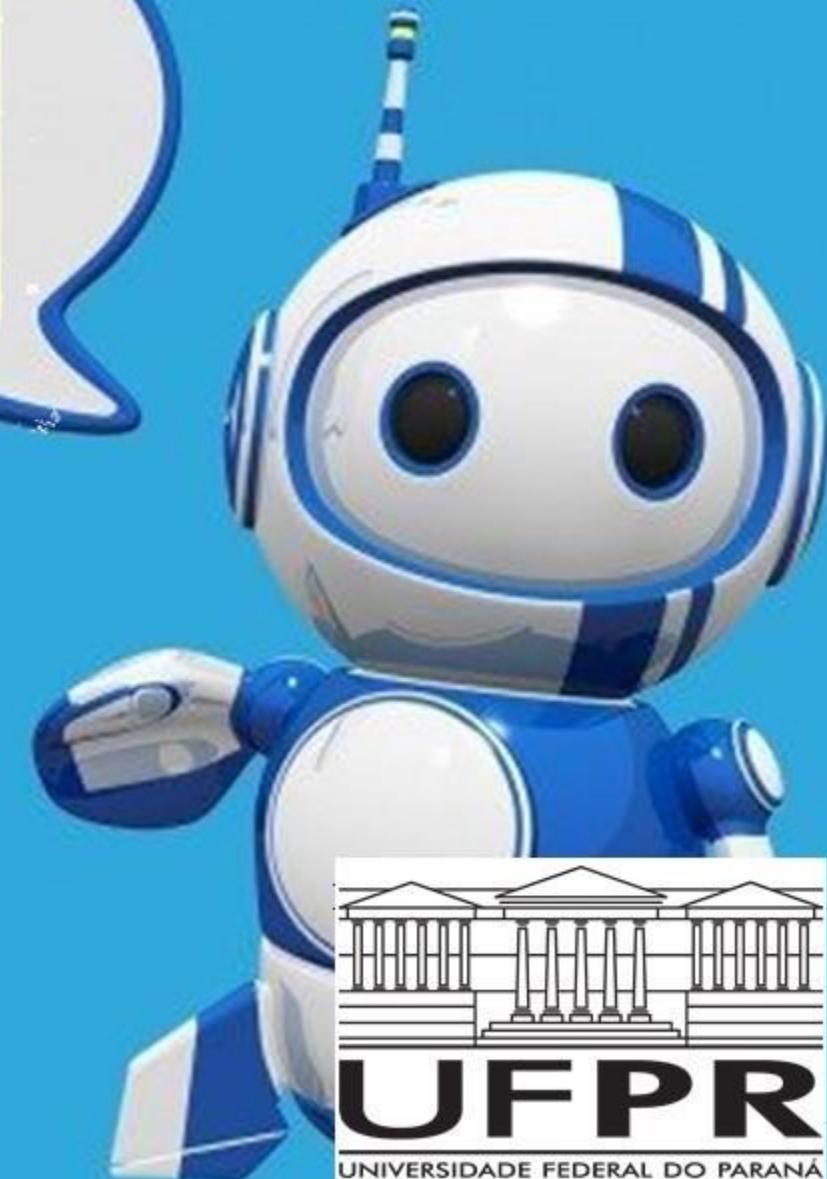


AgendaBot



Chen & Lucas

```
1 import TeleBot          #pip install pyTelegramBotAPI (Módulo do bot usado)
2 from telebot import types # função para o teclado
3 import datetime, threading, time, sched # funções usadas para os lembretes
4 import funcoes
5
6 # Token para o bot (obtida pelo BotFather)
7 token = '661545941:AAFGeT9-HdTrligeFC192HYNDofUUUz-xPfw'
8
9 # Cada usuário que usou o bot é salvo em um arquivo chamado usuarios.txt; Caso o usuário seja novo, um arquivo usuarios.txt será criado;
10 usuariosConhecidos = []    é criado um arquivo usuarios.txt se o arquivo não existir. Se o usuário for novo ele escreve o usuário na última linha.
11 try:
12     with open('usuarios.txt', 'r') as users:
13         for linha in users:
14             usuariosConhecidos.append(int(linha[:-1]))
15 except FileNotFoundError:
16     f = open('usuarios.txt', 'x')
17     f.close()
18
19 # Os envios dos lembretes são salvos em um dicionário para cada usuário, caso (mais para frente) o usuário queira cancelar
20 lembretes = {}
21 for cid in usuariosConhecidos:
22     lembretes[cid] = {}
23
24 # O passo que cada usuário está, é salvo em um arquivo de texto na pasta 'passo' com o nome 'id.txt', onde 'id' é o id do usuário.
25 passoUsuario = {}
26 for cid in usuariosConhecidos:
27     try:
28         with open('passo/{}.txt'.format(cid), 'r') as f:
29             passoUsuario[cid] = int(f.readline())
30     except FileNotFoundError:
31         f = open('passo/{}.txt'.format(cid), 'w')
32         f.write("0")
33         f.close()
34     passoUsuario[cid] = 0
35
36 # Lê as informações (são as informações fornecidas pelo usuário ao criar o evento) temporárias salvas na pasta 'linhatemp' com o nome 'id.txt' e salva no dicionário 'linhatemp'.
37 linhatemp = {}
38 temp = ["titulo", "data", "detalhes", "casolembrete"]
39 for user in usuariosConhecidos:
40     try:
41         f = open('linhatemp/{}.txt'.format(user), 'r')
42         linhas = f.readlines()
43         for i in range(len(linhas)):
44             if '\n' in linhas[i]:
45                 linhas[i] = linhas[i][:-1]
46             if i == 1:
47                 linhatemp[str(user)+temp[i]] = funcoes.texto_para_datetime(linhas[i])
48             if i == 3:
49                 linhatemp[str(user)+temp[i]] = int(linhas[i])
50             else:
51                 linhatemp[str(user)+temp[i]] = linhas[i]
52     except FileNotFoundError:
53         f = open('linhatemp/{}.txt'.format(user), 'x')
54         f.close()
```

