080>



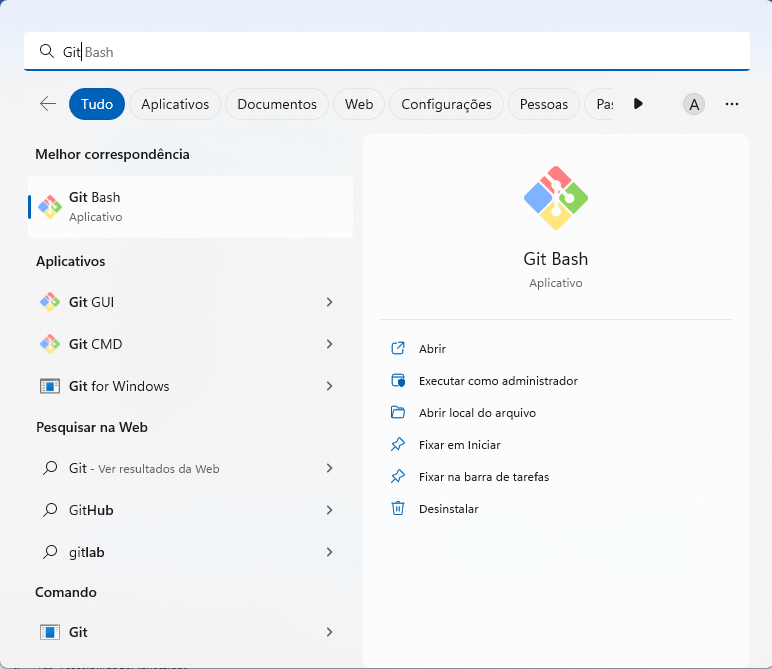
Caso apareça o texto “Whitelabel Error Page”, isso significa que nossa aplicação está funcionando.

# Salvando uma fotografia do projeto

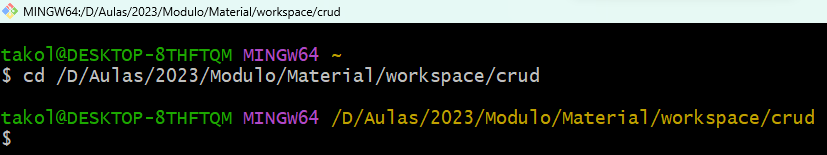
Nesse ponto temos um projeto corretamente configurado e é importante salvarmos uma fotografia desse ponto exato do projeto para que, caso venhamos a “quebrar” nosso projeto ao tentar adicionar uma nova funcionalidade, possamos voltar a ele depois.

Podemos fazer isso através do uso de ferramentas de controle de versão e a ferramenta mais utilizada hoje em dia é o Git. Ele pode ser um pouco complicado às vezes, mas se tratando de uma ferramenta tão utilizada na indústria de desenvolvimento de software é bom começarmos a aprender como utilizá-lo.

1 – Abra o Git Bash



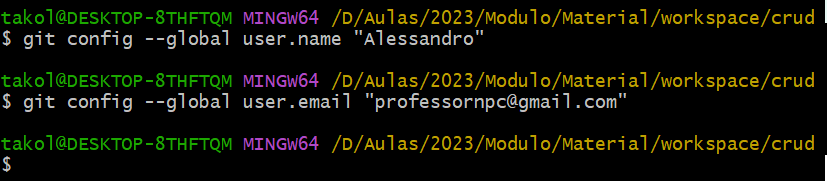
2 – Digite o comando “cd <caminho do diretório do projeto>”



3 – Configure o seu usuário e seu e-mail digitando o comando abaixo

git config --global user.name "Seu Nome"

git config --global user.email "seu\_email@qualquercoisa.com"

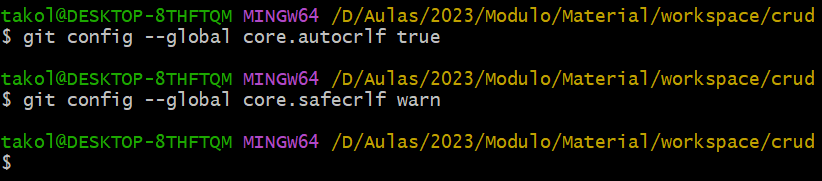


4 – Configure os delimitadores de linha para o bom funcionamento da ferramenta diff

