

UNIVERSIDADE TECNOLÓGICA FEDERAL DO
PARANÁ – CÂMPUS DOIS VIZINHOS

Relatório de aplicação de vendas

Banco de dados 2

Eduardo B. Gomes Neto

Relatório de aplicação de vendas

Trabalho realizado como
requisito avaliativo parcial na
disciplina de Banco de Dados
2, no curso de Engenharia de
Software - Universidade
Tecnológica Federal do
Paraná
Professor: Erinaldo da Silva Pereira

Dois Vizinhos, 2025

1. Introdução

O presente trabalho tem como objetivo demonstrar uma aplicação desenvolvida para ser utilizada como um sistema de vendas, onde seu principal objetivo seria realizar o login com diferentes usuários, realizar o cadastro de produtos e ao fim poder realizar vendas.

O sistema de vendas tem como objetivo facilitar determinadas tarefas voltadas ao âmbito comercial, e também auxiliar na organização das tarefas realizadas pelo usuário, trazendo um maior conforto e confiança nos processos. Além disso, é necessário comentarmos sobre o postgres que foi o banco de dados utilizado para armazenar os dados e também o gerenciamento de usuários e seus privilégios dentro da aplicação. Com base nessas informações apresetaremos a seguir a aplicação, suas funcionalidades e o seu desenvolvimento.

2. Aplicação

O objetivo desse projeto foi desenvolver uma aplicação de vendas simples e fácil de utilizar com uma interface amigável. Sua principal funcionalidade é realizar vendas, porém a adição de produtos se tornou necessárias para facilitar a realização das vendas e a tela de login permitiria o acesso de diferentes usuários, registrando quem realizou as vendas e também limitando o acesso para diferentes grupos de usuários para determinadas funcionalidades.

As ferramentas utilizadas neste trabalho foram: PHP, HTML e CSS. A linguagem de programação PHP foi utilizada para criar a lógica do sistema e juntamente de uma extensão chamada PDO (PHP Data Objects), que permite a comunicação da linguagem com diversos tipos de bancos de dados. Por fim o HTML foi utilizado como a estrutura para a criação da interface, em conjunto com o CSS que a torna muito mais amigável e agradável ao usuário, além de torná-la

mais bonita também.

As funções do nosso sistema são:

- Realizar o Login com diferentes usuários;
- Adicionar/Atualizar os dados dos produtos dentro da aplicação;
- Realizar vendas.

2.1 Telas do sistema

A seguir as telas desenvolvidas para o sistema.

- Logo na primeira tela, temos o login onde o usuário as credenciais criadas no banco de dados e verifica se ele tem permissão para acessá-lo:

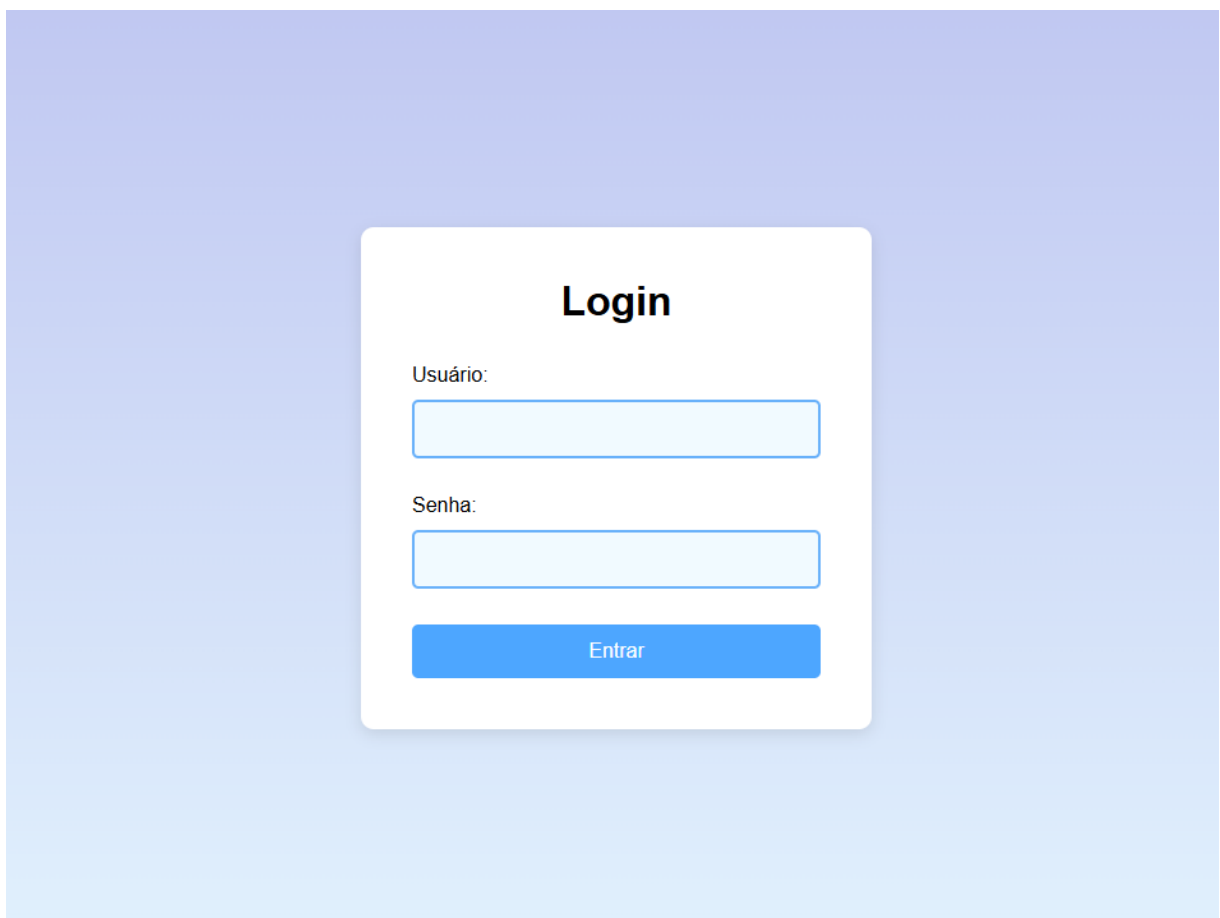
A mockup of a login screen. It features a light blue gradient background. In the center, there is a white rounded rectangle with a subtle shadow. Inside this rectangle, the word "Login" is displayed in a bold, black, sans-serif font. Below the title, there are two labels: "Usuário:" and "Senha:". Each label is followed by a light blue rectangular input field with a thin blue border. At the bottom of the white rectangle, there is a solid blue button with the word "Entrar" in white, centered text.

