

MBA EM **ENGENHARIA DE SOFTWARE**

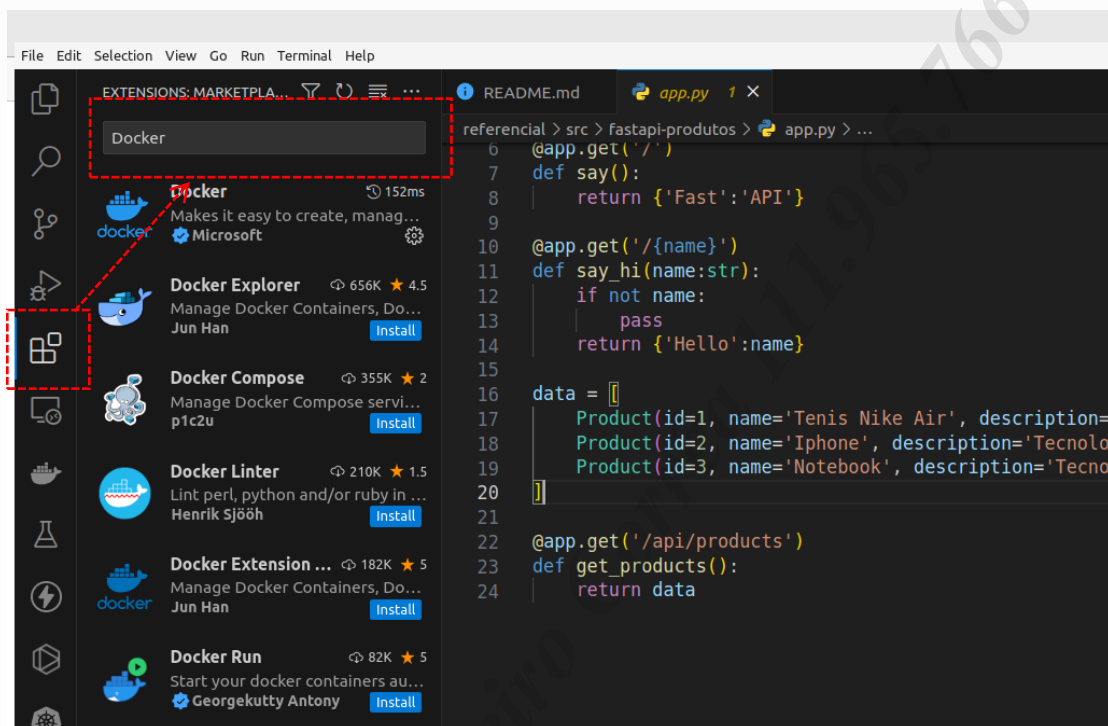
MATERIAL COMPLEMENTAR

Containerização de serviços (Docker)

