Instituto Mauá de Tecnologia

CAMPUS São caetano do sul

engenharIa de computação

sistema para transferência de arquivos:

twodrive

Pedro Henrique de Sousa Matumoto - 21.00784-5

Erick Eiji Nagao - 21.00690-3

Kaiven Yang Su – 20.02146-0

Raphael Fernandes Raymundo – 21.00334-3

Matheus Santos Feitosa – 20.00628-4

são caetano do sul

2023

RESUMO

O trabalho a seguir foi desenvolvido a partir dos conhecimentos obtidos no 3° ano do curso de graduação em Engenharia da Computação do Instituto Mauá de Tecnologia no curso de Linguagens de Programação I, lecionado pelos Professores Igor Cataneo e Robson Calvetti. Para verificar e aplicar os conceitos vistos em aula, foi proposto um projeto semestral de fechamento do curso que requisitava a aplicação de funcionalidades específicas como forma de verificação de aprendizado. O projeto foi iniciado de maneira flexível, permitindo ajustes conforme as necessidades surgidas durante seu desenvolvimento. O trabalho consiste em um sistema desenvolvido em linguagem Java, possuindo dois programas principais: um para o cliente e outro para o servidor, comunicando-se entre si para compartilhar arquivos e sem necessidade de aplicativos terceiros. Avalia-se o resultado como satisfatório, porém sempre passível de aprimoramentos.

Palavras-chave: linguagens, programação, Java, cliente, servidor.

SUMÁRIO

[1 introdução 4](#_Toc1615402699)

[2 Objetivo 5](#_Toc509203548)

[3 desenvolvimento 6](#_Toc437032495)

[3.1 bibliotecas 7](#_Toc1407867755)

[3.2 Arquivos onde serão estanciadas as classes 8](#_Toc1695970594)

[3.3 Explicação de cada arquivo 10](#_Toc932073618)

[3.3.1 Explicação de cada arquivo na parte do servidor 10](#_Toc1171293161)

[3.3.2 Explicação de cada arquivo na parte do cliente 11](#_Toc1771773922)

[4 DIFICULDADES 13](#_Toc319187636)

[5 Considerações finais 14](#_Toc696978449)

[REFERÊNCIAS 15](#_Toc1898771831)

# introdução

O Instituto Mauá de Tecnologia, dentro do curso de Engenharia da Computação, oferece aos alunos a matéria “Linguagens de Programação I”, cujo objetivo é ilustrar e exemplificar conceitos na linguagem orientada à objeto Java (tópico abordado posteriormente na matéria). A disciplina traz em seu escopo desde conceitos básicos, como entrada e saída, até temas mais avançados, como cliente-servidor, interface gráfica e banco de dados. Para fixação dos conceitos, listas de exercícios são oferecidas a cada novo tópico abordado, além de 2 trabalhos semestrais a serem entregues a cada 6 meses. Para o 2° semestre do curso em 2023, propôs-se um projeto integrando todos os conceitos vistos em aula.

Nesse contexto, o presente grupo buscou não só entregar a tarefa requisitada, mas trazer um trabalho passível de aplicação em problemas reais – facilitando, assim, a materialização do aprendizado e a reflexão crítica e prática dos conceitos aprendidos. O projeto foi iniciado com um *brainstorm*, e selecionou-se um sistemapara transferência de arquivos que atendia à dois requisitos: (i) permitiria a aplicação extensiva e diversa (mas não exaustiva) dos conteúdos da disciplina; (ii) depois de finalizado, teria uma aplicação real no dia a dia. A elaboração do conceito do projeto veio com a idealização de um programa que não necessitasse de ferramentas externas ou que fosse mantido exclusivamente na máquina do cliente, como em um banco de dados MySql.

# Objetivo

Aplicar os conceitos ensinados em aula a partir do desenvolvimento de um sistema utilizando a matéria desenvolvida na teoria e nos exercícios da disciplina e utilizando o máximo de tópicos abordados possível. Procura-se, nesse trabalho, unificar, fixar e exercitar a criatividade e as competências técnicas de cada aluno. O produto final do projeto também foi idealizado para que fosse disponibilizado e utilizado publicamente, quando pronto, como base para plataformas de transferência de arquivos.