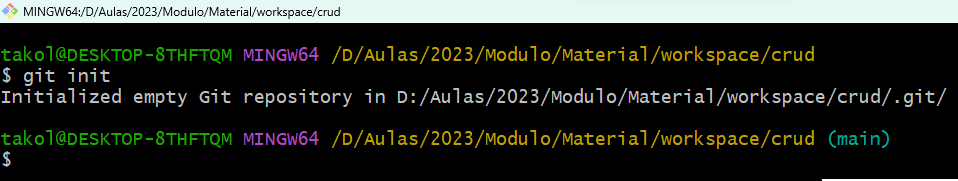
No Windows digite

git config --global core.autocrlf true

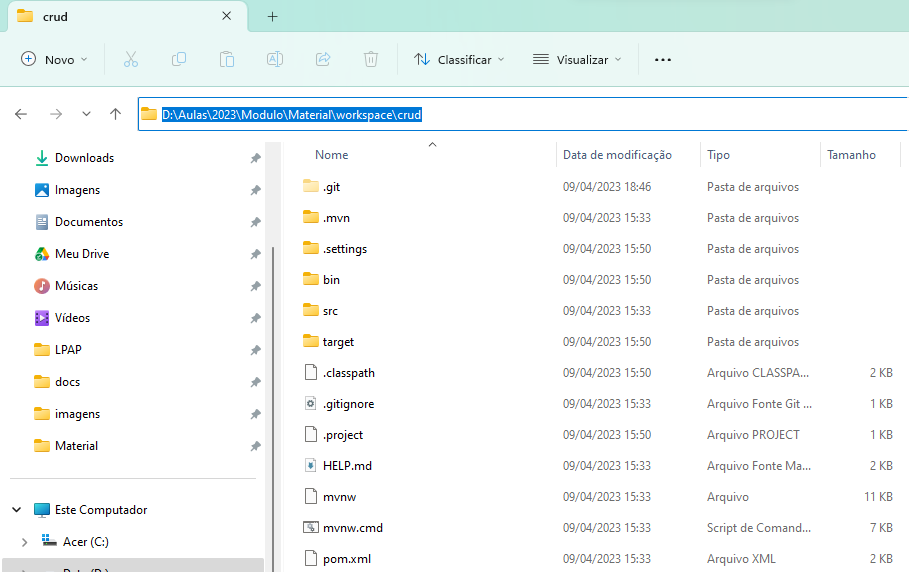
git config --global core.safecrlf warn



5 – Crie um novo repositório com o comando git ini

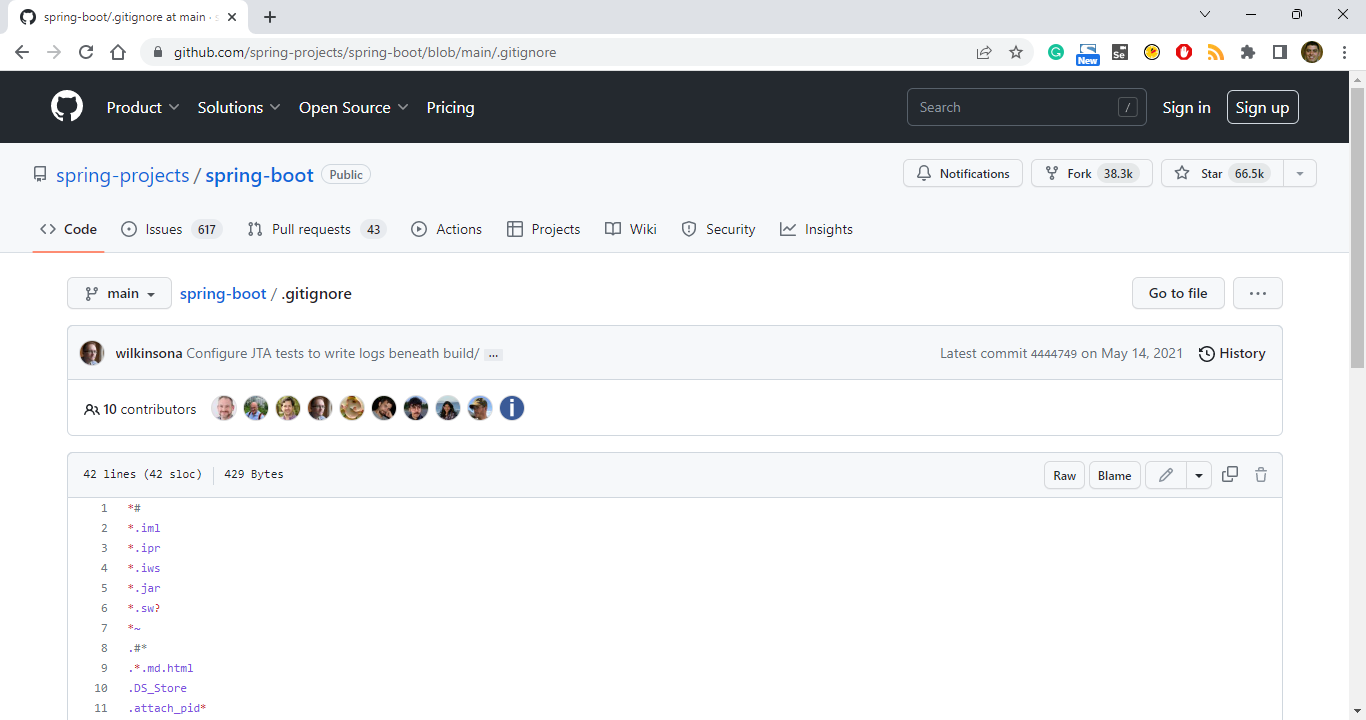


Ao executar esse comando, um diretório “.git”, oculto, é criado dentro do diretório app1

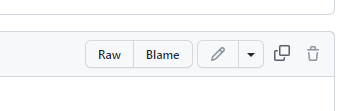


Esse diretório será o repositório que conterá todas as diferentes versões do nosso software nos permitindo recuperar versões antigas caso necessário. NÃO APAGUE ESSE DIRETÓRIO.