Nome	Data de modificaç...	Tipo	Tamanho
582190171.txt	25/11/2018 06:13	Documento de Te...	1 KB
667218364.txt	25/11/2018 16:29	Documento de Te...	1 KB

Nome	Data de modificaç...	Tipo	Tamanho
__pycache__	25/11/2018 06:14	Pasta de arquivos	
editartemp	25/11/2018 07:50	Pasta de arquivos	
editartempcaso	25/11/2018 07:50	Pasta de arquivos	
editartempdata	25/11/2018 07:53	Pasta de arquivos	
eventos	25/11/2018 07:42	Pasta de arquivos	
linhatemp	25/11/2018 07:38	Pasta de arquivos	
passo	25/11/2018 07:35	Pasta de arquivos	
funcoes.py	25/11/2018 06:13	Python File	2 KB
trabalho_de_progamacao.py	25/11/2018 08:43	Python File	37 KB
usuarios.txt	25/11/2018 07:35	Documento de Te...	1 KB

```
1 import datetime
2 def validar_data(dia, mes, ano):
3     """
4     Valida a data
5     """
6     try:
7         datetime.date(ano, mes, dia)
8         return True
9     except ValueError:
10        return False
11
12 def dois_digitos(n):
13     """
14     Ao colocar '05:02' estava salvando '5:2' que são bem diferentes, por isso foi criada essa função para deixar '05:02' == '05:02'
15     """
16     if n < 10:
17         return "0"+str(n)
18     else:
19         return str(n)
20
21 def texto_para_datetime(texto):
22     """
23     Converte o texto no formato "DD/MM/AAAA HH:MM" para um datetime.
24     """
25     dataT = texto.split()
26     data = [int(n) for n in dataT[0].split("/")]
27     hora = [int(n) for n in dataT[1].split(":")]
28     return datetime.datetime(data[-1], data[-2], data[-3], hora[0], hora[1])
29
30 def datetime_para_texto(dt):
31     """
32     Converte um datetime para um texto no formato "DD/MM/AAAA HH:MM"
33     """
34     return "{}/{}/{}/{} {}:{}{}".format(dois_digitos(dt.day), dois_digitos(dt.month), dt.year, dois_digitos(dt.hour), dois_digitos(dt.minute)))
35
36 def salvar_edicao(eventos, cid):
37     """
38     Salva uma edição feita no dicionário para o arquivo.
39     """
40     linhas_salvar = []
41     for linha in eventos[cid]:
42         linhas_salvar.append(linha[0]+"\n")
43         linhas_salvar.append(datetime_para_texto(linha[1])+"\n")
44         linhas_salvar.append(linha[2]+"\n")
45         if linha[3] == 0:
46             linhas_salvar.append("0\n")
47         else:
48             linhas_salvar.append(datetime_para_texto(linha[3])+"\n")
49     with open('eventos/{}.txt'.format(cid), 'w') as f:
50         f.writelines(linhas_salvar)
```

15/11/2018

23:03 ✓

Qual é o horário? (digite no formato
HH:MM)

23:03

23:05

23:03 ✓

Digite mais detalhes sobre o evento:

23:03

Ah não

23:03 ✓

Deseja adicionar lembrete?

23:03

SIM

23:03 ✓

Quantas horas antes do evento você
deseja ser lembrado?