# desenvolvimento

## bibliotecas

O desenvolvimento do projeto se deu a partir de algumas bibliotecas, compiladas a seguir:

import DTO.\*;

import java.awt.Desktop;

import java.awt.image.BufferedImage;

import java.io.BufferedOutputStream;

import java.io.BufferedReader;

import java.io.ByteArrayInputStream;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.sql.ResultSet;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.io.InputStream;

import java.io.PrintWriter;

import javax.swing.ImageIcon;

import javax.swing.JFileChooser;

import javax.swing.JOptionPane;

import javax.imageio.ImageIO;

import javax.sound.sampled.AudioFileFormat;

import javax.sound.sampled.AudioFormat;

import javax.sound.sampled.AudioInputStream;

import javax.sound.sampled.AudioSystem;

import javax.sound.sampled.Clip;

import javax.sound.sampled.LineEvent;

import javax.sound.sampled.LineListener;

import javax.swing.JFrame;

import javax.swing.JLabel;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Blob

import java.io.\*;

import javax.crypto.\*;

import javax.crypto.spec.\*;

import java.security.\*;

import java.security.cert.\*;

import java.net.\*;

import java.util.\*;

A biblioteca javax.swing foi utilizada para criar as janelas, inserir os botões interativos, os campos de texto e as etiquetas que indicam os campos de texto. A ImageIcon foi usada para inserir a imagem inicial no JFrame.

A biblioteca java.awt foi utilizada para construir e estilizar o JFrame (janela de interação com o usuário), colocando os layouts que melhor se adequavam às requisições. O java.awt.event foram utilizados para implementar o funcionamento dos botões.

A biblioteca java.sql foi utilizada para integrar as interações do JDK e do banco de dados MySQL – que, junto ao driver para conexão, são responsáveis pelo desenvolvimento do banco de dados em si. O uso do MySQL obedece à diretriz obrigatória proposta pelos professores responsáveis.

As bibliotecas javax.crypto e java.security são necessárias para a criptografia das senhas armazenadas no banco de dados. A biblioteca java.util e java.net são responsáveis por implementar a seção de cliente servidor, também diretriz obrigatória para a realização do projeto.

## Arquivos onde serão estanciadas as classes

Figura 1 ‒ Captura de tela para ilustrar os arquivos da parte do servidor

Interface gráfica do usuário, Texto, chat ou mensagem de texto

Descrição gerada automaticamente

Figura 2 – Primeira captura de tela para ilustrar os arquivos da parte do cliente

Texto

Descrição gerada automaticamente

Figura 3 – Segunda captura de tela para ilustrar os arquivos da parte do cliente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente com confiança média

*Figura 4 –* Captura de tela para ilustrar os arquivos para internacionalização

Texto

Descrição gerada automaticamente

*Figura 5 –* Imagens utilizadas para construir as interfacesTexto

Descrição gerada automaticamente

## Explicação de cada arquivo

Cada arquivo é responsável por instanciar métodos referentes à algum setor do projeto.

### Explicação de cada arquivo na parte do servidor

* CommandsDB

Classe responsável por lidar com toda a parte relacionada com banco de dados, como executar *statements* de update, create, delete e select (CRUD). As validações de login, as operações de seleção do arquivo a ser baixado, são exemplos de operações implementadas nessa classe.

* ConnFactory

A classe ConnFactory proporciona métodos para estabelecer, fechar e gerenciar conexões com um banco de dados MySQL. Ela utiliza o padrão Singleton para garantir uma única instância da conexão durante a execução do programa. A classe encapsula detalhes da conexão, como URL, usuário e senha, fornecendo métodos simples como getConn() para obter a conexão ativa, closeConn() para fechá-la e métodos adicionais para operações mais complexas. Se ocorrerem problemas na conexão, a classe lança exceções apropriadas para garantir um tratamento adequado de erros durante a execução do programa. Essa abstração facilita o acesso e o gerenciamento de conexões a bancos de dados MySQL em aplicações Java.