Imagem 1 – Tela de login

- Após realizar o login o usuário irá entrar na tela que chamamos de dashboard que lista os caminhos existentes para que ele possa acessar as outras telas do sistema, como fazer venda ou manutenção de estoque e também um botão sair que fecha a sessão dele e retorna a tela de login:

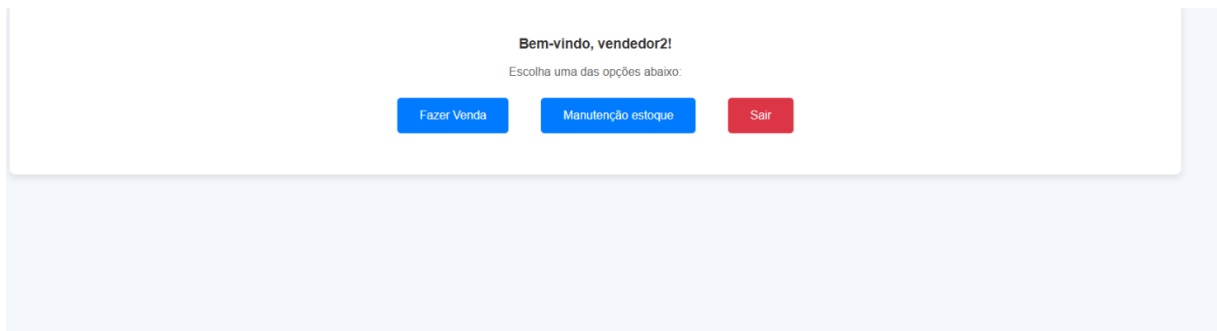


Imagem 2 - Dashboard

- A seguir na página Fazer Venda, podemos encontrar uma listagem dos produtos já cadastrados no banco de dados, demonstrando sua descrição, preço, estoque. Permite selecionar os produtos para vendas e selecionar sua quantidade. Produtos com estoque zero ficam destacados como sem estoque e não podem participar de vendas:

Produtos Disponíveis

Buscar por produto...

monitor

Preço: R\$ 300,00

Estoque: 2

☐ Selecionar para venda

Quantidade:

testeste

Preço: R\$ 100,00

Estoque: 0

Produto sem estoque

☐ Selecionar para venda

Quantidade:

mouse

Preço: R\$ 250,00

Estoque: 251

☐ Selecionar para venda

Quantidade:

Fazer Venda

Voltar

1

2

3

4

5

Imagem 3 – Vendas(Produtos disponíveis)

- Ao realizar uma venda o sistema apresenta uma mensagem de venda realizada, irá subtrair a quantidade vendida do estoque do item e atualizará q quantidade de estoque:

Produtos Disponíveis

coiled cable

Preço: R\$ 85,00

Estoque: 15

☐ Selecionar para venda
 Quantidade:

caderneta

Preço: R\$ 150,00

Estoque: 15

☐ Selecionar para venda
 Quantidade:

caderno hotwheels

Preço: R\$ 150,00

Estoque: 10

☐ Selecionar para venda
 Quantidade:

caderno123

Preço: R\$ 10,00

Estoque: 10

☐ Selecionar para venda
 Quantidade:

mouse12241

Preço: R\$ 100,00

Estoque: 50

☐ Selecionar para venda
 Quantidade:

cadernoteste

Preço: R\$ 100,00

Estoque: 14

☐ Selecionar para venda
 Quantidade:

headset

Preço: R\$ 500,00

Estoque: 0

Produto sem estoque

☐ Selecionar para venda
 Quantidade:

teste123456789

Preço: R\$ 100,00

Estoque: 11

☐ Selecionar para venda
 Quantidade:

play5

Preço: R\$ 100,00

Estoque: 5

☐ Selecionar para venda
 Quantidade:

Voltar

Venda(s) realizada(s) com sucesso!

1

2

3

4

5

Imagem 4 – Venda realizada

- Voltando ao dashboard e acessando a Manutenção de estoque, nos deparamos com um outro formulário que permite Cadastrar um produto não existente ou adicionar estoque/alterar preço de um produto já existente:

The image shows a web form titled "Cadastrar Novo Produto / Adicionar Estoque e Alterar Preço". It contains three input fields: "Nome do Produto:", "Preço:", and "Quantidade de Estoque:". Below these fields is a large blue button labeled "Adicionar Produto / Atualizar Estoque e Preço". At the bottom left of the form is a smaller, light gray button labeled "Voltar". The entire form is set against a light blue background.

Cadastrar Novo Produto / Adicionar Estoque e Alterar Preço

Nome do Produto:

Preço:

Quantidade de Estoque:

Adicionar Produto / Atualizar Estoque e Preço

Voltar

Imagem 5 – Manutenção de estoque

- Adicionando novo produto o sistema retorna uma mensagem de sucesso:

The image shows a web application interface with a light blue background. At the top, a green notification box displays the message "Produto adicionado com sucesso!". A red arrow points from the right side of the screen towards this message. Below the notification, there is a white form box with a blue header that reads "Cadastrar Novo Produto / Adicionar Estoque e Alterar Preço". The form contains three input fields: "Nome do Produto:", "Preço:", and "Quantidade de Estoque:". Below these fields is a large blue button labeled "Adicionar Produto / Atualizar Estoque e Preço". At the bottom left of the form is a smaller, light gray button labeled "Voltar".

