

## **A3 Sistemas Distribuídos**

Elaine Santana Gonzaga- 12722131402

Herbert Lopes Santana da Guia- 1272211389

Guilherme Goes Xavier Gonçalves- 12722131927

Priscila Barbosa de Oliveira- 1272213416

Sabrina Filgueiras Alves Raiol 1272217396

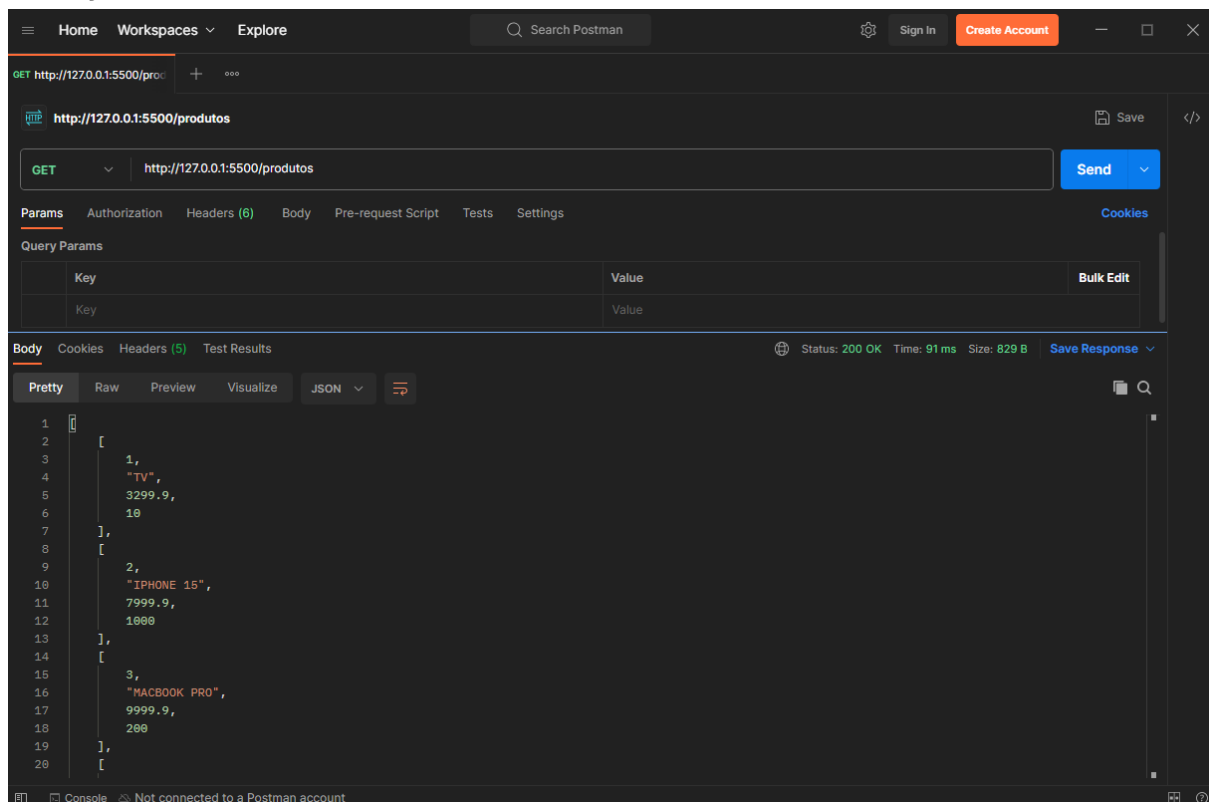
# RELATÓRIO

Este documento destina-se ao trabalho da UC de Sistemas Distribuídos, que tem como objetivo criar uma aplicação que faça a simulação de dados de venda de uma rede de lojas. Segue os passos para a realização e execução da aplicação.

## Passo 1: Ambiente de Desenvolvimento:

É preciso ter um ambiente de desenvolvimento adequado configurado. Por isso, é preciso ter instalado uma IDE ou a configuração de um ambiente virtual, porque estamos utilizando a linguagem Python em nossa aplicação. Necessário ter a linguagem Python instalado na máquina. Usamos como componentes principais na aplicação o Banco de dados SQLite3, BackEnd, Flask, API.