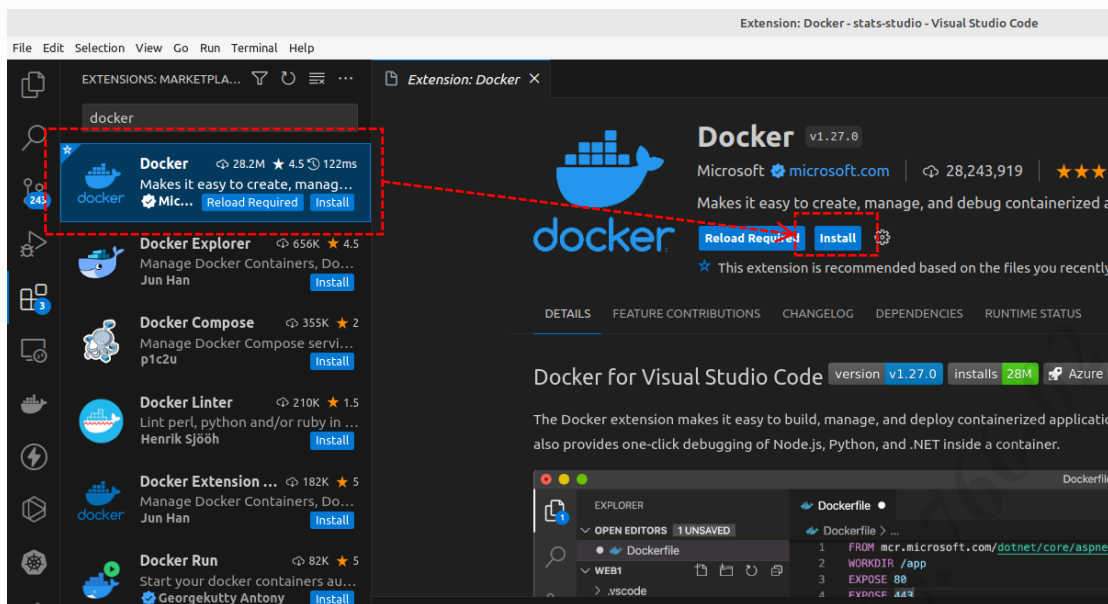
Professor Helder Prado Santos

TUTORIAL EXTENSÃO DOCKER DO VS CODE E ACESSO AO TERMINAL DO CONTAINER

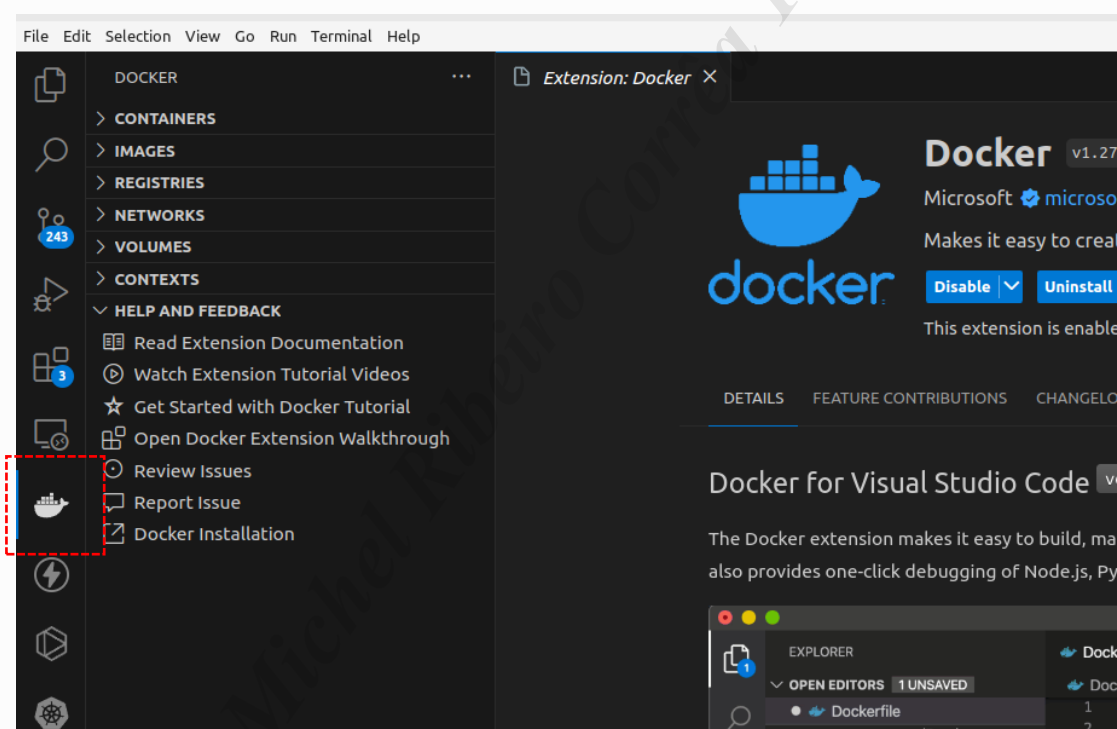
1. Acessar a aba extensões do VS Code clicando no botão sinalizado na figura abaixo e digitando “Docker”, como está sinalizado no quadrado em vermelho.