Imagem 6 – Produto adicionado

- Se informarmos o nome de um produto já existente, o sistema irá identificar que ele já existe e com isso irá assumir o novo preço que o usuário informar e somar a quantidade de estoque informada ao estoque já existente:

The image shows a web application interface. At the top, a green banner displays the message "Estoque e/ou preço do produto atualizado com sucesso!". A red arrow points from the right side of the image towards this banner. Below the banner is a white form titled "Cadastrar Novo Produto / Adicionar Estoque e Alterar Preço". The form contains three input fields: "Nome do Produto:", "Preço:", and "Quantidade de Estoque:". Below these fields is a blue button labeled "Adicionar Produto / Atualizar Estoque e Preço". At the bottom left of the form is a grey button labeled "Voltar".

Imagem 7 – Estoque atualizado

3. Indexação

Em resumo os índices utilizam as partes principais de dados a partir de uma tabela para melhorar a capacidade de procura. A seguir demonstraremos os dois índices que utilizamos, um para melhorar a eficiência na consulta de produtos e o outro utilizando o BRIN para melhorar a eficiência em uma consulta que filtra vendas por datas.

```
CREATE INDEX idx_produtos_nome ON produtos (nome);
```

Como os nomes dos produtos são frequentemente consultados para verificar

existência ou para buscas exatas, um índice B-TREE acelera essas pesquisas.

```
1 CREATE INDEX idx_produtos_nome ON produtos (nome);  
2 select * from produtos
```

Imagem 8 – Index 1

E a outra com o índice BRIN a qual foi criada uma consulta específica para teste:

```
CREATE INDEX vendap ON vendas USING BRIN (data_venda)  
select * from vendas  
WHERE data_venda BETWEEN '2025-02-07 00:00:00' AND '2025-02-07 23:59:59'
```

Imagem 9 – Index 2

Onde listará as vendas realizadas em um determinado período de data/tempo ao qual o BRIN é mais eficiente, principalmente quando analisamos um contexto com uma grande massa de dados.

4. Processamento de transações e concorrência

Criação de duas transações disputando o mesmo recurso de forma simultânea:

Nesse exemplo abaixo podemos notar a respeito de duas transações de usuários diferentes disputando o mesmo produto.

Nesta imagem podemos notar que o prompt da esquerda realiza um update e o da direita também, porém o update não é realizado até que o da esquerda conclua a sua transação.

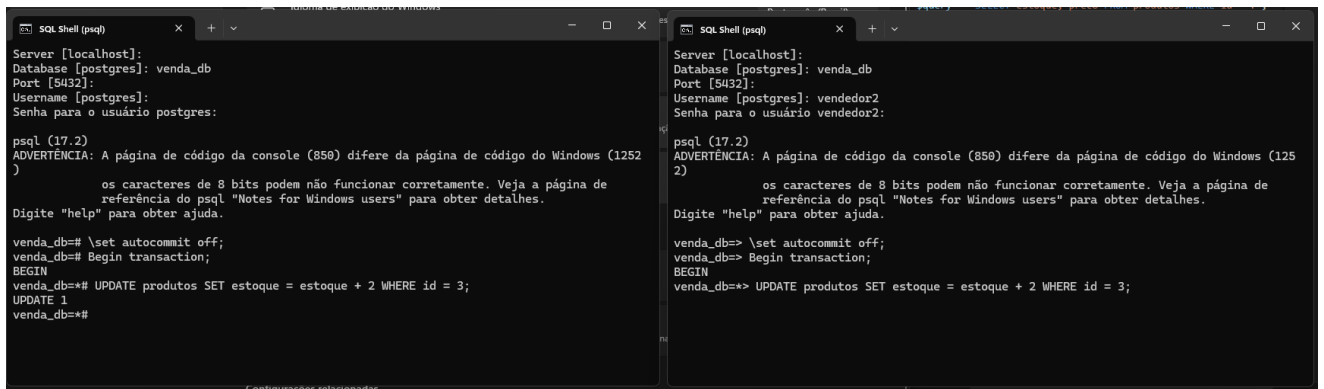


Imagem 10 – Transação 1

Nesta outra imagem, notamos que após o Commit, automaticamente a transação que estava aguardando (da direita) realiza o update no mesmo momento do commit da outra transação.

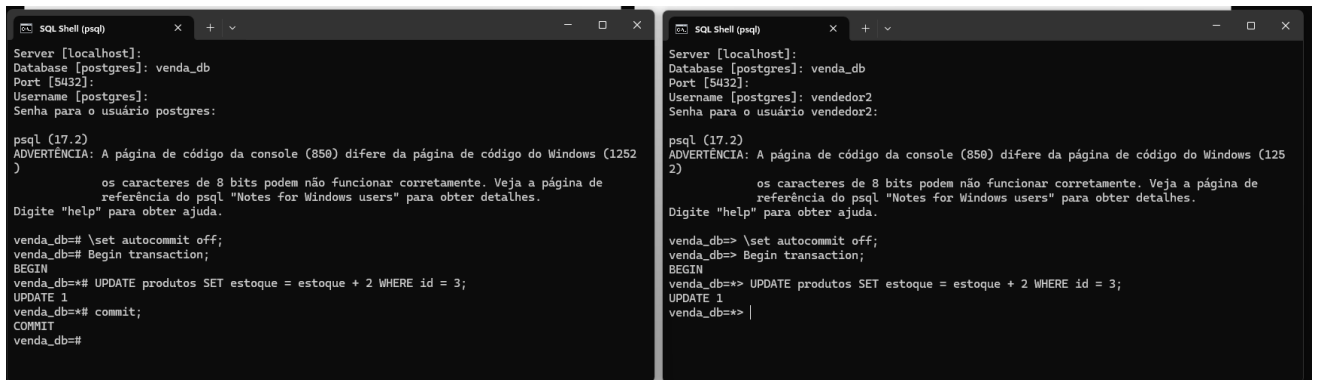


Imagem 11 – Transação 2

Dessa forma a transação da esquerda realizou o seu commit e a da direita também está liberada para realizar o commit dela, como na imagem abaixo:

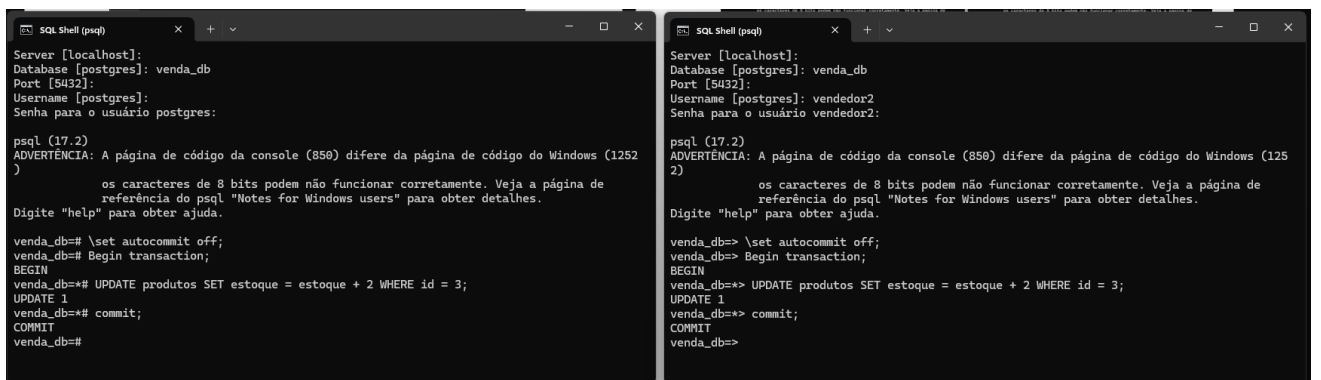


Imagem 12 – Transação 3

5. Recuperação

Uma das partes mais importantes quando se trata de dados está em manter os arquivos seguros e realizar cópias de segurança extras para evitar a perda dos dados.

No presente trabalho criamos um script .bat que dispara uma série de comandos ativando o pg_dump, que realiza o backup do banco de dados ao qual ele foi configurado, nesse caso o venda_db. Para que essa rotina pudesse ser automatizada de forma simples e eficaz foi criado um agendador de tarefas no sistema operacional, para que ele rodasse o script backup.bat todos os dias em determinado horário.

Logo como temos um script que com apenas dois cliques ele realiza o backup e salva no diretório configurado, o agendador de tarefas do windows irá executá-lo todo dia, mantendo uma rotina.

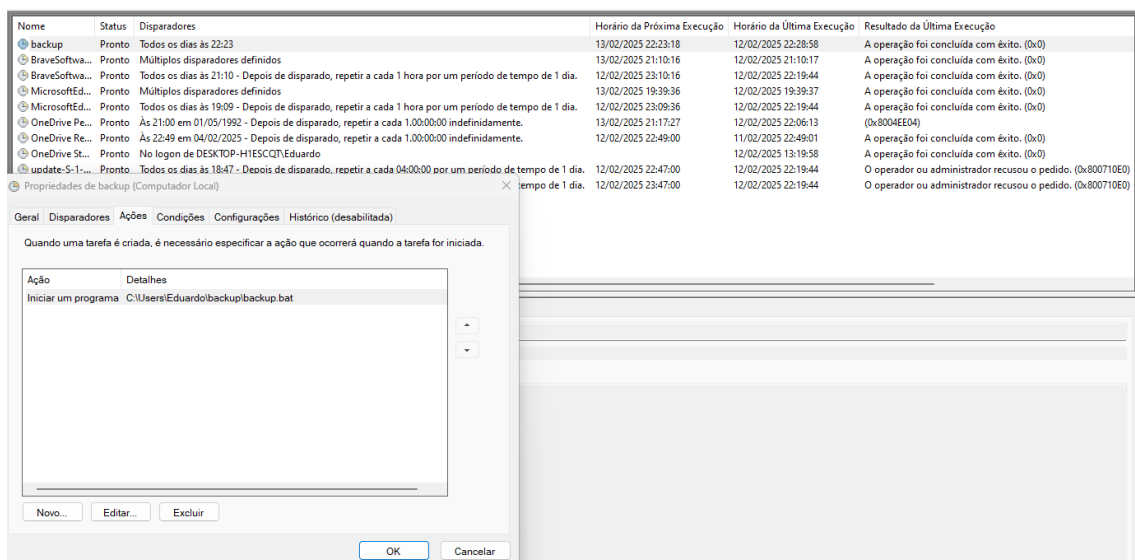


Imagem 13 – Agendador de tarefas

6. Segurança

Para tratarmos da segurança da nossa aplicação como um todo, utilizamos do mecanismo de segurança que o próprio SGDB nos proporciona que seria o DAC (Discretionary Access Control), onde o DBA, autoridade máxima do banco de dados, decide quem pode acessar ou modificar os dados de acordo com os privilégios concedidos ao usuário/grupo que ele pertence.

Para demonstrar criamos três usuários chamados vendedor, vendedor2 e gerente.

