FMT CHAT – PROJETO FINAL POO

Victor Hugo Gonçalves Zeschau
IFC-CAMPUS RIO DO SUL
hugozeschau@hotmail.com.br

Resumo. *Este artigo tem como assunto principal representar o projeto final da matéria de Programação Orientada a Objeto no curso de Ciência da Computação, nesse projeto será implementado um sistema de bate-papo, usando "Socket" permitindo assim comunicação entre máquinas,*

Palavras-chave: *Projeto final; Bate-papo; Socket; Comunicação.*

1. Introdução

Um sistema de bate-papo feito para usuários se comunicarem de uma maneira mais prática usando de Sockets (um fluxo de comunicação entre processos através de uma rede de computadores), porém podendo ser adaptado para quaisquer necessidades, contendo um sistema de login para controlar o acesso de usuários e uma sala geral onde os todos os usuários podem se comunicar normalmente.

Além disso a possibilidade de mandar mensagens privadas ao usuário que desejar, todos os registros de mensagens são passados para um "Json", além disso o sistema contará com uma função de criptografia de mensagens, impossibilitando o acesso a leitura da mensagem por terceiros, logo somente a pessoa receber a mensagem conseguirá visualizar a mensagem.

2. Funcionalidades

Executando a classe "Servidor Socket" primeiro o servidor será gerado e assim usuários com o mesmo sistema em suas máquinas conseguiram se conectar nele. Executando o "Cliente Socket" a aplicação começa e de cara temos um sistema de login, colocando seu nome de usuário o cliente será direcionado ao bate-papo, onde todas as mensagens de outros usuários aparecerão.

Fora isso o algoritmo conta com comandos para especificar as ações, como o comando "/sair" desliga o usuário e interrompe sua conexão com o servidor, o comando "/tell" também pode ser muito útil, permitindo o utilizador mandar uma mensagem privada a outro .

O comando "/tell" possui uma variação, sendo ela o "/tellcript" onde o utilizador manda uma mensagem privada e criptografada ao outro usuário, somente o usuário que recebeu aquela mensagem consegue ter acesso a ela, além de ser o único capaz de descriptografar a mensagem. Fora isso ainda existe o comando "/listar

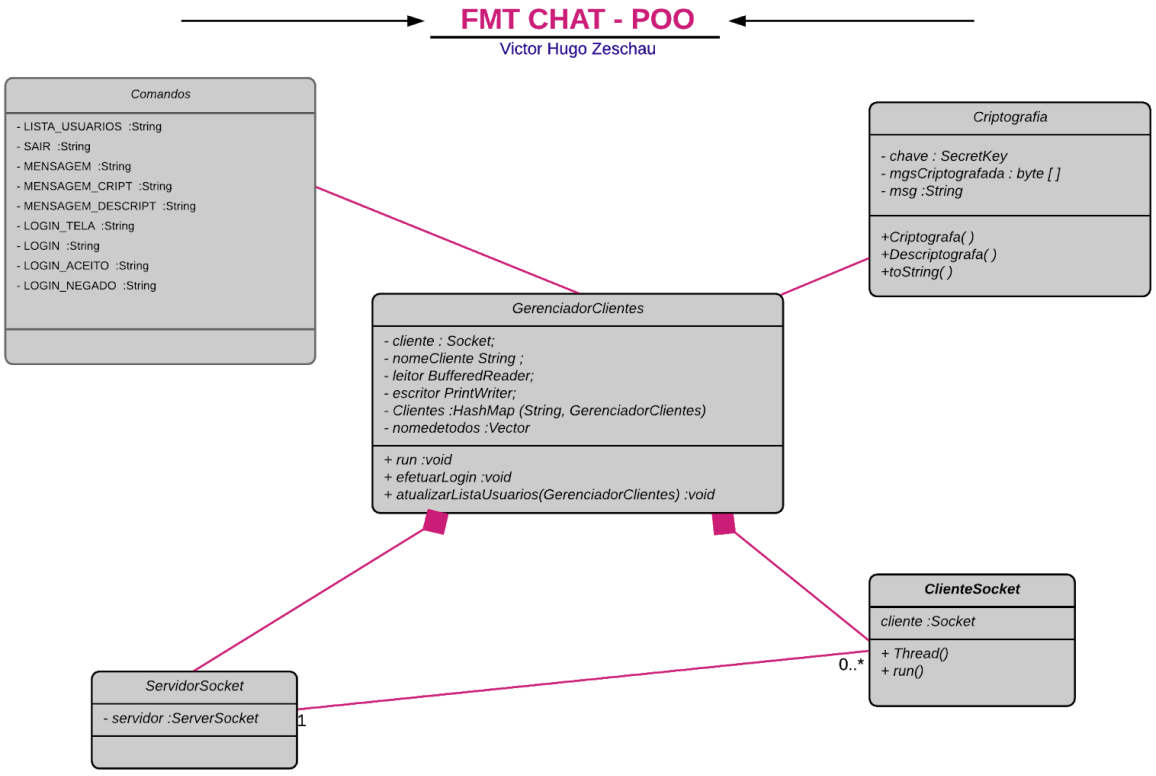
clientes", como o proprio nome ja diz, lista todos os clientes onlines na aplicação, todos os comando ficam localizados em uma classe externa, sendo eles "strings estáticas", facilitando na hora da criação do código.