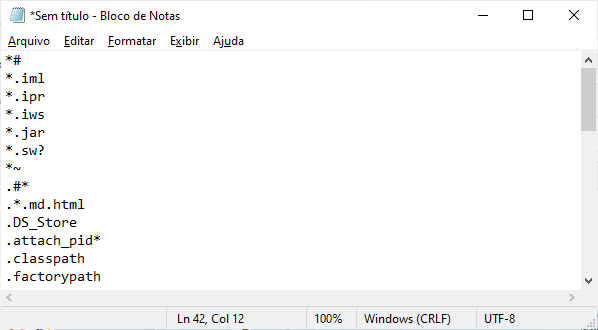
6 – Acesso o endereço <https://github.com/spring-projects/spring-boot/blob/main/.gitignore>



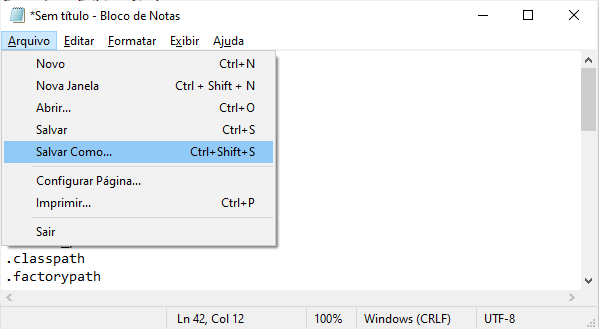
7 – Clique no botão copiar



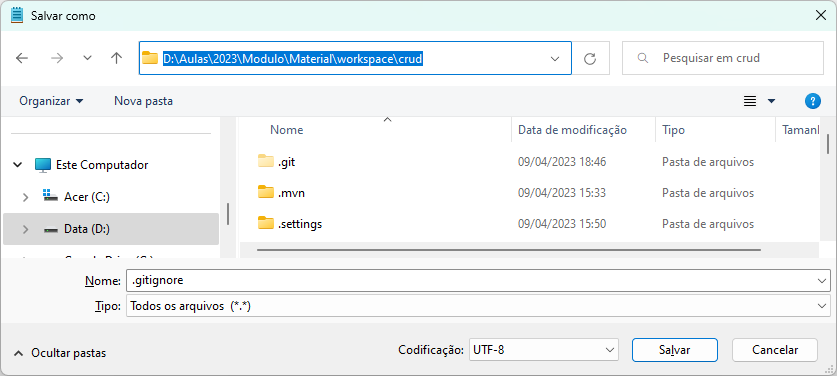
8 – Abra o notepad e cole o conteúdo copiado



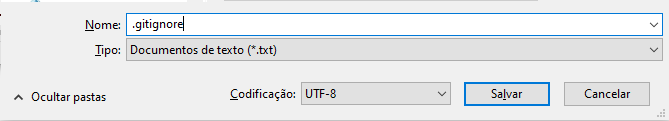
9 – Clique em Arquivo > Salvar como...



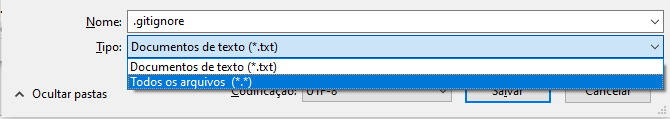
10 – Selecione a pasta do projeto (.../workspace/crud)



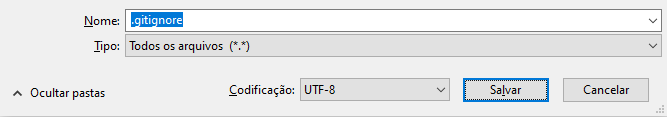
11 – Preencha o nomo do arquivo como “.gitignore” (observe o ponto inicial)



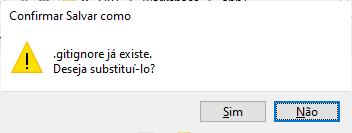
12 – Selecione o tipo “Todos os arquivos (\*.\*)”