- Para iniciar primeiro tentamos realizar a autenticação do usuário

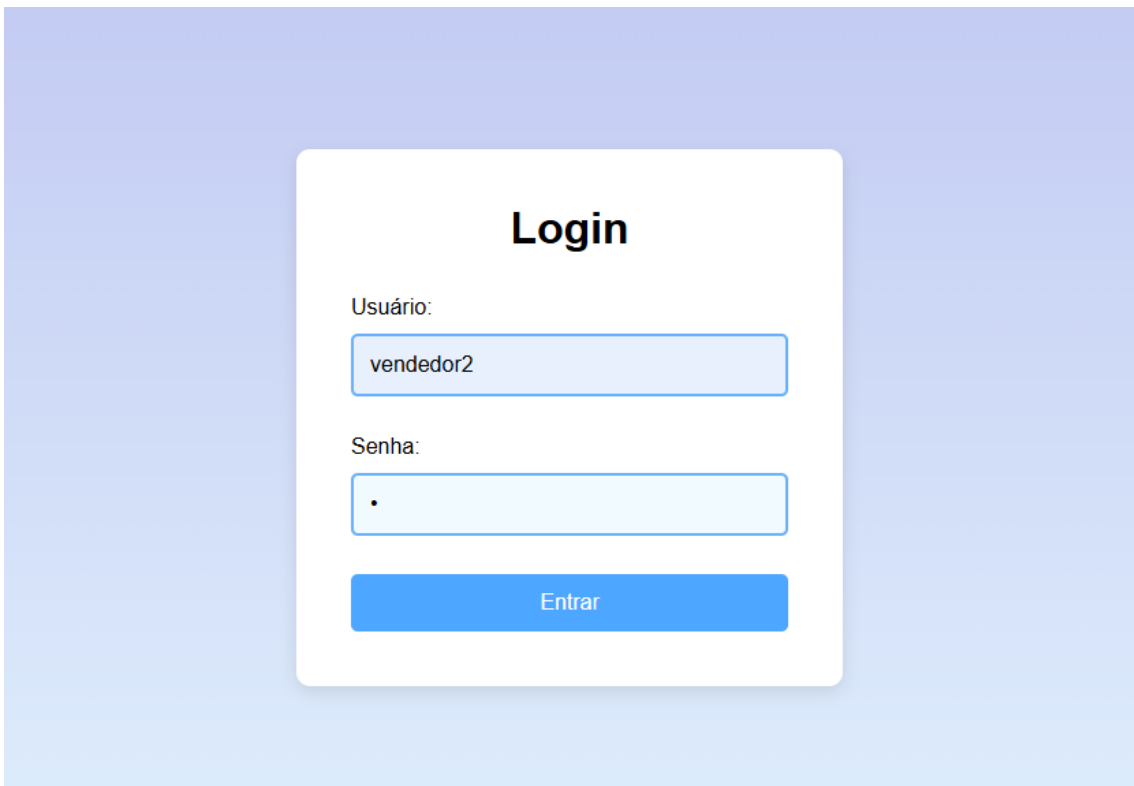
A screenshot of a login form titled "Login" centered on a light blue background. The form is a white rounded rectangle. It contains two input fields: the first is labeled "Usuário:" and contains the text "vendedor2"; the second is labeled "Senha:" and contains a single dot. Below the password field is a blue button with the text "Entrar" in white.

Imagem 14 – Login exemplo

- Detalhe que se tentarmos fazer login com usuário inexistente o sistema retorna um erro:

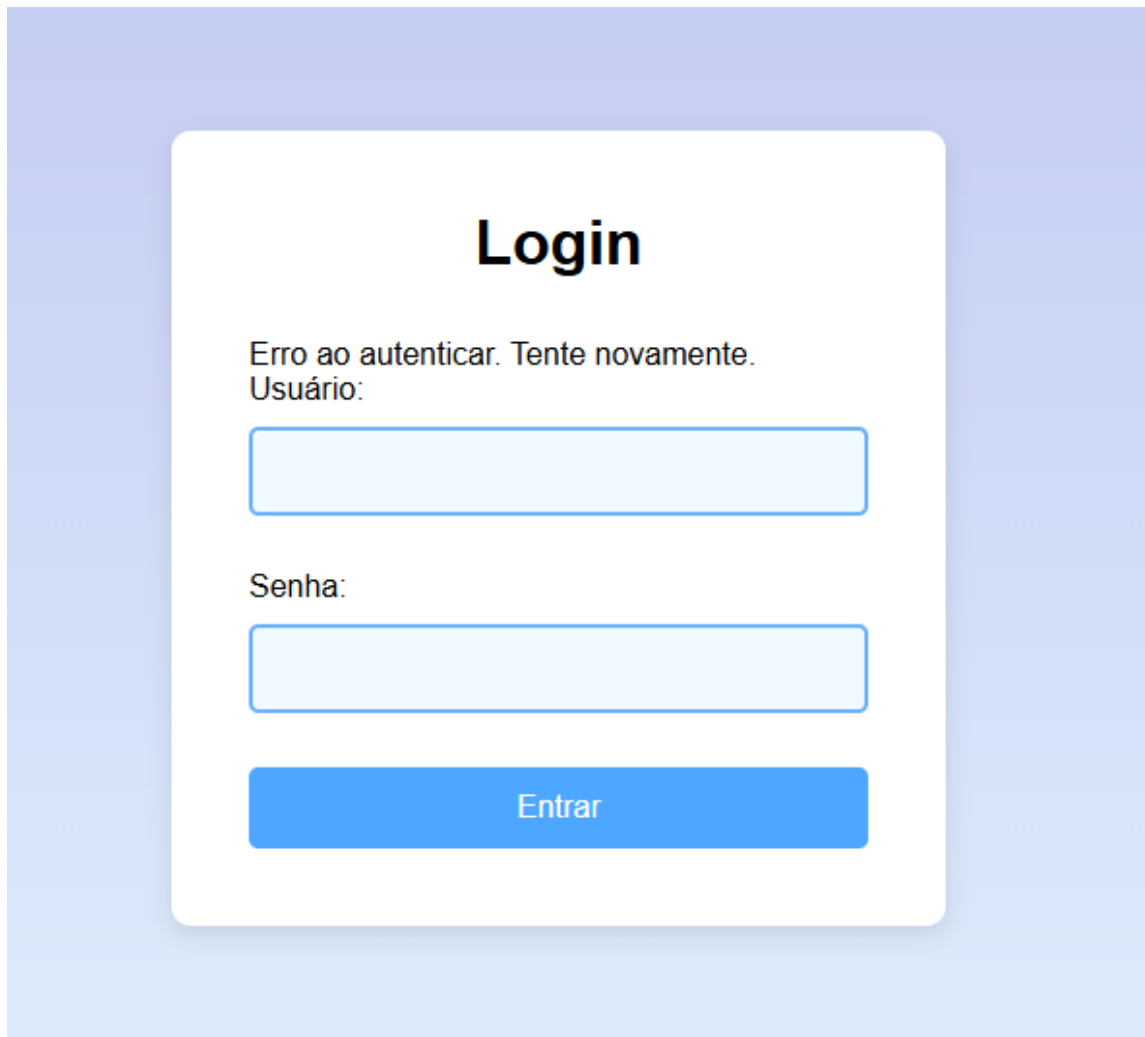


Imagem 14 – Login erro

- Iremos testar uma funcionalidade a qual o vendedor2 não tem acesso que é a adição de produtos, ao tentar inserir o sistema retorna uma exceção:

Fatal error: Uncaught PDOException: There is no active transaction in C:\xampp\htdocs\adicionar_produto.php:53 Stack trace: #0 C:\xampp\htdocs\adicionar_produto.php(53): PDO->rollBack() #1 {main} thrown in **C:\xampp\htdocs\adicionar_produto.php** on line **53**

- Mas o vendedor pode realizar vendas normalmente:

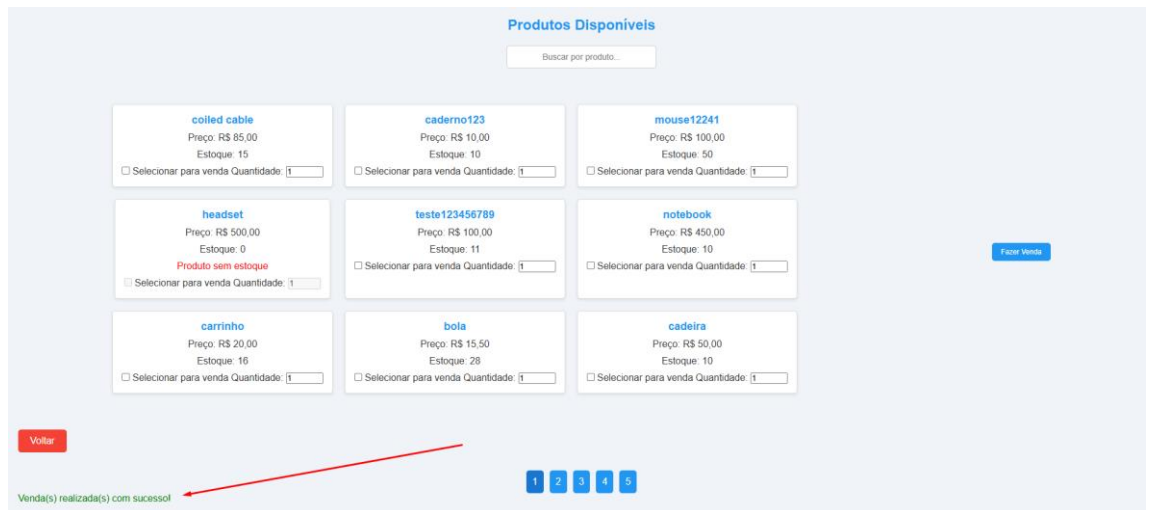


Imagem 15 – Privilégio exemplo

- Agora daremos o privilégio ao grupo_vendedor ao qual o vendedor2 pertence para que ele consiga realizar a ação impedida anteriormente

