UNIVERSIDADE SALVADOR

CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

Alison Braz Santos - RA 1272116871
Carlos Alberto do Bomfim São Luiz Moreira - RA 1272118121
Gilson Sousa Silva - RA 1272118998
Lucas Vital Rocha - RA 1272116103
Thiago dos Santos -RA 1272117023

Sistemas Distribuídos e Mobile

Salvador-BA
2023

Relatório SOCKET de Vendas

Projeto apresentado como requisito parcial de avaliação da Unidade Curricular *Sistemas Distribuídos e Mobile* do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas da Universidade Salvador - UNIFACS.

Docentes: Wellington Lacerda S da Silva.

Eduardo Sidney da Silva Xavier.

Salvador-BA

2023.1

RESUMO

Este relatório especifica as funcionalidades da aplicação Socket de Vendas. Por meio deste, será possível tomar ciência de suas funcionalidades, quais ferramentas é necessário para utilizar a aplicação, qual linguagem foi utilizada e comentários dos códigos.

Apresentaremos de forma resumida e objetiva cada um dos clientes, especificando as funções e como utilizá-las.

O objetivo deste relatório tem como listar as funcionalidades solicitadas, deixando de forma clara e objetiva os requisitos. O presente relatório visa ser escrito de forma abrangente e clara para que possa ser utilizado por todos que compõem o time e ainda ser compreendido por aqueles que tenham interesse .

SUMÁRIO

1 INTRODUÇÃO

A aplicação socket de vendas é um sistema que permite o registro de vendas por meio de uma comunicação cliente-servidor utilizando sockets. A aplicação possui cinco clientes/vendedores, um vendedor/genérico, um cliente/gerente, um servidor principal, uma classe para geração de eleição, uma comunicação entre servidores e um banco de dados SQLite para armazenamento das vendas.

Os cincos vendedores estão distribuídos em três lojas, são dois vendedores para a loja 1, dois vendedores para a loja 2 e um vendedor para a loja 3.

O vendedor genérico realiza a venda quando algum dos vendedores se torna um servidor, servindo como apoio para que nenhum dos vendedores perca o registro de suas vendas.

Todas as vendas são registradas no banco de dados SQLite com a comunicação de socket e servidores, os servidores estão divididos em servidor principal e servidores temporários, caso o servidor principal fique fora do ar, o servidor temporário assume as operações até o servidor principal retornar, através de um processo de eleição.

O gerente pode consultar todas as vendas realizadas, ter dados de um vendedor, saber qual vendedor realizou um volume maior de vendas, qual loja tem a maior venda, consultar vendas por um filtro de período, todos os dados são consultados através do banco de dados, onde tem todos os registros das operações.

O código da aplicação está dividido em diferentes partes para melhor organização e funcionalidade.

A linguagem utilizada na construção da aplicação socket de vendas foi o Python. O critério para a escolha da linguagem Python, foi por atender melhor o tipo de aplicação, facilitando na sua construção e no desempenho do desenvolvimento.

2 Ferramentas para utilização da aplicação

Para que seja possível utilizar a aplicação Socket de Vendas é necessário ter todos os códigos no mesmo repositório ou pasta. A pasta que compõem os códigos devem ser baixadas no seu computador para melhor funcionalidade, não é possível utilizá-las em mobile, pois o mesmo não faz parte dos requisitos e não foi implementado.

Certifique-se de ter **Python** instalado em seu sistema, deve ter a extensão para o BD **SQLite 3.**

Antes de iniciar verifique se existe na pasta da aplicação o registro do banco de dados "**dados_venda.db**", Você deve excluí-la, para não ter lixo de dados.

Você deve iniciar executando o servidor em **servidor.py** para que as operações sejam registradas, caso o servidor não seja executado, o vendedor em execução irá assumir como servidor temporário.

Não é possível manter dois servidores em execução, caso deseje retornar ao servidor principal, deve parar a execução do servidor temporário.

Você pode executar em uma IDE Python, ou no terminal, ou no prompt, ou ainda diretamente na pasta clicando em cada um das classes, pois os mesmo já se encontra com a extensão **.py,** lembrando que o seu computador deve está configurado para aceitar extensões.

