

IMPLEMENTAÇÃO DE SISTEMA DE CHAT EM PYTHON - CHATTTTÔ

ALUNOS:

MATHEWS EDWARDS – 201765503AB

JOAO VICTOR LOPES BORGES – 201665528AB

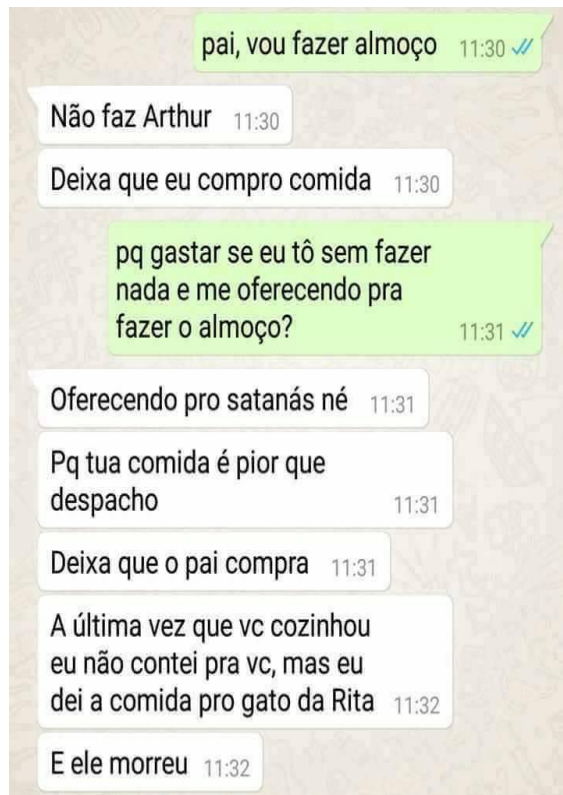
INTRODUÇÃO

Um chat é uma aplicação para conversação entre pessoas a partir de uma interface que possibilite a troca de mensagens.

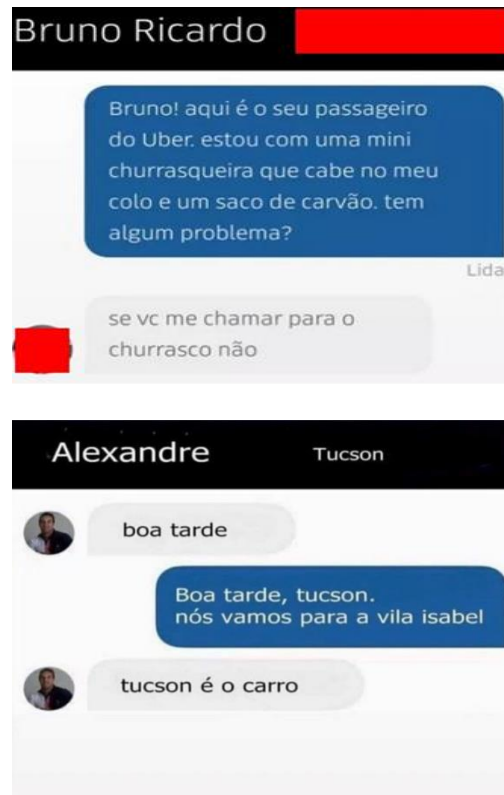
Desde a criação da internet, houveram vários sistemas de chat como por exemplo o IRC! que era baseado no protocolo Internet Relay Chat (IRC), ou o MSN fundamentado em Javascript e C#.

Existem diversos tipos de chats, mas os dois principais são o chat privado (que se realiza entre duas pessoas, onde somente as mesmas podem visualizar as mensagens) e o chat público (onde qualquer pessoa pode ver e enviar mensagens).