* GeneralData

A classe GeneralData encapsula dados relacionados a um usuário, incluindo informações como nome de usuário, email e senha. Além disso, ela suporta dados associados a áudio, texto e imagem. A classe inclui métodos para acessar e modificar esses dados, bem como um método toString() para fornecer uma representação de string útil para depuração. Esta abstração facilita a manipulação e transferência de dados em aplicações Java, fornecendo uma estrutura coesa para organizar e acessar informações diversas.

* Diretório doclogs

Diretório onde são armazenados em formato de arquivo texto todos os documentos presentes no banco de dados assim que uma requisição é feita. O nome do arquivo diz o ano, dia do ano, hora, minuto e segundo onde foi criado o mesmo.

* Chave simétrica

Arquivo criado pela classe CryptoAES, contém as informações necessárias para criptografar ou descriptografar uma cadeia de caracteres.

* CryptoAES

A classe CryptoAES em Java proporciona funcionalidades de criptografia e descriptografia utilizando o algoritmo AES (Advanced Encryption Standard). A chave simétrica é armazenada em um arquivo, sendo necessário carregá-la para operações subsequentes. A classe faz uso de fluxos de objetos para serializar e deserializar objetos, como a chave simétrica. No sistema, apenas o servidor tem posse da chave simétrica, sendo sempre a mesma para criptografar e armazenar tudo no banco de dados

* Impressora

Em conjunto com a classe CryptoAES, tem métodos para transformar Strings em bytes hexadecimais para fazer as criptografias

* MainApp

Programa principal, que deve ser executado para criar uma instância de servidor no IP em questão, este conectando-se com o banco de dados e realizando internamente (sem interação alguma do cliente) todas operações necessárias.

* ServidorBatePapo

A classe ServidorBatepapo é responsável por gerenciar a comunicação entre clientes em um aplicativo de bate-papo. Utilizando sockets, ela estabelece uma conexão com os clientes e gerencia a troca de mensagens. Além disso, o servidor lida com operações relacionadas a um banco de dados MySQL, como autenticação de usuários, envio e recebimento de arquivos de áudio, texto e imagem.

A classe incorpora recursos como o envio e recebimento de arquivos, criptografia AES para proteger as informações sensíveis durante a transmissão e operações de manipulação de texto. A lógica de negócios inclui funcionalidades como login, cadastro, remoção e envio de arquivos, bem como a capacidade de baixar esses arquivos.

O servidor é projetado para ser escalável, suportando múltiplos clientes simultaneamente por meio de threads. Ele utiliza uma classe de manipulação de texto e uma classe de acesso ao banco de dados para encapsular funcionalidades específicas.

Em resumo, a classe ServidorBatepapo fornece uma base sólida para a implementação de um servidor de bate-papo robusto e seguro em Java, com recursos avançados de comunicação e gerenciamento de dados.

* SocketCliente

A classe SocketCliente em Java encapsula operações básicas de comunicação de um cliente por meio de sockets. Ela possui métodos para enviar e receber mensagens, além de gerenciar o fechamento adequado dos recursos do socket. A classe utiliza BufferedReader e PrintWriter para leitura e escrita de dados nos fluxos de entrada e saída do socket. Exceções de IO são tratadas para garantir uma operação robusta. O construtor imprime informações sobre a conexão e inicializa os fluxos de entrada e saída.

* TextManipulator

Classe responsável por escrever ou ler arquivos texto, ela é responsável por geral os logs dos arquivos mencionados anteriormente.

### Explicação de cada arquivo na parte do cliente

* ClienteBatePapo

A classe ClienteBatepapo em Java é parte de um sistema de bate-papo cliente-servidor usando sockets. Seu método principal, start, inicia a comunicação, conecta-se ao servidor, envia e recebe mensagens, exibindo a mensagem recebida. A classe utiliza um objeto SocketCliente para gerenciar a conexão. Métodos como send e getMessage facilitam o envio e recebimento de mensagens. A estrutura modular permite uma implementação flexível e extensível para aplicações de bate-papo.