GRANT INSERT, ON produtos TO grupo_vendedor

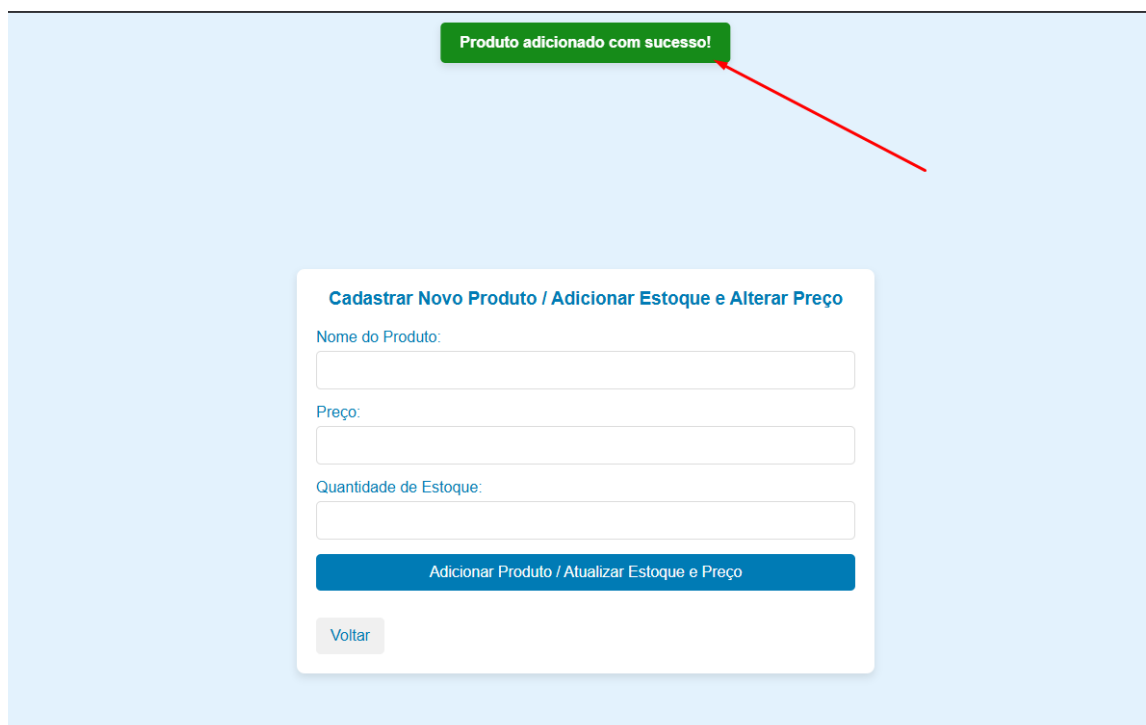
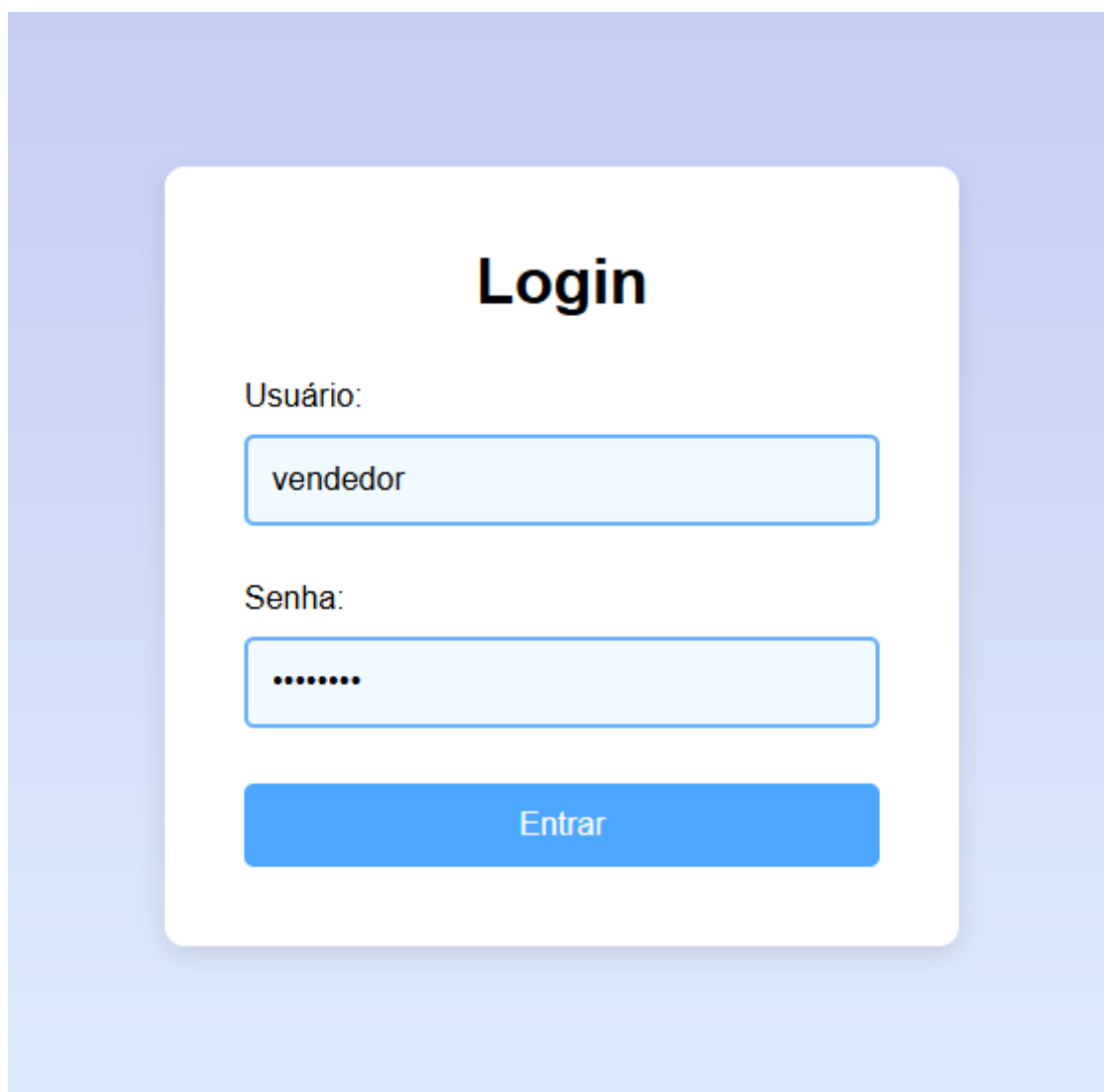


Imagem 16 – Exemplo privilégio concedido

- Testaremos também se o outro usuário do grupo_vendedor o vendedor,

poderá adicionar produtos também:

A login form titled "Login" is centered on a light blue background. The form is a white rounded rectangle with a subtle shadow. It contains two input fields: the first is labeled "Usuário:" and contains the text "vendedor"; the second is labeled "Senha:" and contains a masked password represented by eight dots. Below the password field is a solid blue button with the text "Entrar" in white.