As resposta dos comandos são imediatas, caso ocorra um tempo maior que 30 segundos para que um registro de uma venda ou qualquer tipo de consulta seja realizada, aconselho fechar todas as telas e iniciar o processo novamente, verifique se a memória do seu equipamento não está sobrecarregada.

No repositório do GitHub tem um vídeo com o tutorial completo de como utilizar nossa aplicação SOCKET de Vendas.

3 Clientes/Vendedores

Existe cinco vendedores, sendo, dois vendedores para a loja de Nº 1, identificados como; Vendedor 1 e Vendedor 4, já para a Loja de Nº 2, os dois vendedores são; Vendedor 2 e Vendedor 5, para a Loja de Nº 3 existe um único

vendedor identificado como; Vendedor 3.

3.1 Funcionalidades;

Registro de Venda: Cada cliente/vendedor pode registrar uma venda fornecendo informações como o nome do vendedor, ID da loja, data da venda e valor da venda.

3.1.1 Cadastrar Vendas;

Para realizar a operação de vendas no sistema é necessário seguir os passos da ferramenta de utilização. Na opção que aparecerá no cursor da sua tela, você deve digitar 1 para a opção de Cadastrar Vendas. Digite a data seguindo o Dia, Mês e Ano, separando por barras “ / ” e em seguida irá solicitar o valor, digite normalmente, não serão aceitos valores negativos.

3.1.2 Sair;

Caso já tenha terminado os registros, ou simplesmente não deseje realizar nenhum registro, é só digitar a opção zero “ 0 “ para Sair, o vendedor será encerrado, mas não se preocupe com os registros realizados, todos estão salvos no Banco de Dados.

4 Vendedor/Genérico

O vendedor genérico tem a função de registrar uma venda para qualquer um dos cinco vendedores, é utilizado quando o servidor principal está inativo e tem a necessidade de um dos vendedores assumir como servidor temporário.

4.1 Funcionalidades;

Registro de Venda: Cada cliente/vendedor pode registrar uma venda fornecendo informações como o nome do vendedor, ID da loja, data da venda e valor da venda.

4.1.1 Cadastrar Vendas;

Para realizar a operação de vendas no sistema é necessário seguir os passos da ferramenta de utilização. Na opção que aparecerá no cursor da sua tela, você deve digitar 1 para a opção de Cadastrar Vendas. Digite o nome do vendedor, o ID da loja, a data seguindo o Dia, Mês e Ano, separando por barras “ / ” e em seguida irá solicitar o valor, digite normalmente, não serão aceitos valores negativos.

4.1.2 Sair;

Caso já tenha terminado os registros, ou simplesmente não deseje realizar nenhum registro, é só digitar a opção zero “ 0 “ para Sair, o vendedor será encerrado, mas não se preocupe com os registros realizados, todos estão salvos no Banco de Dados.

5 Cliente/Gerente

O Gerente é responsável por gerenciar todas as vendas dos respectivos vendedores e lojas, por meio do Gerente é possível ter os dados de vendas de um vendedor, dados de vendas da rede de lojas, podendo também separar por período, e ainda visualizar quem é o melhor vendedor e qual loja tem o maior volume de vendas.

5.1 Funcionalidades;

O Gerente tem seis opções dentro das suas funcionalidades, por meio de comunicação com o servidor, é possível consultar ao banco de dados as operações realizadas pelos vendedores.
Na função **menu** é exibido o menu de opções para o cliente/gerente, por lá são fornecidas informações adicionais, como os nomes dos vendedores e os IDs das lojas.
Pela a função de Consulta são fornecidas diversas funções que correspondem às opções do menu. Cada função solicita as informações necessárias ao usuário e chama a função **consulta_bd** para enviar a consulta ao servidor.

5.1.1 Consultar Banco de Dados;

A consulta ao banco de dados é feita através da função; **consulta_bd:** Essa função é responsável por enviar uma consulta para o servidor, solicitando informações sobre as vendas. Ela recebe como parâmetros o código da operação e os dados relacionados à consulta, como o nome do vendedor, o ID da loja, ou as datas de início e fim. A função cria uma mensagem em formato JSON com esses dados e a

envia para o servidor principal através de uma conexão socket. Em seguida, recebe a resposta do servidor e a exibe na tela.

5.1.1.1 Total de vendas de um vendedor;

Para consultar o total de vendas de um vendedor, deve digitar a opção 1 em seguida o nome correto do vendedor, o nome do vendedor está no espelho do cursor, certifique-se em usar todos caracteres conforme a referência.