2.1. Limitações

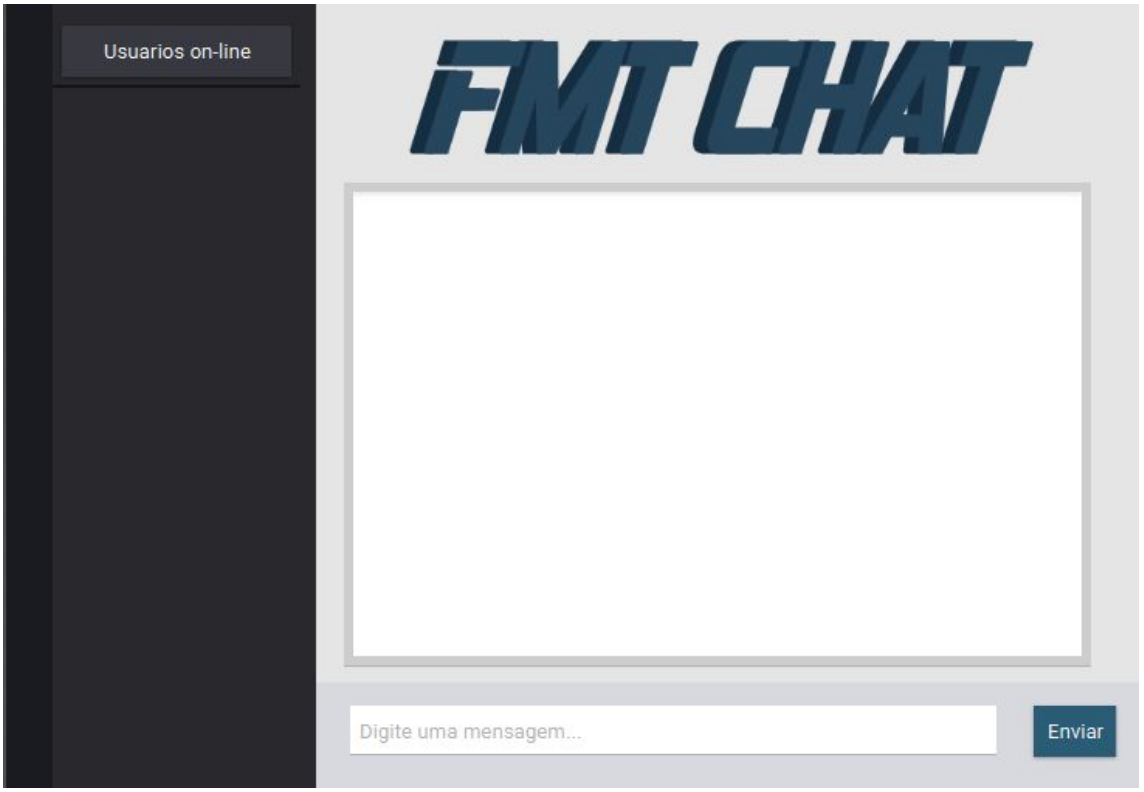
Uma implementação interessante seria a possibilidade do servidor gerar um log de mensagens, com datas e horários de todas as mensagens, o armazenamento desses logs em um banco de dados junto com o cadastro de todos os usuários.

O salvamento das mensagens dos usuários em um nuvem, para sempre que logarem no sistema as mensagens anteriores com aquele usuário aparecer ,possibilidade de fazer o upload e download de imagens do sistema, e uma forma de implementar emoticons no chat, para aumentar a interação com os usuários.

3. Diagrama de Classes (Uml)



4. Wireframe



5. Referências

Medeiros, Hiago. Utilizando Criptografia Simétrica em Java. devmedia, 2014. Disponível em: <https://www.devmedia.com.br/utilizando-criptografia-simetrica-em-java/31170> Acesso em: 23/11/2019.

MessageDigest, disponível em <http://docs.oracle.com/javase/7/docs/api/java/security/MessageDigest.html> Acesso em: 01/12/2019.

STALLINGS, William. Criptografia e segurança de redes: Princípios e práticas, 4 ed. São Paulo: Prentice Hall, 2008. Acesso em: 01/12/2019.

Lanhellas, Ronaldo. Java Socket: Entendendo a classe Socket e a ServerSocket em detalhes, 2015. Disponível em: https://www.devmedia.com.br/java-sockets-criando-comunicacoes-em-java/9465

Acesso em: 27/11/2019.

Aquiles,Alexandre. Pequeno chat feito em Java usando Sockets.disponível em 2013.

Disponível em:
<https://gist.github.com/alexandreaquiles/ad3d7287e059fb463981>Acesso em:
01/12/2019.