2. A extensão do Docker vai aparecer na primeira posição e os cliques devem ser feitos como aparecem na figura, primeiro na janela da extensão do Docker e depois no botão “Install”.

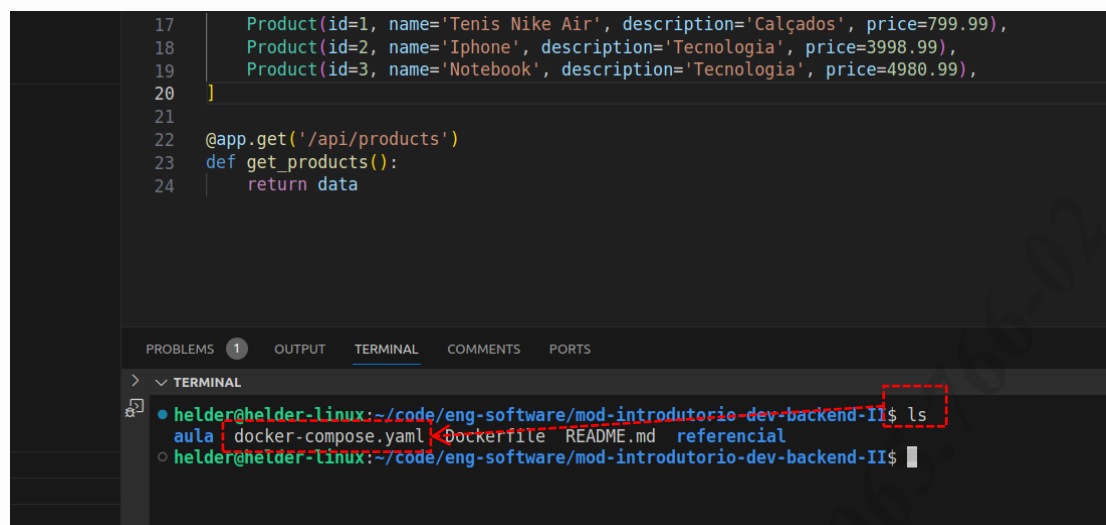


3. O ícone do Docker vai aparecer como uma nova aba do seu VS Code, assim como mostra na figura abaixo.



4. Nesse momento é importante identificar se você está no caminho correto ou não. No terminal do VS Code, digite o comando `ls` e aperte enter, se aparecer um arquivo chamado `docker-compose.yaml`, quer dizer que você

está no caminho correto. Os passos devem seguir assim como está sendo mostrado na figura abaixo.



The screenshot shows a code editor with Python code defining a list of products and a GET endpoint. Below the code, a terminal window displays the command `ls` and its output, which lists files in the current directory. A red dashed box highlights the terminal output.

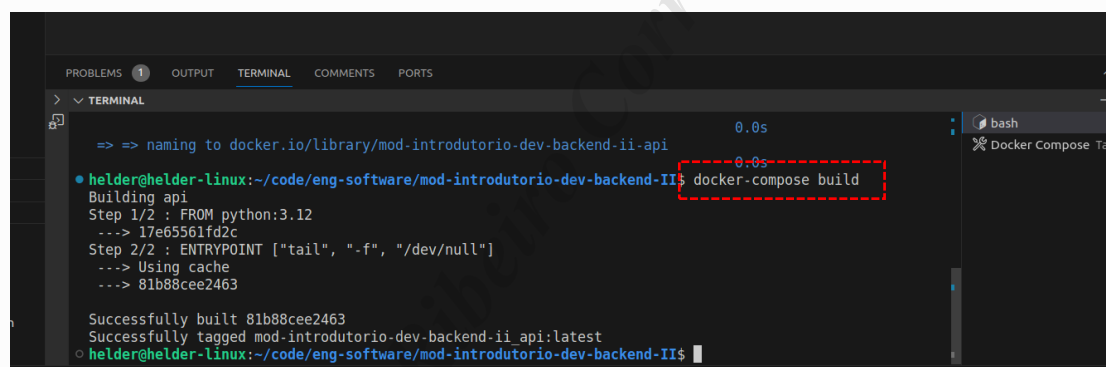
```
17 Product(id=1, name='Tenis Nike Air', description='Calçados', price=799.99),
18 Product(id=2, name='Iphone', description='Tecnologia', price=3998.99),
19 Product(id=3, name='Notebook', description='Tecnologia', price=4980.99),
20 ]
21
22 @app.get('/api/products')
23 def get_products():
24     return data
```