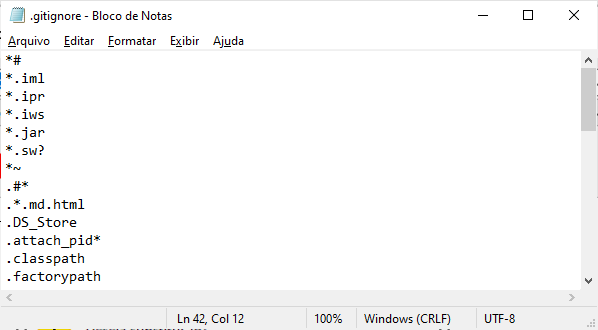


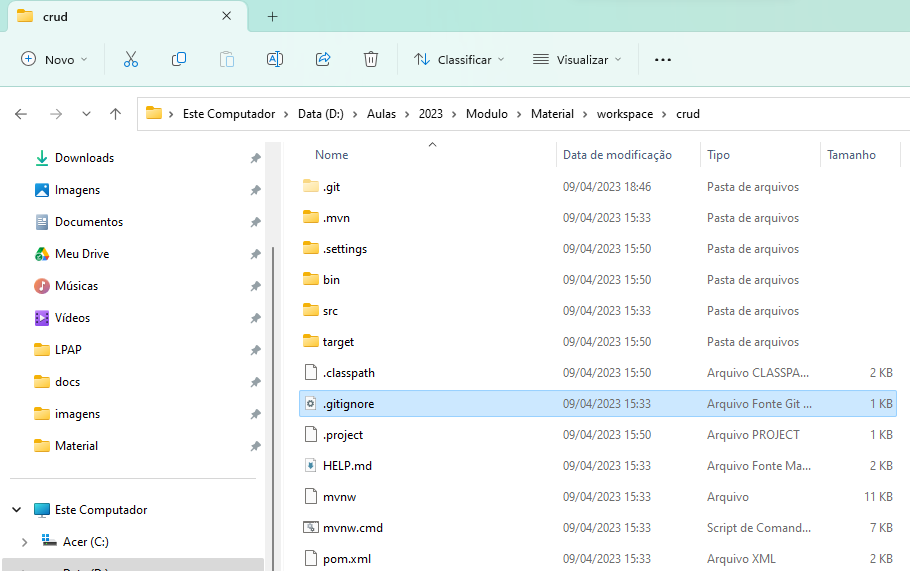
13 – Deixe a codificação como “UTF-8” e clique em Salvar



14 – Se o arquivo já existir e for perguntado se deseja substituí-lo, clique em Sim.



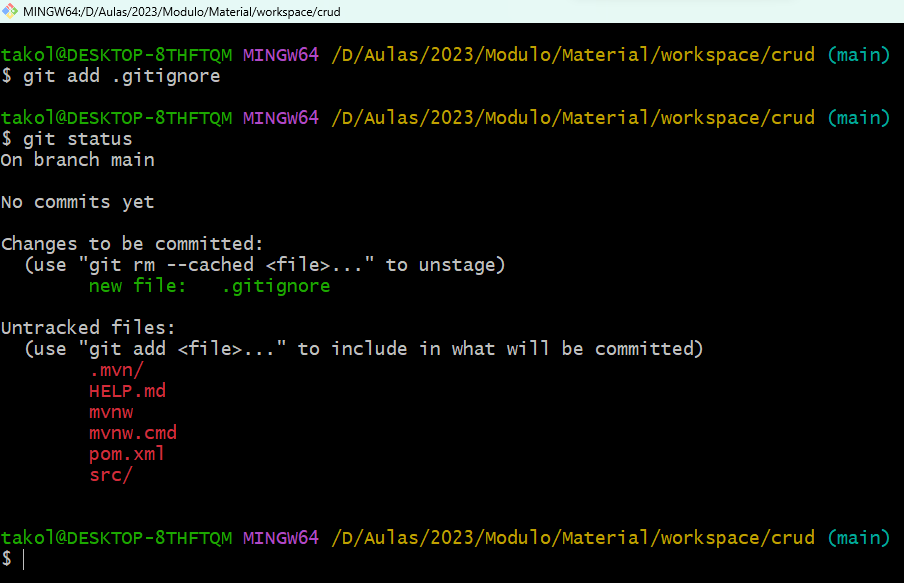




15 – Volte ao git bash e digite os comandos abaixo

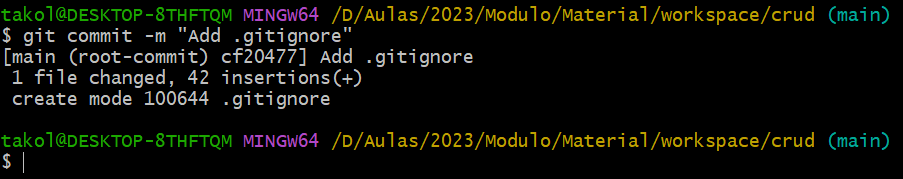
git add .gitignore

git status



16 – Em seguida digite o comando

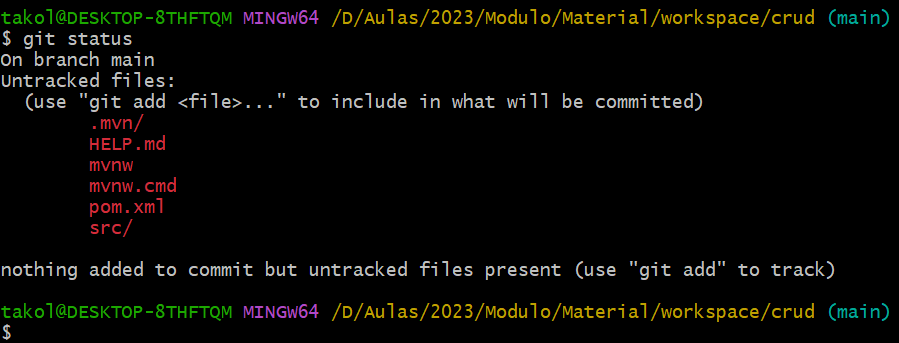
git commit -m "Add .gitignore"



Isso fará com que o git exclua os arquivos compilados deixando apenas o código fonte dentro do repositório e com isso deixando o repositório mais leve.