EXEMPLOS DE CHATS PRIVADOS



REPRODUÇÃO: WHAT'S APP

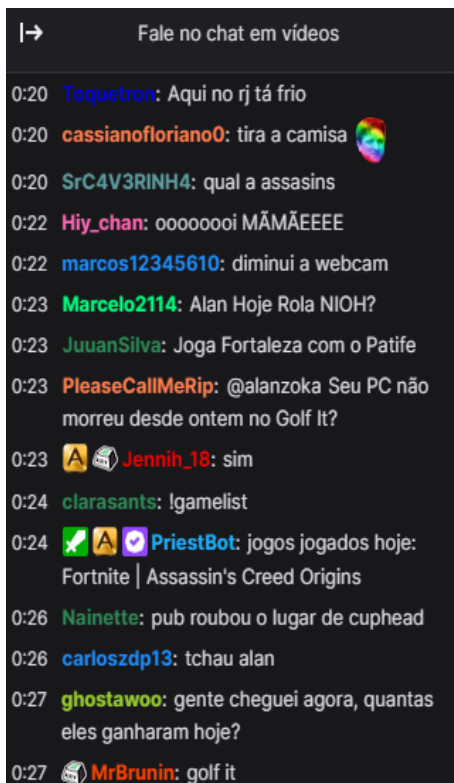


REPRODUÇÃO: UBER

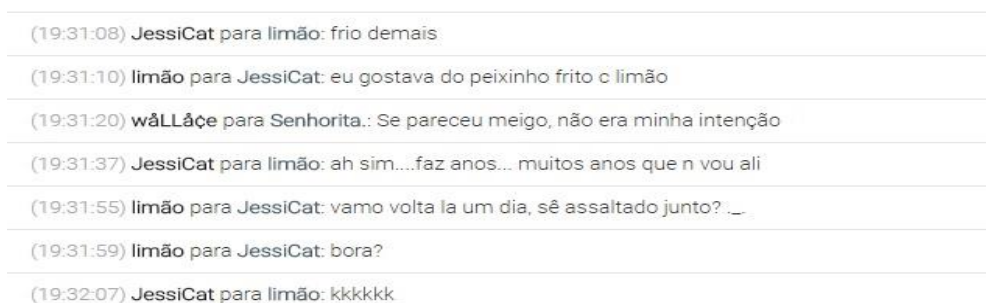


REPRODUÇÃO: FACEBOOK

EXEMPLOS DE CHATS PÚBLICOS



REPRODUÇÃO: TWITCH



REPRODUÇÃO: BATE-PAPO UOL



REPRODUÇÃO: LEAGUE OF LEGENDS



REPRODUÇÃO: CS:GO

CHATTTTÔ

Para o nosso projeto de chat em Python, importamos a biblioteca Tkinter para fazer as interfaces do chat e também a biblioteca sqlite3 para criar nosso sistema de chat público com dois bancos de dados (BD_usuariosCadastrados.py e BD_chatMessages.py).

A função do primeiro banco de dados é criar uma database (usuariosCadastrados.db) de cadastros de usuários e fazer a leitura/verificação dos dados quando alguém tentar se conectar ao sistema.

Já para o segundo banco de dados, temos a criação de uma database (chatMessages.db) onde ficarão armazenadas as informações de timestamp, nome e msg para cada mensagem que o usuário enviar no sistema.

BIBLIOTECAS IMPORTADAS

tkinter:

O pacote tkinter (“interface Tk”) é a interface Python padrão para o kit de ferramentas Tk GUI. Tanto o Tk quanto o tkinter estão disponíveis na maioria das plataformas Unix, bem como em sistemas Windows.

datetime:

O módulo datetime fornece classes para manipulação de datas e horas. Embora a aritmética de data e hora seja suportada, o foco da implementação está na extração eficiente de atributos para formatação e manipulação de saída.

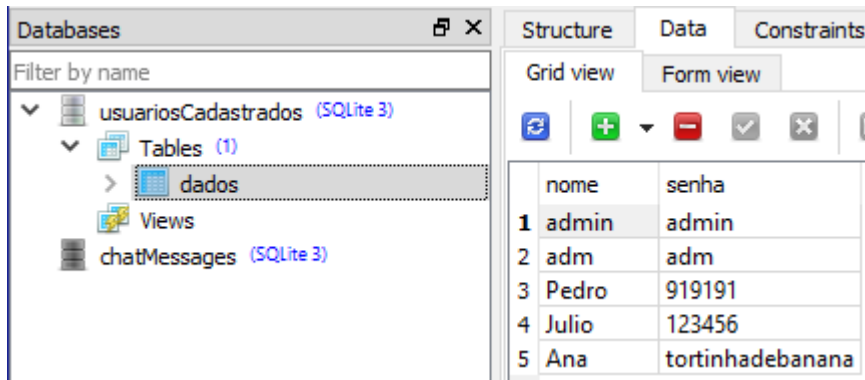
sqlite3:

SQLite é uma biblioteca C que fornece um banco de dados leve que não requer um processo de servidor separado e permite acessar o banco de dados usando uma variante não padrão da linguagem de consulta SQL. Alguns aplicativos podem usar SQLite para armazenamento interno de dados. Também é possível fazer o protótipo de um aplicativo usando SQLite e, em seguida, portar o código para um banco de dados maior, como PostgreSQL ou Oracle.

os.path:

É uma biblioteca python para gerenciamento de arquivos a partir do caminho relativo do mesmo. Tem diversas funções já implementadas como os.exists(path) ou a os.remove(path), a qual utilizamos no projeto para apagar os bancos de dados de execuções anteriores.

BANCOS DE DADOS



Databases

Filter by name

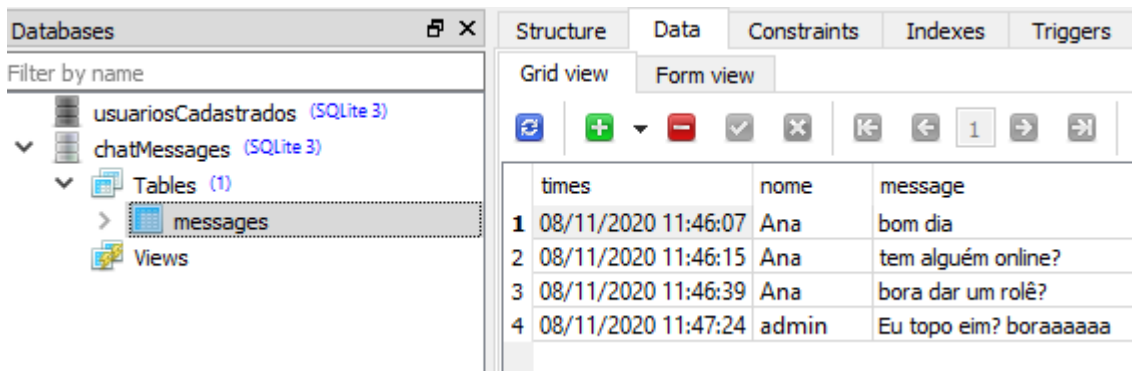
- usuariosCadastrados (SQLite 3)
 - Tables (1)
 - dados
 - Views
- chatMessages (SQLite 3)

Structure | Data | Constraints

Grid view | Form view

	nome	senha
1	admin	admin
2	adm	adm
3	Pedro	919191
4	Julio	123456
5	Ana	tortinhadebanana

REPRODUÇÃO: SQLiteStudio/usuariosCadastrados.db



Databases

Filter by name

- usuariosCadastrados (SQLite 3)
- chatMessages (SQLite 3)
 - Tables (1)
 - messages
 - Views