PROBLEMS 1 OUTPUT TERMINAL COMMENTS PORTS

TERMINAL

```
helder@helder-linux:~/code/eng-software/mod-introductorio-dev-backend-II$ ls
aula  docker-compose.yaml  Dockerfile  README.md  referencial
```

5. Para criar o seu ambiente de desenvolvimento é necessário gerar as imagens do container com o comando `docker-compose build`, assim como mostra a figura abaixo.

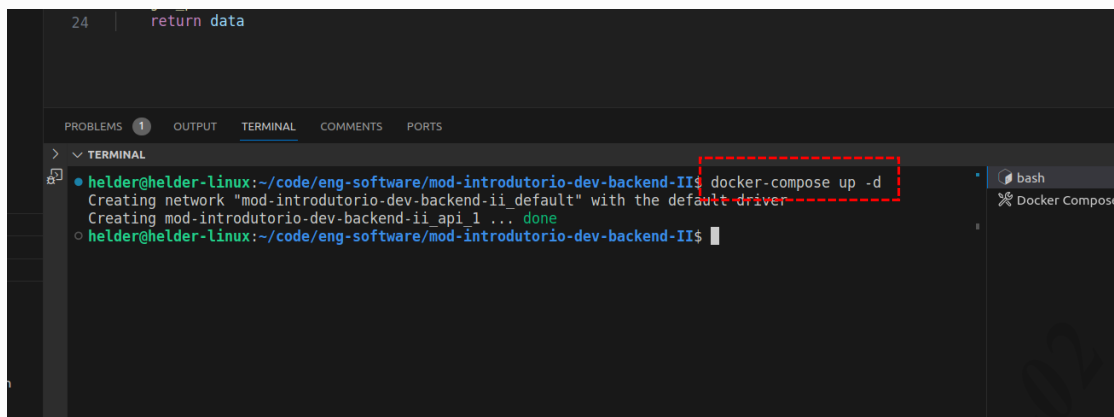


The screenshot shows a terminal window with the output of the `docker-compose build` command. The output shows the building of the 'api' service, including the steps for pulling the base image and running the build command. A red dashed box highlights the command `docker-compose build`.