5.1.1.2 Total de vendas de uma loja;

Escolha a opção 2 para consultar o total de vendas de uma loja, em seguida será solicitado o ID da loja, digite o ID correspondente a loja que deseja obter as informações.

5.1.1.3 Total de vendas de toda a rede de lojas por período;

Para obter os dados de venda da rede de lojas por um período é simples, escolha a opção 3 e em seguida digite a data inicial e a data final separados por barras “/” conforme ilustrado na aplicação.

5.1.1.4 Melhor vendedor;

Digite a opção 4 para obter informações do melhor vendedor, a aplicação retornará qual vendedor realizou maior volume de vendas durante todo o período de funcionamento da rede.

5.1.1.5 Melhor loja;

Nesta opção digitando 5 a aplicação retornará qual loja teve o maior valor de vendas em todo o período.

5.1.2 Sair;

Terminou os trabalhos? Na opção zero “0” finaliza a aplicação.

6 Servidor

O código do servidor principal é responsável por receber as conexões dos clientes, processar as solicitações recebidas e responder com os resultados das consultas ou com uma mensagem de erro, quando aplicável.

O servidor realiza a conexão cliente/servidor através do socket, as requisições são feitas por várias funções, onde realiza o registro de vendas dos vendedores e as consultas realizadas pelo gerente.

6.1 Funções do Código;

6.1.1 Função **validar_venda**;
Essa função é responsável por receber uma mensagem de venda do cliente/vendedor, validar os dados da venda e inseri-la no banco de dados. A função extrai as informações da mensagem e realiza a validação dos campos. Em seguida, estabelece uma conexão com o banco de dados SQLite, insere os dados da venda na tabela correspondente e retorna uma mensagem de confirmação ou erro.
6.1.2 Função **msg_consulta_bd**;
Essa função é responsável por receber uma mensagem de consulta do cliente/gerente, realizar a consulta no banco de dados e retornar o resultado. A função extrai o código da operação e os dados da mensagem de consulta. Em seguida, estabelece uma conexão com o banco de dados SQLite, executa a consulta desejada e retorna o resultado da consulta ou uma mensagem de erro.
6.1.3 Função **cnx_cliente**;
Essa função é responsável por estabelecer a comunicação com o cliente, receber a mensagem enviada pelo cliente e encaminhá-la para a função correspondente, seja para validar uma venda ou para realizar uma consulta. A resposta obtida é enviada de volta ao cliente.
6.1.4 Função **start_server**;
Essa função inicia o servidor socket, estabelecendo uma conexão na porta especificada. Em seguida, o servidor aguarda por conexões de clientes. Quando uma conexão é estabelecida, a função **cnx_cliente** é chamada para tratar a comunicação com o cliente. Após a conclusão da comunicação, a conexão é encerrada.

7 Banco de Dados

O banco de dados SQLite é responsável por armazenar os registros de

vendas realizados pelos vendedores no sistema. Ele cria uma tabela "venda" com os campos adequados, valida os dados de uma venda e registra a venda no banco de dados, o código estabelece uma conexão com o BD através da função **sqlite3.connect().**

Foi criado uma tabela de vendas na função **execute()** do objeto **cursor,** A tabela possui os seguintes campos:

id: INTEGER, chave primária, autoincrementável.
nome_vendedor: TEXT, armazena o nome do vendedor.
loja_id: TEXT, armazena o identificador da loja.
data_venda: TEXT, armazena a data da venda.
valor_venda: REAL, armazena o valor da venda.

8 Eleição

O servidor utiliza a classe **Eleicao** para gerenciar a eleição de um servidor temporário. A classe possui métodos para iniciar o servidor temporário, verificar o servidor principal e eleger o servidor temporário.

Se o servidor principal estiver ativo, ele processa as operações de venda e consultas normalmente.

Caso contrário, se o servidor principal estiver inativo, o cliente recebe uma mensagem de erro ao tentar realizar uma operação de venda ou consulta. Em seguida, é iniciado o processo de eleição, onde o servidor temporário se comunica com o servidor principal e é eleito como o novo servidor temporário. As operações de venda que ocorreram durante a ausência do servidor principal são armazenadas em uma lista chamada **operacoes_venda_temporario** e são exibidas quando o servidor principal é ativado novamente.