### Execução da API



## Passo 2: Realizando o download do código:

Realize o download do arquivo propiciado via GitHub abaixo:

<https://github.com/GuilhermeXGoncalves/Entrega-UC-SD>

## Passo 3: Preparando o Ambiente:

Recomendamos a criação de um ambiente virtual.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + - [ ] [ ] ... ^ X
● PS C:\Users\guilh\Desktop\Entrega-UCSD> python -m venv venv
○ PS C:\Users\guilh\Desktop\Entrega-UCSD> [ ]
```

Ativando o ambiente virtual

```
● PS C:\Users\guilh\Desktop\Code> .\venv\Scripts\Activate.ps1
○ (venv) PS C:\Users\guilh\Desktop\Code> [ ]
```

## Passo 4: Instalando Dependências:

COMANDOS:

SQLite3: pip install sqlite3:

```
● (venv) PS C:\Users\guilh\Desktop\Code> pip install sqlite3
ERROR: Could not find a version that satisfies the requirement sqlite3 (from versions: none)
ERROR: No matching distribution found for sqlite3

[notice] A new release of pip is available: 23.2.1 -> 23.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

Flask: pip install Flask:

```
● (venv) PS C:\Users\guilh\Desktop\Code> pip install flask
Collecting flask
  Obtaining dependency information for flask from https://files.pythonhosted.org/packages/36/42/015c23896649b98c809c69388a805a571a3bea44362fe87e33fc3afa01f/flask-3.0.0-py3-none-any.whl.metadata
  Using cached flask-3.0.0-py3-none-any.whl.metadata (3.6 kB)
Collecting Werkzeug>=3.0.0 (from flask)
  Obtaining dependency information for Werkzeug>=3.0.0 from https://files.pythonhosted.org/packages/c3/fc/254c3e9b5feb89ff5b9076a23218dafbc9c96ac5941e90b71206e6313b/werkzeug-3.0.1-py3-none-any.whl.metadata
  Using cached werkzeug-3.0.1-py3-none-any.whl.metadata (4.1 kB)
Collecting Jinja2>=3.1.2 (from flask)
  Using cached Jinja2-3.1.2-py3-none-any.whl (133 kB)
Collecting itsdangerous>=2.1.2 (from flask)
  Using cached itsdangerous-2.1.2-py3-none-any.whl (15 kB)
Collecting click>=8.1.3 (from flask)
  Obtaining dependency information for click>=8.1.3 from https://files.pythonhosted.org/packages/00/2e/d53fa4befb2cfa713304affc7ca780ce4fc1fd871052771b58311a3229/click-8.1.7-py3-none-any.whl.metadata