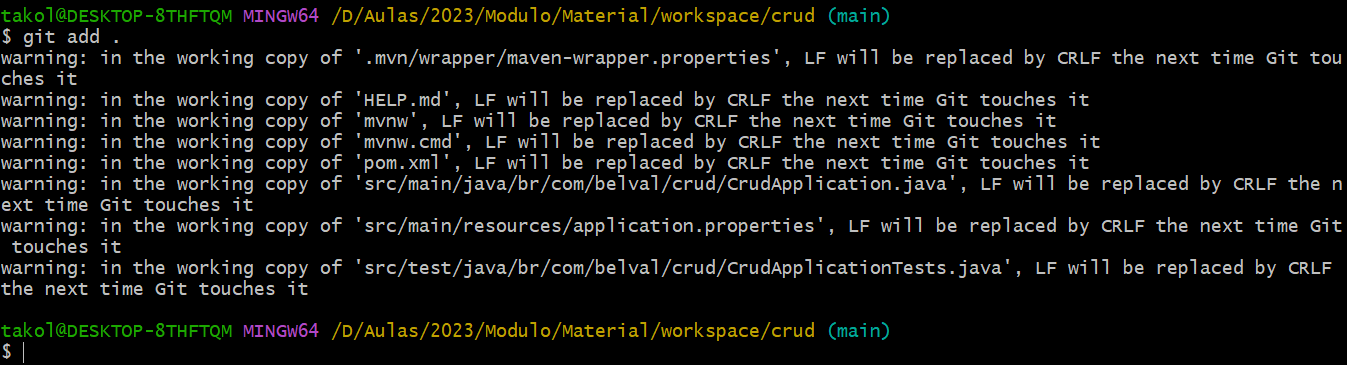
16 – Execute o comando abaixo para ver o status do “ambiente de trabalho” do git

git status

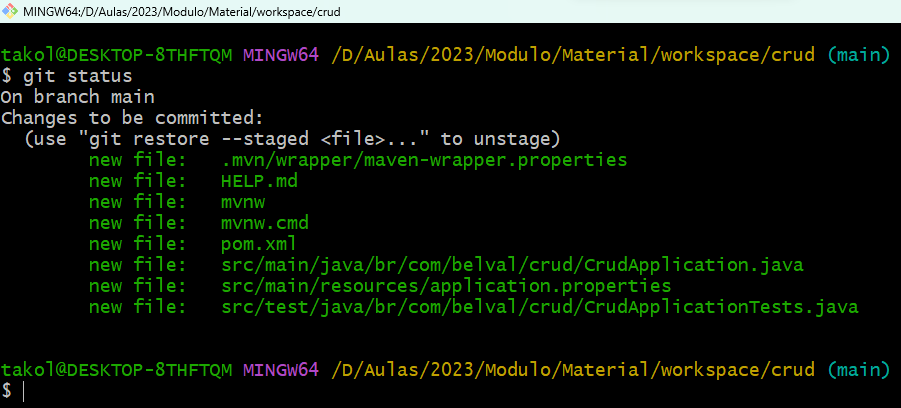


17 – Agora, vamos adicionar os fontes do projeto para “stage” do git, digite o comando abaixo

git add .

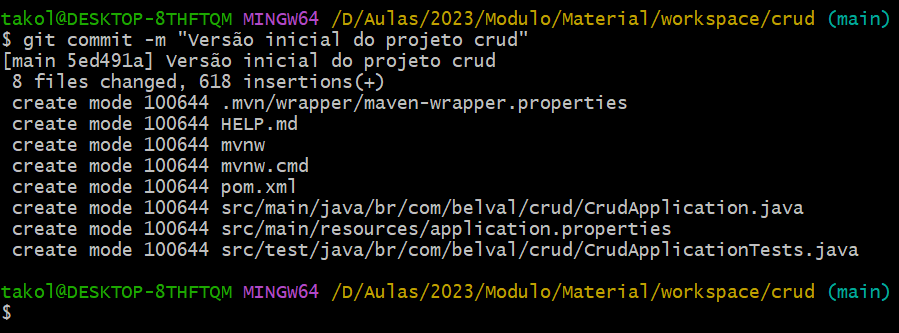


18 – Execute um comando “git status” para ver o que será adicionado ao repositório