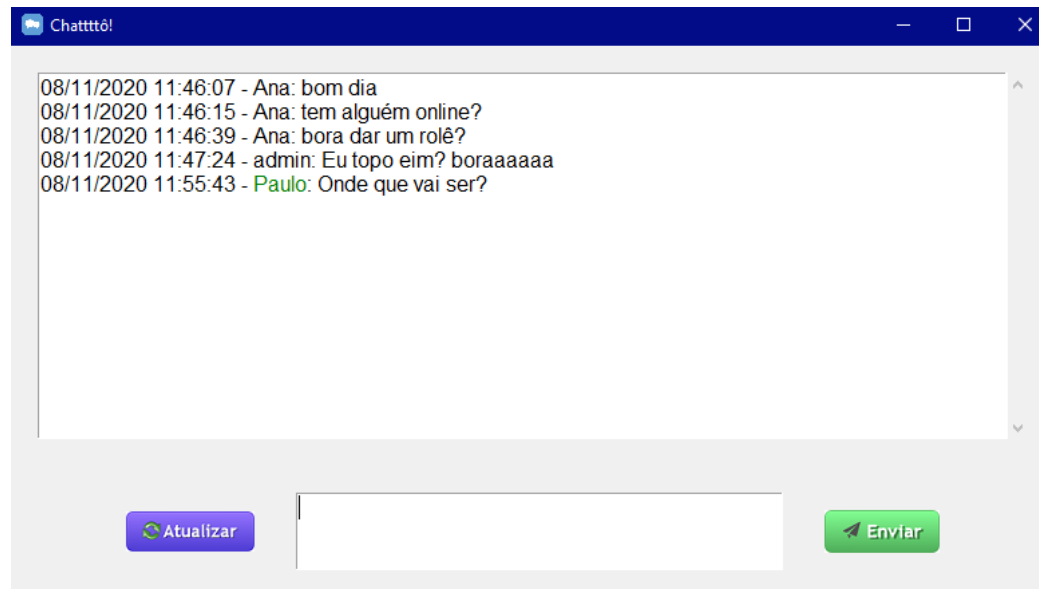
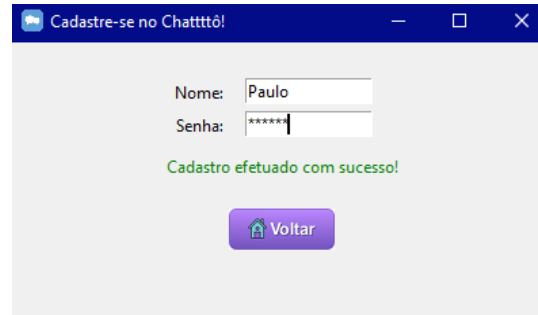
Structure | Data | Constraints | Indexes | Triggers

Grid view | Form view

	times	nome	message
1	08/11/2020 11:46:07	Ana	bom dia
2	08/11/2020 11:46:15	Ana	tem alguém online?
3	08/11/2020 11:46:39	Ana	bora dar um rolê?
4	08/11/2020 11:47:24	admin	Eu topo eim? boraaaaaa

REPRODUÇÃO: SQLiteStudio/chatMessages.db

PRINCIPAIS TELAS



EXPRESSÕES REGULARES

```
# Busca o nome do usuário no textbox, marca-o com uma tag e colore seu nome
def pesquisa_usuario(self):
    # Remove a tag 'found' do index 1 até o final (END)
    self.textbox.tag_remove('found', '1.0', END)

    # Se o usuário existir, vamos colorir seu nome
    if self.nome:
        # Index do começo da string no ScrooledText é sempre 1.0
        idx = '1.0'
        while 1:
            #Encontra a string desejada a partir do index 1
            idx = self.textbox.search(self.nome, idx, nocase=1, stopindex=END)

            # Se não encontra a String, sai do loop
            if not idx:
                break

            # Ultima soma de posição da posição atual e do tamanho do texto
            lastidx = '%s+%dc' % (idx, len(self.nome))

            # Marca a palavra encontrada com uma tag 'found'
            self.textbox.tag_add('found', idx, lastidx)
            idx = lastidx

        # Marca a string encontrada com uma cor
        self.textbox.tag_config('found', foreground=self.cor)
```

Uma **expressão regular** ou "Regex" provê uma forma concisa e flexível de identificar cadeias de caracteres como palavras ou padrões de caracteres em um texto.

No Chattertô!, definimos o nome do usuário logado no chat como uma expressão regular e a partir do método “pesquisa_usuario()”, fazemos a busca da expressão em todas as mensagens do chat de forma que toda vez que o nome do usuário for encontrado, seu nome será colorido com uma cor aleatória.

DEMONSTRAÇÃO

REFERÊNCIAS

TKINTER — Python interface to Tcl/Tk: The Python Standard Library. Disponível em: <https://docs.python.org/3/library/tkinter.html>. Acesso em: 02 nov. 2020.

RESIZABLE() method in Tkinter | Python. Disponível em: <https://www.geeksforgeeks.org/resizable-method-in-tkinter-python/>. Acesso em: 04 nov. 2020.

SILVA, Regis da. **Lendo as informações do banco de dados**. Disponível em: <http://pythonclub.com.br/gerenciando-banco-dados-sqlite3-python-parte1.html#lendo-as-informacoes-do-banco-de-dados>. Acesso em: 03 nov. 2020.

PYTHON 3 - GUI Programming (Tkinter). Disponível em: https://www.tutorialspoint.com/python3/python_gui_programming.htm. Acesso em: 05 nov. 2020.

Obrigado!