Além disso, a função **validar_venda** foi modificada para verificar se o servidor principal está ativo antes de inserir a venda no banco de dados. Se o servidor principal estiver ativo, a venda é registrada normalmente; caso contrário, é retornado um status "Ativo" para indicar que o servidor está em execução naquele vendedor, mas não pode realizar operações de venda no momento, a operação de venda deve ser realizada pelo o vendedor genérico em 4.1.1 Cadastrar Vendas.

9 Comunicação Entre Servidores

Essa classe realiza a comunicação entre o servidor principal e o servidor temporário

no processo de eleição para verificar se o servidor principal está ativo através da classe **ComunicacaoServidores.**

10 Principais Seções do Código

A seguir, são descritas as principais seções do código:

10.1 Importações;

São realizadas algumas importações necessárias para o funcionamento da aplicação, incluindo o módulo **socket** e o módulo **json** e algumas classes importantes.

10.2 Constantes;

São definidas as constantes **HOST** e **PORT**, que representam o endereço IP e a porta utilizada para a conexão socket.

10.3 Funções;

10.3.1 **enviar_venda**; Essa função é responsável por enviar os dados de uma venda para o servidor. Ela recebe como parâmetros o nome do vendedor, o ID da loja, a data da venda e o valor da venda. A função cria uma mensagem em formato JSON com esses dados e a envia para o servidor através de uma conexão socket.

10.3.2 **op_eleicao**; Caso o servidor esteja indisponível, a função inicia o processo de eleição, importando a classe **Eleicao** do módulo **op_eleicao**.

10.3.3 **menu**; Essa função exibe o menu de opções para o cliente/vendedor. No caso do código fornecido, existe um menu específico para os vendedores que possui a opção de registrar uma venda e sair do programa e um menu específico para o cliente/gerente que possui outras funcionalidades.

10.3.4 **registrar_venda**; Essa função é chamada quando o usuário escolhe a opção de registrar uma venda no menu. Ela solicita ao usuário as informações necessárias para o registro da venda e chama a função **enviar_venda** para enviar os dados para o servidor.

10.4 Loop Principal:

O código principal consiste em um loop infinito que exibe o menu, solicita a escolha do usuário e realiza a ação correspondente. Caso a opção seja registrar uma venda, a função **registrar_venda** é chamada. Caso a opção seja sair, o loop é encerrado e o programa é finalizado.

O código principal consiste em um loop infinito que exibe o menu, solicita a escolha do usuário e realiza a ação correspondente. Caso a opção seja realizar uma consulta, a função correspondente é chamada. Caso a opção seja sair, o loop é encerrado e o programa é finalizado.

10 Conclusão

O socket de vendas utiliza uma comunicação cliente-servidor por meio de sockets. A aplicação consiste em cinco vendedores distribuídos em três lojas, um vendedor genérico, um cliente gerente e um servidor principal. A aplicação usa um banco de dados SQLite para armazenar as vendas.

Os vendedores podem registrar vendas fornecendo informações como nome do vendedor, ID da loja, data da venda e valor da venda. O vendedor genérico atua como suporte para os vendedores caso o servidor principal fique inativo e um dos vendedores se torne o servidor, permitindo que este vendedor não perca o registro de suas vendas.

O cliente gerente tem acesso a funcionalidades que permitem consultar todas as vendas, obter dados de um vendedor específico, verificar o vendedor com o maior volume de vendas, identificar a loja com a maior venda e filtrar as vendas por período. Essas consultas são realizadas por meio do banco de dados que armazena todos os registros das operações.

O relatório também fornece informações sobre as ferramentas necessárias para utilizar a aplicação, como ter o Python instalado, a extensão SQLite3 e ter todos os códigos no mesmo repositório ou pasta. Além disso, descreve o funcionamento das diferentes partes do código, como os clientes/vendedores, o vendedor genérico, o cliente gerente, o servidor e a comunicação entre servidores.

O banco de dados SQLite é responsável por armazenar os registros de vendas, a classe Eleicao gerencia o processo de eleição de um servidor temporário quando o servidor principal está inativo.

Por meio deste relatório é fornecido uma visão geral da aplicação de vendas, descrevendo suas funcionalidades, as partes envolvidas e as ferramentas necessárias para utilizá-la.