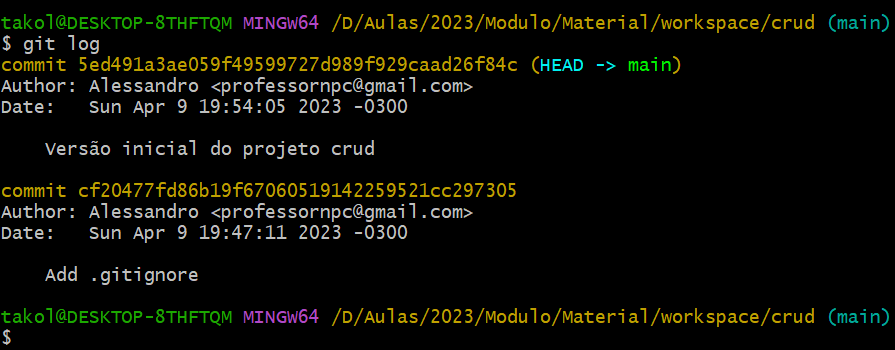
19 – Execute o comando abaixo para salvar a versão atual dos fontes no repositório.

git commit -m "Versão inicial do projeto crud"



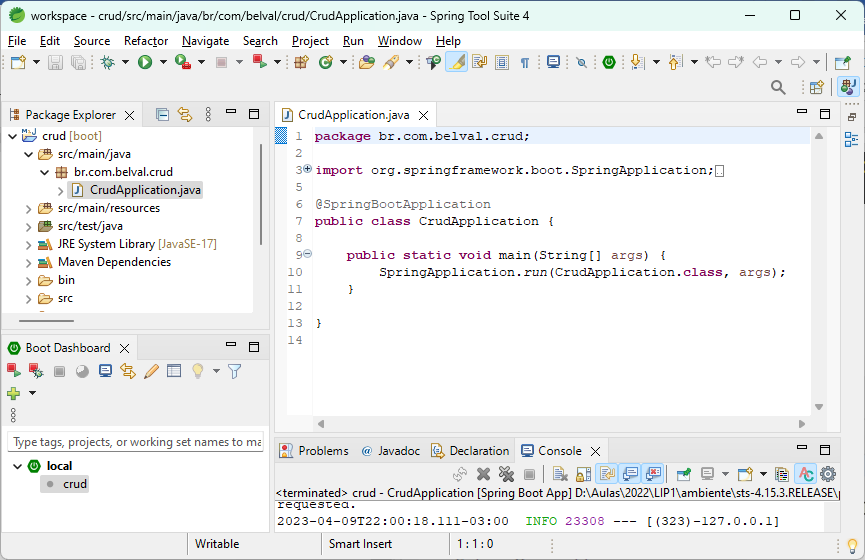
20 – Verifique o histórico das versões do seu software digitando o comando abaixo

git log



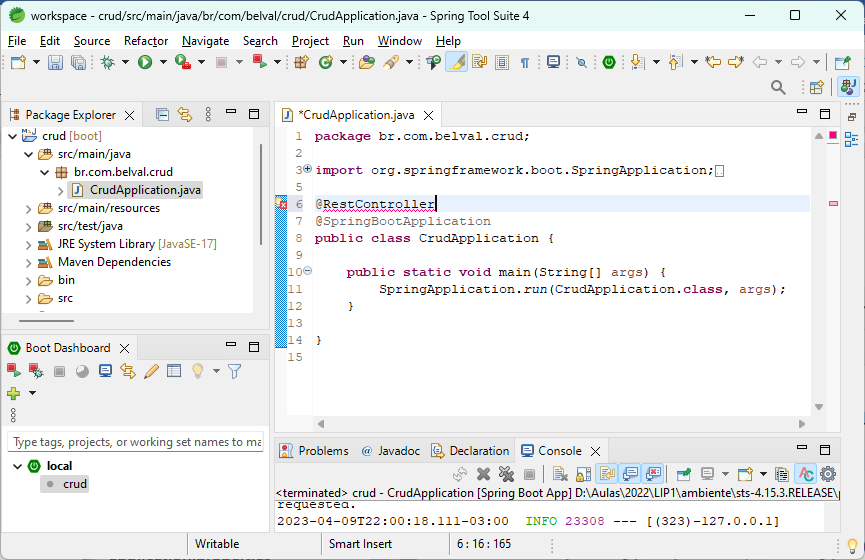
# Fazendo a aplicação responder algo

1 – Abra o arquivo CrudApplication.java



2 – Para que nossa classe possa responder alguma coisa, a primeira coisa a fazer é “anotar” a classe como sendo um “Controller”. Talvez a versão mais simples de controller que podemos utilizar são os “RestControllers” que permitirão que nossos métodos respondam um texto diretamente para os clientes.