Login

Usuário:

Senha:

Entrar

Imagem 17 – Login exemplo(2)

The image shows a web interface with a light blue background. At the top center, there is a green rectangular box with the text "Produto adicionado com sucesso!". A red arrow points upwards from the top of the form below to this message box. The form itself is a white rounded rectangle with a blue header "Cadastrar Novo Produto / Adicionar Estoque e Alterar Preço". It contains three input fields labeled "Nome do Produto:", "Preço:", and "Quantidade de Estoque:". Below these fields is a large blue button with the text "Adicionar Produto / Atualizar Estoque e Preço". At the bottom left of the form is a smaller, light gray button labeled "Voltar".

Produto adicionado com sucesso!

Cadastrar Novo Produto / Adicionar Estoque e Alterar Preço

Nome do Produto:

Preço:

Quantidade de Estoque:

Adicionar Produto / Atualizar Estoque e Preço

Voltar

Imagem 18 – Exemplo privilégio grupo

- E por fim acessaremos com o usuário gerente que tem acesso a aplicação toda:

Login

Usuário:

gerente

Senha:

•

Entrar

Imagem 19 – Login exemplo(3)

Produtos Disponíveis

Buscar por produto...

coiled cable

Preço: R\$ 85,00

Estoque: 15

☐ Selecionar para venda Quantidade:

mouse12241

Preço: R\$ 100,00

Estoque: 50

☐ Selecionar para venda Quantidade:

headset

Preço: R\$ 500,00

Estoque: 0

Produto sem estoque

☐ Selecionar para venda Quantidade:

teste123456789

Preço: R\$ 100,00

Estoque: 11

☐ Selecionar para venda Quantidade:

notebook

Preço: R\$ 450,00

Estoque: 10

☐ Selecionar para venda Quantidade:

carrinho

Preço: R\$ 20,00

Estoque: 16

☐ Selecionar para venda Quantidade:

bola

Preço: R\$ 15,50

Estoque: 28

☐ Selecionar para venda Quantidade:

cadeira

Preço: R\$ 50,00

Estoque: 10

☐ Selecionar para venda Quantidade:

mesa

Preço: R\$ 120,00

Estoque: 15

☐ Selecionar para venda Quantidade:

Voltar

Fazer Venda

Venda(s) realizada(s) com sucesso

1

2

3

4

5

Imagem 20 – Privilégio gerente

Produto adicionado com sucesso!

Cadastrar Novo Produto / Adicionar Estoque e Alterar Preço

Nome do Produto:

Preço:

Quantidade de Estoque:

Adicionar Produto / Atualizar Estoque e Preço

Voltar

Imagem 20 – Privilégio gerente (2)

7. Conclusão

Diante o exposto acima fica é notável a necessidade de certas medidas serem implementadas no banco de dados, entre elas, o sistema de indexação para realizar uma busca mais eficiente no banco de dados. Outro ponto fundamental para o trabalho foi o desenvolvimento da criação de uma rotina de backup para evitar possíveis perdas de dados, essa rotina de backup foi pensada para esse banco e sistema, por ser simples e eficiente, é importante comentar que há outros possíveis backups que poderiam ser utilizados. Por fim, foi desenvolvido a questão da segurança pensando no sistema DAC(Discretionary Access Control), que utiliza o conceito de papéis, agrupando usuários em funções. Assim, foi um trabalho com diversos pontos de extrema importância que devem ser implementados da melhor forma possível dependendo de cada banco de dados e o tipo de sistema.

Apêndice A – Link do Github para os arquivos da aplicação

<https://github.com/edubgn1/projetobd2/>

Apêndice B – Códigos para criação da estrutura do banco de dados

```
CREATE TABLE produtos (  
    id SERIAL PRIMARY KEY,  
    nome VARCHAR(255) NOT NULL,  
    preco DECIMAL(10, 2) NOT NULL,  
    estoque INT NOT NULL  
);  
  
CREATE TABLE vendas (  
    id SERIAL PRIMARY KEY,  
    id_produto INT NOT NULL,  
    quantidade INT NOT NULL,  
    total DECIMAL(10, 2) NOT NULL,  
    usuario_nome VARCHAR(255) NOT NULL,  
    data_venda TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
    FOREIGN KEY (id_produto) REFERENCES produtos(id)  
);
```

Apêndice C – Script de backup

```
@echo off  
REM Definir a data no formato DD-MM-YYYY  
set x=%DATE:~0,2%-~%DATE:~3,2%-~%DATE:~6,4%  
set date=%x%  
  
REM Definir o nome do arquivo de backup  
set BACKUP_DIR=C:\Users\eduardo\backup  
set BACKUP_FILE=%BACKUP_DIR%\Trabalho_Vendas_%date%.backup  
  
REM Criar o diretório de backup caso não exista  
if not exist "%BACKUP_DIR%" mkdir "%BACKUP_DIR%"  
  
REM Exibir o nome do arquivo de backup  
echo Backup file name is: %BACKUP_FILE%
```

```
REM Definir a senha do PostgreSQL
SET PGPASSWORD=12345
```

```
REM Executar o pg_dump para criar o backup
pg_dump -h localhost -p 5432 -U postgres -F c -b -v -f
"%BACKUP_FILE%" venda_db
```

pause

Apêndice D – Códigos usuários

```
A – CREATE USER gerente WITH PASSWORD '1';
CREATE USER vendedor WITH PASSWORD 'senha123';
```

```
B - CREATE ROLE grupo_gerente;
CREATE ROLE grupo_vendedor;
```

```
C - GRANT ALL PRIVILEGES ON TABLE produtos TO grupo_gerente;
GRANT ALL PRIVILEGES ON TABLE vendas TO grupo_gerente;
```

```
GRANT SELECT, UPDATE ON TABLE produtos TO grupo_vendedor;
GRANT SELECT, INSERT, UPDATE ON TABLE vendas TO
grupo_vendedor;
```

```
D - GRANT grupo_gerente TO gerente;
GRANT grupo_vendedor TO vendedor;
```

```
E –
GRANT INSERT ON TABLE produtos TO grupo_vendedor;
```