* CryptoAES

A classe CryptoAES em Java proporciona funcionalidades de criptografia e descriptografia utilizando o algoritmo AES (Advanced Encryption Standard). A chave simétrica é armazenada em um arquivo, sendo necessário carregá-la para operações subsequentes. A classe faz uso de fluxos de objetos para serializar e deserializar objetos, como a chave simétrica. No sistema, apenas o servidor tem posse da chave simétrica, sendo sempre a mesma para criptografar e armazenar tudo no banco de dados

* Impressora

Em conjunto com a classe CryptoAES, tem métodos para transformar Strings em bytes hexadecimais para fazer as criptografias

* LoginScreen

A classe LoginScreen é uma interface gráfica de usuário (GUI) que oferece funcionalidades de login para um aplicativo. Com uma estrutura baseada em JFrame, ela inclui campos para inserção de login e senha, botões para login, registro e saída, bem como uma opção de seleção de idioma no menu.

A interface é internacionalizada, suportando vários idiomas, incluindo português, inglês, espanhol, francês e alemão. Os elementos da GUI são ajustados dinamicamente conforme o idioma selecionado, proporcionando uma experiência amigável ao usuário.

A classe utiliza um sistema de eventos para lidar com ações do usuário, como cliques em botões ou seleção de idiomas. Além disso, ela incorpora uma imagem (logo.png) no topo da tela para personalização visual.

A tela de login permite a transição para uma tela de registro (RegisterScreen) e oferece a funcionalidade de fechar o aplicativo. Os campos de login e senha podem ser acessados por métodos públicos, permitindo a recuperação das informações inseridas pelo usuário.

Em resumo, a classe LoginScreen fornece uma interface intuitiva e localizada para a autenticação em um aplicativo, com recursos adicionais de internacionalização e personalização visual.

* MainApp

O aplicativo que deve ser executado para iniciar um cliente na máquina, que reúne e executa todas as classes relativas ao cliente.

* Todos os arquivos Properties

Arquivos que armazenam as traduções de cada língua disponibilizada nas telas: Português, Espanhol, Alemão, Francês e Inglês.

* OptionScreenAudio

Classe que implementa botões para realizar operações com arquivos de áudio, exclusivamente WAVs.

* OptionScreenImg

Classe que implementa botões para realizar operações com arquivos de imagem, exclusivamente PNGs.

* OptionScreenText

Classe que implementa botões para realizar operações com arquivos de imagem, exclusivamente PNGs.

* OptionScreenView

Classe que implementa botões para realizar operações apenas de visualização de todos os documentos.

* RegisterScreen

Classe que implementa botões e campos de texto para realizar o cadastro de novos usuários no sistema.

* SelectScreen

Classe que implementa botões para escolha de que tipo de arquivo é desejado trabalhar.

* SocketCliente

A classe SocketCliente em Java encapsula operações básicas de comunicação de um cliente por meio de sockets. Ela possui métodos para enviar e receber mensagens, além de gerenciar o fechamento adequado dos recursos do socket. A classe utiliza BufferedReader e PrintWriter para leitura e escrita de dados nos fluxos de entrada e saída do socket. Exceções de IO são tratadas para garantir uma operação robusta. O construtor imprime informações sobre a conexão e inicializa os fluxos de entrada e saída.

# DIFICULDADES

Durante o desenvolvimento, constatamos algumas dificuldades:

* No início, enfrentou-se dificuldade em consolidar uma ideia que refletisse a necessidade de utilizar todos os conceitos obtidos – dada sua abrangência. Esse problema foi confrontado e resolvido com o refinamento das ideias da sessão de *brainstorming* e com o modelo flexível do desenvolvimento do projeto.
* Desafios na etapa de debug, especialmente com o mal funcionamento no prompt. Para superar esse obstáculo, nos dedicamos a entender as causas do mal funcionamento e, a partir disso, encontrar o erro e resolvê-lo.
* Dificuldades técnicas em manipular os arquivos – devido à nossa falta de prática com operações de byte a byte – e em entender o funcionamento e as diferenças de tipos de arquivo.