Coloque “@RestController” antes de “@SpringBootApplication”

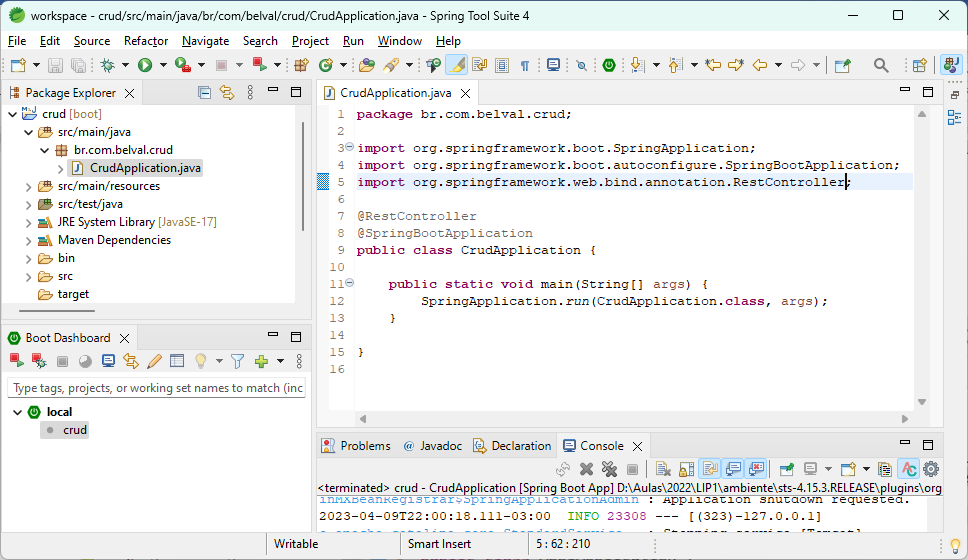


Como você pode ver, o nosso fonte apresenta um erro de compilação.

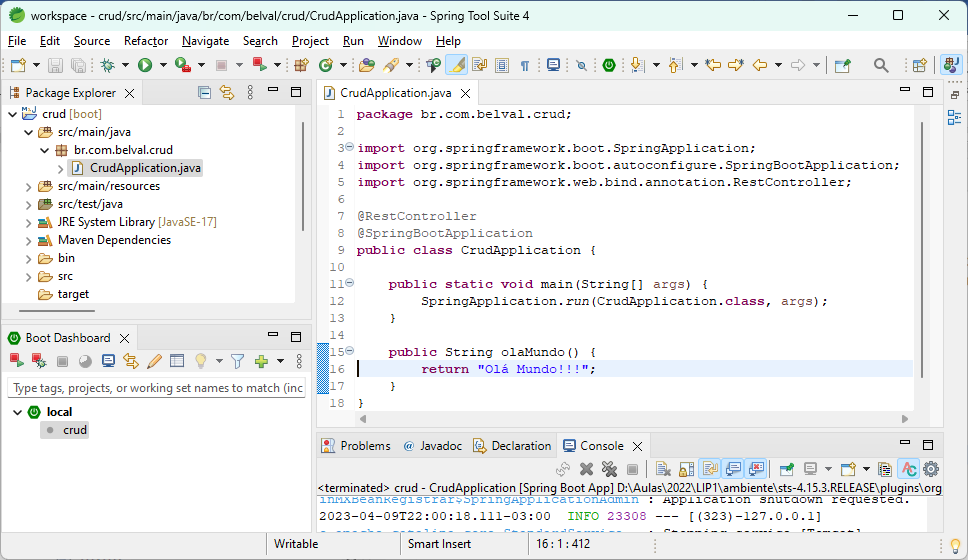
O problema aqui é que não adicionamos a importação da classe correspondente à anotação “@RestController”.

Para resolver isso podemos usar um atalho.

3 – Pressione “CTRL + SHIFT + O” que o ambiente automaticamente adiciona o *import* necessário.



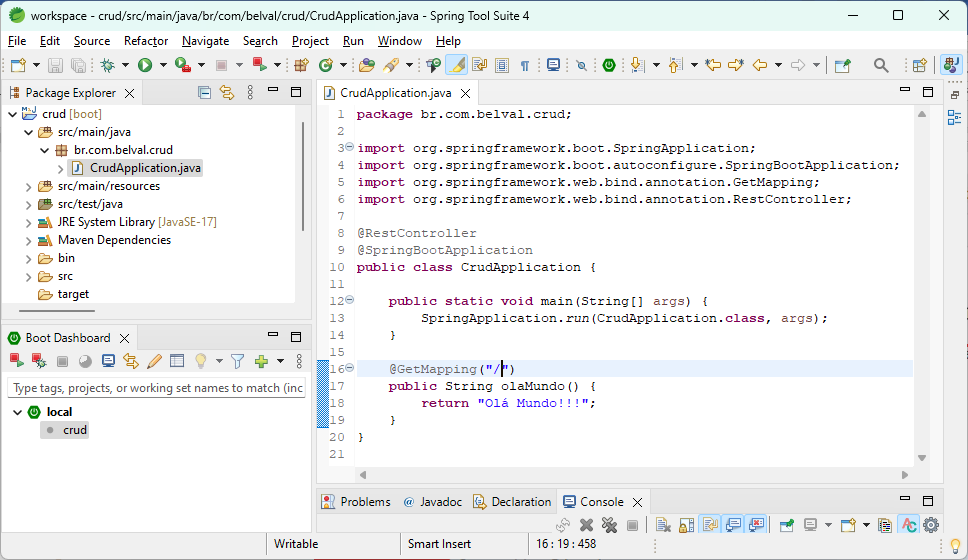
4 – Crie um método olaMundo() como abaixo