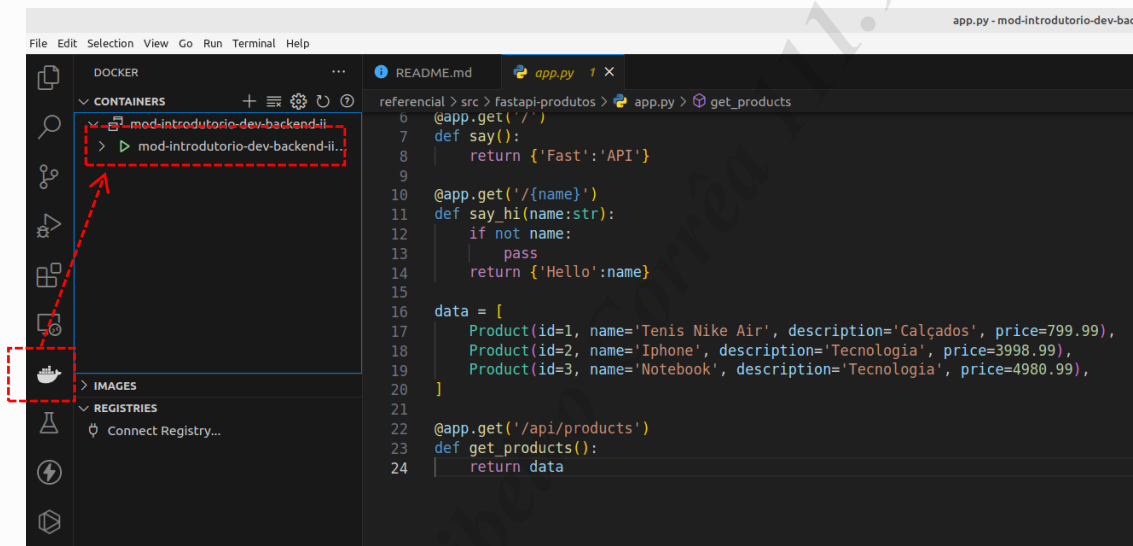
```
=> => naming to docker.io/library/mod-introductorio-dev-backend-ii-api 0.0s
helder@helder-linux:~/code/eng-software/mod-introductorio-dev-backend-II$ docker-compose build
Building api
Step 1/2 : FROM python:3.12
--> 17e65561fd2c
Step 2/2 : ENTRYPOINT ["tail", "-f", "/dev/null"]
--> Using cache
--> 81b88cee2463

Successfully built 81b88cee2463
Successfully tagged mod-introductorio-dev-backend-ii api:latest
```

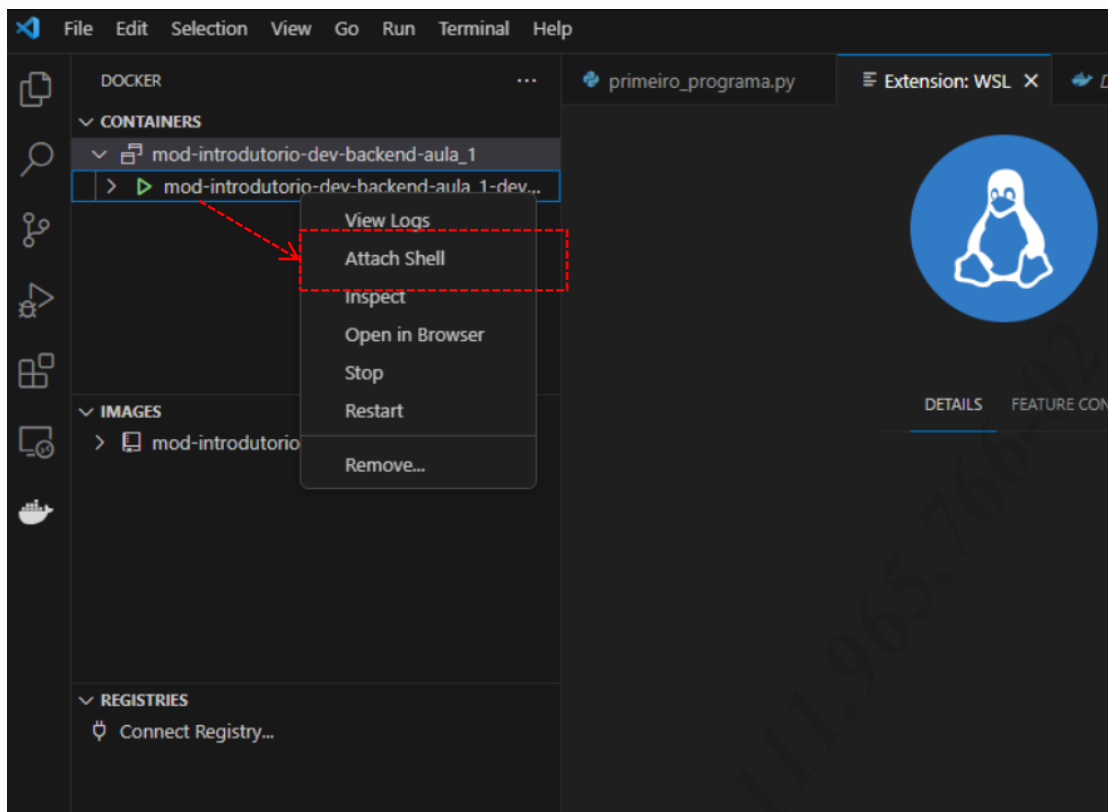
6. Agora que as imagens dos containers já estão geradas, é possível subir e acessar os containers utilizando o comando `docker-compose up -d`, assim como está sendo mostrado na figura abaixo.



