

# Sistema de Gerenciamento de Reservas de Mesas

## Comunicação Cliente-Servidor em Python

### Resumo

Este projeto implementa um sistema de reservas de mesas para um restaurante, utilizando arquitetura cliente-servidor com sockets TCP em Python e banco de dados SQLite. O sistema permite que diferentes tipos de usuários interajam:

- Atendentes: criar ou cancelar reservas.
- Garçons: confirmar reservas.
- Gerentes: gerar relatórios por mesa, por período ou por garçom.

### Objetivos

- Desenvolver um sistema distribuído simples.
- Trabalhar com comunicação via sockets.
- Praticar manipulação de banco de dados SQLite.
- Simular perfis de usuários com diferentes permissões.

### Tecnologias Utilizadas

Tecnologia	Descrição
Python	Linguagem de programação principal
Sockets TCP	Comunicação entre cliente e servidor
SQLite	Armazenamento local de dados
VS Code	Ambiente de desenvolvimento
Ubuntu OS	Sistema operacional utilizado

## Estrutura do Projeto

```
A3_SD/
├── cliente_atendente/
│   └── cliente_atendente.py
├── cliente_garcom/
│   └── cliente_garcom.py
├── cliente_gerente/
│   └── cliente_gerente.py
├── dataBase/
│   ├── banco.py
│   └── banco.sqlite
├── server/
│   └── servidor.py
├── img/
│   └── unifacs-seeklogo.png
└── README.md
```

## Funcionamento do Sistema

### Atendente

- Criar reservas
- Cancelar reservas
- Comandos enviados: ATENDENTE\_CRIAR, ATENDENTE\_CANCELAR

### Garçom

- Confirmar reservas
- Comando enviado: GARCOM\_CONFIRMAR

### Gerente

- Relatórios por mesa: GERENTE\_RELATORIO\_MESA
- Relatórios por período (formato dd/mm/aaaa): GERENTE\_RELATORIO\_PERIODO
- Relatórios por garçom: GERENTE\_RELATORIO\_GARCOM

## Modelo da Tabela no Banco de Dados

Campo	Tipo	Descrição
<b>id</b>	INTEGER	ID da reserva (PK)
<b>data</b>	TEXT	Data da reserva (ISO: yyyy-mm-dd)
<b>hora</b>	TEXT	Hora da reserva
<b>numero_mesa</b>	INTEGER	Número da mesa
<b>quantidade_pessoas</b>	INTEGER	Número de pessoas
<b>nome_responsavel</b>	TEXT	Nome da pessoa que fez a reserva
<b>status</b>	TEXT	'reservado' ou 'confirmado'
<b>garcom_id</b>	INTEGER	ID do garçom que confirmou (opcional)

## Execução

### 1. Iniciar o Servidor

💡 No Linux/macOS, use:

```
python3 server/servidor.py
```

💡 No Windows, o comando pode ser:

```
python server/servidor.py
```

### 2. Iniciar Clientes (em terminais separados)

```
python3 cliente_atendente/cliente_atendente.py
```

```
python3 cliente_garcom/cliente_garcom.py
```

```
python3 cliente_gerente/cliente_gerente.py
```