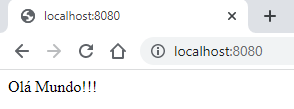
5 – Anote esse método com

@GetMapping(“/”)

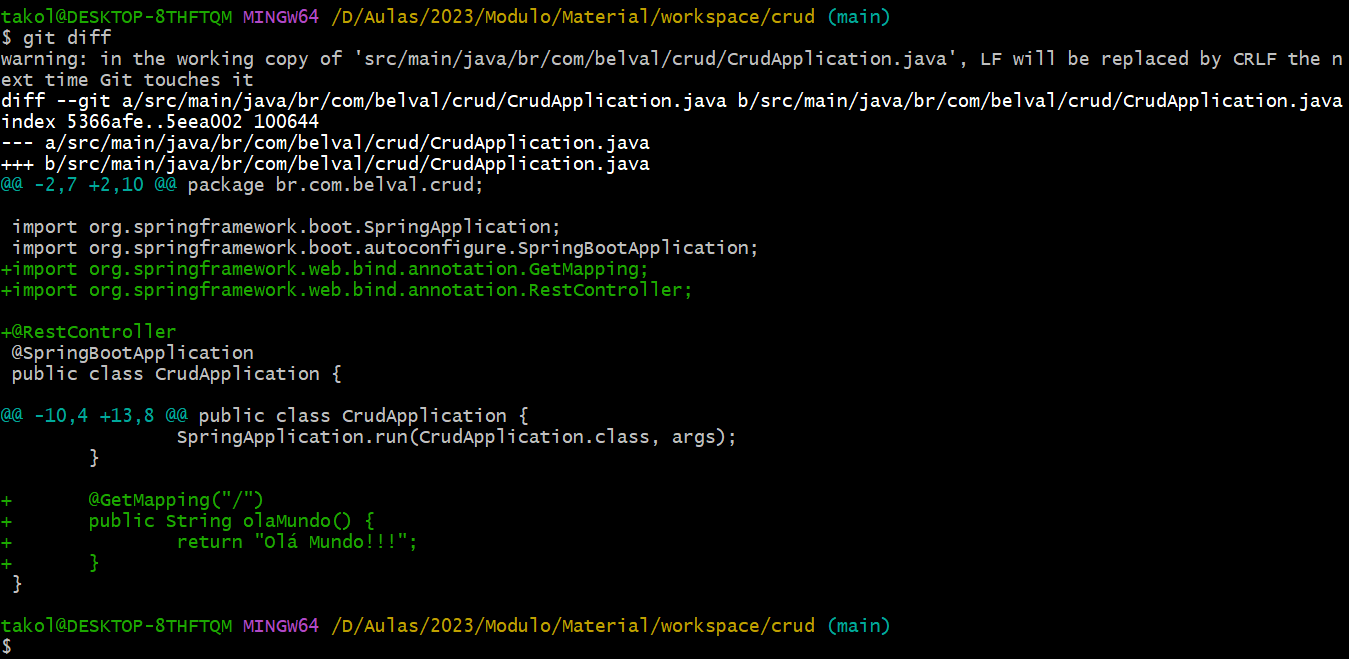
E na sequência use o atalho “CTRL + SHIFT + O” para adicionar o *import* necessário.



6 – Execute a aplicação novamente e acesse o endereço “localhost:8080/”



7 – Pare a aplicação, e no git bash, execute o comando “git diff”

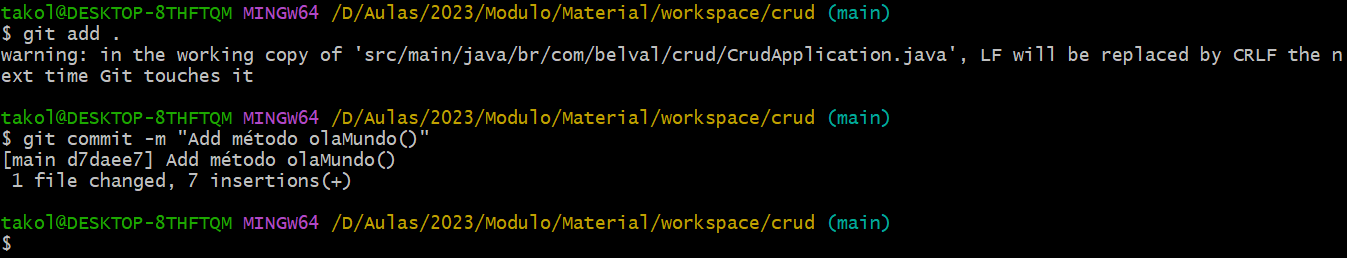


Observe que, as linhas que começam com “+” e que aqui também estão verdes, são aquelas que foram adicionadas em relação à última versão do nosso software.

8 – Se estiver tudo certo, execute os comandos abaixo e salve uma nova fotografia do nosso projeto.

git add .

git commit -m “Add método olaMundo()”



9 – Para verificar o histórico do nosso projeto, execute o comando “git log”