# Considerações finais

Em suma, o grupo conseguiu sintetizar o conhecimento adquirido nas aulas diretamente no desenvolvimento do projeto. Fomos capazes de obter uma perspectiva prática, pesquisando sobre recursos de configuração e customização. Entende-se que o projeto cumpre seu papel em utilizar o máximo de conceitos possível ao mesmo tempo que seu formato final é uma aplicação amigável, compacta e de uso simples.

No que diz respeito às melhorias, apontamos que o projeto poderia ser refinado a fim de ser mais compacto, com mais funções e redesenhado para ser mais escalável. Essas limitações não são, no entanto, entraves definitivos ou comprometedores da qualidade do produto final - são, antes, oportunidades de um trabalho de refinamento e expansão do escopo do projeto final.

No geral, o trabalho em sua totalidade foi proveitoso para todos os membros do grupo, permitindo-nos aplicar concretamente os conceitos; o desenvolvimento da aplicação foi uma grande oportunidade para aprofundar nosso conhecimento teórico e desenvolver habilidades essenciais de análise, interpretação e correção de código e seus erros. A partir dos resultados obtidos e das interpretações desenvolvidas, temos confiança de que nosso aplicativo contribuirá significativamente para estruturar aplicações futuras, para uso pessoal ou público.

###### REFERÊNCIAS

CALVETTI, Robson. Interfaces Gráficas. 2023.

CALVETTI, Polimorfismo, Vetores e ArrayList. 2023.

CALVETTI, Conexão com BD usando JDBC. 2023.

CALVETTI, Executável .jar. 2023.

CALVETTI, Slides. 2023

GeeksforGeeks. File getAbsolutePath() method in Java with Examples - GeeksforGeeks. Disponível em: https://www.digitalocean.com/community/tutorials/java-file-path-absolute-canonical. Acesso em: 30 mai. 2023.

DevMedia. Como criar e gravar dados em TXT com Java – DevMedia. Disponível em: https://www.digitalocean.com/community/tutorials/java-file-path-absolute-canonical. Acesso em: 01 jun. 2023.

Alura. Como adicionar imagem no meu frame | Java | Alura - Cursos online de tecnologia. Disponível em: https://cursos.alura.com.br/forum/topico-como-adicionar-imagem-no-meu-frame-92628. Acesso em: 22 mai. 2023.

StackOverFlow. java - Como fechar uma JDialog após terminar de executar uma Thread?.

Disponível em: https://pt.stackoverflow.com/questions/13040/como-fechar-uma-jdialog-ap%C3%B3s-terminar-de-executar-uma-thread. Acesso em: 16 mai. 2023.

Encontrar diretório da classe em tempo de execução. Disponível em: <https://www.guj.com.br/t/encontrar-diretorio-da-classe-em-tempo-de-execucao/129250/7>. Acesso em: 1 nov. 2023.

Convertendo String para byte[]. Disponível em: <https://www.guj.com.br/t/convertendo-string-para-byte/117904/2>. Acesso em: 3 nov. 2023.

WAV BEEP. Disponível em: <http://www.simphonics.com/library/WaveFiles/Production%20Wavefiles/MISC/-WPTBEEP.WAV>. Acesso em: 5 nov. 2023.

Transferir arquivos via Socket. Disponível em: <https://www.guj.com.br/t/transferir-arquivos-via-socket/100935/6>. Acesso em: 13 nov. 2023.

Java: Class File ( a classe File ) - Obtendo informações de arquivos e diretórios. Disponível em: <https://www.programacaoprogressiva.net/2012/08/java-class-file-classe-file.html>. Acesso em: 13 nov. 2023.

‌

‌

‌

‌