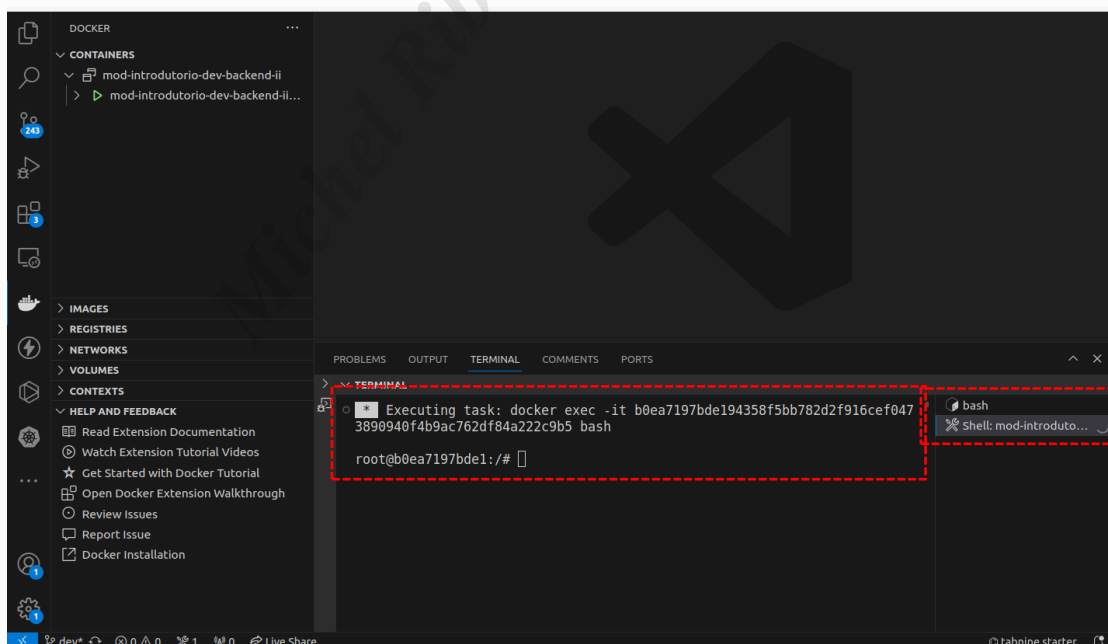
7. Depois que os containers forem criados, eles irão aparecer na aba da extensão do Docker no VS Code, assim como mostra a figura e os quadrados vermelhos de sinalização.



8. Para acessar o terminal do container é só clicar com o botão direito do mouse e seleccionar a opção Attach Shell que vai aparecer através de uma janela abaixo do containers. A imagem abaixo mostra o que deverá aparecer nesse momento.



9. Após essa operação, um novo terminal irá aparecer na parte inferior do VS Code e o nome do terminal vai ser diferente do que é utilizado pelo seu usuário, isso quer dizer que você está acessando o terminal do container. Os quadrados vermelhos sinalizam o que deve acontecer quando você seguir os mesmos passos.



10. Para identificar se tudo foi feito da forma correta, você pode utilizar a função de alguma coisa que foi instalada apenas no container. A imagem abaixo mostra o exemplo do acesso do console da linguagem Python através do terminal do container com o comando python, isso só funcionou porque o container tinha o Python instalado.