  Using cached click-8.1.7-py3-none-any.whl.metadata (3.0 kB)
Collecting blinker>=1.6.2 (from flask)
  Obtaining dependency information for blinker>=1.6.2 from https://files.pythonhosted.org/packages/fa/2a/7f3714cbc6356a8efec525ce7a8613d581072edeb53eb7b9754f33db087/blinker-1.7.0-py3-none-any.whl.metadata
  Using cached blinker-1.7.0-py3-none-any.whl.metadata (1.9 kB)
Collecting colorama (from click>=8.1.3->flask)
  Using cached colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Collecting MarkupSafe>=2.0 (from Jinja2>=3.1.2->flask)
  Obtaining dependency information for MarkupSafe>=2.0 from https://files.pythonhosted.org/packages/44/44/dbaf65876e258facd05f586ddc158387ab89963e7f223551afc9c2e24c2/MarkupSafe-2.1.3-cp312-cp312-win_and64.whl.metadata
  Using cached MarkupSafe-2.1.3-cp312-cp312-win_and64.whl.metadata (3.0 kB)
Using cached flask-3.0.0-py3-none-any.whl (99 kB)
Using cached blinker-1.7.0-py3-none-any.whl (13 kB)
Using cached click-8.1.7-py3-none-any.whl (97 kB)
Using cached werkzeug-3.0.1-py3-none-any.whl (226 kB)
Using cached MarkupSafe-2.1.3-cp312-cp312-win_and64.whl (16 kB)
Installing collected packages: MarkupSafe, itsdangerous, colorama, blinker, Werkzeug, Jinja2, click, flask
Successfully installed Jinja2-3.1.2 MarkupSafe-2.1.3 Werkzeug-3.0.1 blinker-1.7.0 click-8.1.7 colorama-0.4.6 flask-3.0.0 itsdangerous-2.1.2

[notice] A new release of pip is available: 23.2.1 -> 23.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip
○ (venv) PS C:\Users\guilh\Desktop\Code> [ ]
```

```

• (venv) PS C:\Users\guilh\Desktop\Code> git clone https://github.com/GuilhermeXGoncalves/Entrega-UC-SD.git
Cloning into 'Entrega-UC-SD'...
remote: Enumerating objects: 44, done.
Receiving objects: 100% (44/44), 7.17 KiB | 918.00 KiB/s, done./44)
remote: Counting objects: 25% (11/44)
remote: Counting objects: 100% (44/44), done.
remote: Compressing objects: 100% (23/23), done.
remote: Total 44 (delta 23), reused 35 (delta 14), pack-reused 0
• (venv) PS C:\Users\guilh\Desktop\Code>

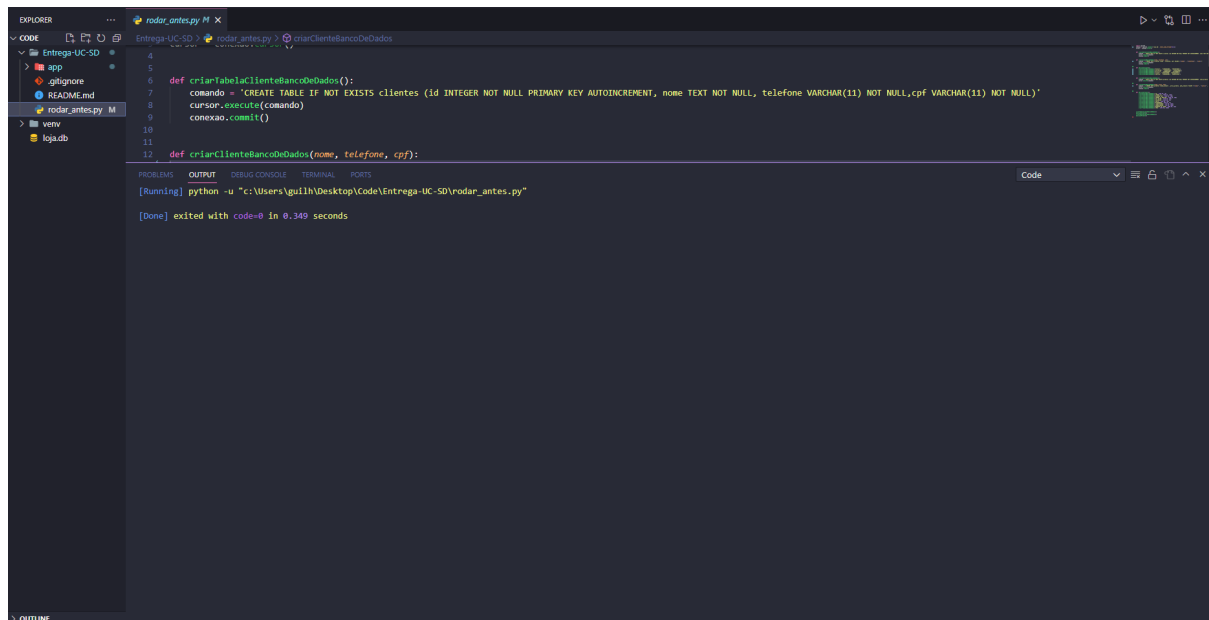
```

Para o sucesso da execução, clone o repositório para a sua máquina e verifique se a mesma tem todas as dependências instaladas dentro do ambiente virtual.

## Passo 5: Execute o Código:

Abra os arquivos dos códigos no seu ambiente de desenvolvimento, olhe com atenção e execute conforme pedido à baixo.

Primeiro é necessário rodar o Script “rodar\_antres.py” usando o “CTRL, ALT, N” antes de iniciar o servidor, após rodar o Script pode fechar.



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows the project structure with files like 'app', 'gignore', 'README.md', 'rodar\_antres.py', 'venv', and 'loja.db'. The main editor area displays the code for 'rodar\_antres.py', which includes a function to create a database table and another to create a client record. The Output pane at the bottom shows the command 'python -u "c:\Users\guilh\Desktop\Code\Entrega-UC-SD\rodar\_antres.py"' being executed, resulting in '[Done] exited with code=0 in 0.349 seconds'.

```

4
5
6 def criarTabelaClienteBancoDeDados():
7     comando = 'CREATE TABLE IF NOT EXISTS clientes (id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, nome TEXT NOT NULL, telefone VARCHAR(11) NOT NULL, cpf VARCHAR(11) NOT NULL)'
8     cursor.execute(comando)
9     conexao.commit()
10
11
12 def criarClienteBancoDeDados(nome, telefone, cpf):

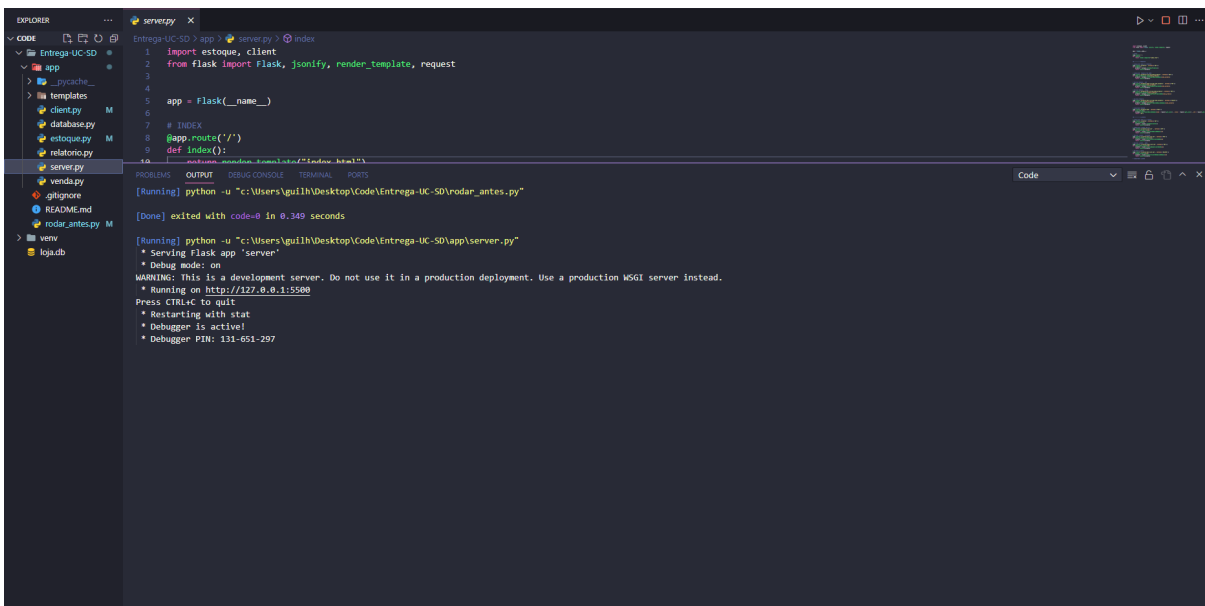
```

```

[Running] python -u "c:\Users\guilh\Desktop\Code\Entrega-UC-SD\rodar_antres.py"
[Done] exited with code=0 in 0.349 seconds

```

Após ser realizado a tarefa acima, faça a inicialização do servidor.



The screenshot shows a VS Code editor with a project named 'Entrega-UC-SD'. The file explorer on the left shows a directory structure with files like 'app.py', 'client.py', 'database.py', 'entoque.py', 'relatorio.py', 'server.py', 'venda.py', 'README.md', 'rodar\_antes.py', 'venv', and 'lga.db'. The main editor window displays the 'server.py' file, which contains the following code:

```
1 import estoque, client
2 from flask import Flask, jsonify, render_template, request
3
4
5 app = Flask(__name__)
6
7 # INDEX
8 @app.route('/')
9 def index():
10     return render_template("index.html")
```

The terminal window at the bottom shows the command to run the application: `python -u "c:\Users\guilh\Desktop\Code\Entrega-UC-SD\rodar_antes.py"`. The output indicates that the application is running on `http://127.0.0.1:5500` and provides instructions for debugging and quitting.

Abra o Script “app.py” e rode o código, assim irá ser iniciado o servidor. Onde encontrará o gerenciamento de clientes, estoques, vendas e geração de relatórios. Terá também cinco clientes e 10 produtos já cadastrados.

## Por que usar o Python?

As razões para escolher Python ao desenvolver a aplicação, é que Python tem uma sintaxe limpa e legível, mais “próxima” da linguagem humana (linguagem de alto nível), o que torna mais fácil escrever e entender o código, além de Python ser considerada uma ótima linguagem para iniciantes. Assim temos um desenvolvimento rápido e com menos porcentagem de erros.

Python possui uma coleção gigantesca de bibliotecas e frameworks para diferentes propósitos, na qual usamos o Flask para o desenvolvimento.

Python é uma linguagem de programação de código aberto, o que significa que é livre para uso e distribuição. Isso reduz custos e permite acesso a uma ampla gama de ferramentas e recursos.

Resumindo, Python é uma escolha popular para desenvolvimento de software devido à sua simplicidade, versatilidade, vasta quantidade de bibliotecas e suporte da comunidade. É uma linguagem que pode ser utilizada para uma ampla gama de aplicações em diferentes domínios.