(O número máximo que você pode
digitar é: 0.0)

23:03

0

23:03 ✓

Lembrete definido para 15/11/2018
às 23:5

23:03

Evento "Prova" - 15/11/2018 salvo!

```

55 # Lê os eventos salvos em arquivos de texto para cada usuário e colocá-os em uma matriz no dicionário 'evento' na chave 'id'. As datas salvas em formato de texto são convertidas para datetime e a matriz é organizada
56
57 eventos = {}
58 for user in usuariosConhecidos:
59     try:
60         f = open('eventos/{}.txt'.format(user), 'r')
61         eventos[user] = []
62         lista = f.readlines()
63         for i in range(len(lista)):
64             lista[i] = lista[i][:-1]
65             for i in range(0, len(lista), 4):
66                 eventos[user].append(lista[i:i+4])
67         f.close()
68     except FileNotFoundError:
69         f = open('eventos/{}.txt'.format(user), 'x')
70         eventos[user] = []
71         f.close()
72
73 for user in usuariosConhecidos:
74     for i in range(len(eventos[user])):
75         if eventos[user][i][-1] != "0":
76             eventos[user][i][-1] = funcoes.texto_para_datetime(eventos[user][i][-1])
77         else:
78             eventos[user][i][-1] = 0
79             eventos[user][i][1] = funcoes.texto_para_datetime(eventos[user][i][1])
80
81 for cid in usuariosConhecidos:
82     for k in range(len(eventos[cid])):
83         for i in range(1, len(eventos[cid]) - k):
84             if eventos[cid][i][1] < eventos[cid][i-1][1]:
85                 eventos[cid][i], eventos[cid][i-1] = eventos[cid][i-1], eventos[cid][i]
86
87 # Criação de teclados
88 comandos = ['Adicionar evento', 'Lista de eventos']
89 selecionarComando = types.ReplyKeyboardMarkup()
90 selecionarComando.add(*comandos)
91
92 sim_ou_não = types.ReplyKeyboardMarkup()
93 sim_ou_não.add('SIM', 'NÃO')
94
95
96 cancelar = types.ReplyKeyboardMarkup()
97 cancelar.add('CANCELAR')
98
99 # Esconde o teclado
100 esconderTeclado = types.ReplyKeyboardRemove()
101
102 # Essa função serve só para evitar erros que aconteceriam se tentasse ler o passo de um usuário que não está no dicionário.
103 def passo_do_usuario(uid):
104     # Recebe o id do usuário, retorna o passo em que ele está. Se o usuário não estiver na lista de usuário conhecidos e se o passo é colocado em 0, são criados arquivos, dicionários e lista necessários.
105     if uid in passoUsuario:
106         return passoUsuario[uid]

```

667218364.txt - Bloco de notas

Arquivo Editar Formatar Exibir Ajuda

b

```

107
108     else:
109         usuariosConhecidos.append(uid)
110         lembretes[uid] = {}
111         eventos[uid] = []
112         f = open('usuarios.txt', 'a')
113         f.write(str(uid)+"\n")
114         f.close()
115
116     try:
117         f = open('linhatemp/{}.txt'.format(uid), 'x')
118         f.close()
119     except FileExistsError:
120         pass
121     try:
122         f = open('eventos/{}.txt'.format(uid), 'x')
123         f.close()
124     except FileExistsError:
125         pass
126
127     passoUsuario[uid] = 0
128     v = open('passo/{}.txt'.format(uid), 'w')
129     v.write("0")
130     v.close()
131     return 0
132
133 # Essa função é para criar um teclado, ela é usada na lista e mostra ao usuário um teclado com números para cada evento e a opção de cancelar
134 def teclado_editar(quantidade_itens):
135     tecladoEditar = types.ReplyKeyboardMarkup()
136     tecladoEditar.add(*[str(n)+""] for n in range(1, quantidade_itens + 1)]+["CANCELAR"])
137     return tecladoEditar
138
139 # Salva no dicionário de eventos e no arquivo do usuário cid, o seu evento e organiza os eventos do dicionário por ordem de data.
140 def salvar_evento(eventos, cid, titulo, data, detalhes, lembrete, selecionarComando):
141     # O bot vai tentar colocar as informações dentro da matriz de eventos (#1), mas se no dicionário nada estiver salvo no cid, vai dar KeyError, nesse caso (#2) é criada uma matriz com o evento dentro.
142     try:#1
143         eventos[cid].append([titulo, data, detalhes, lembrete])
144     except KeyError:#2
145         eventos[cid] = [[titulo, data, detalhes, lembrete]]
146
147     if lembrete == 0:
148         lembreteTexto = "0"
149     else:
150         lembreteTexto = funcoes.datetime_para_texto(lembrete)
151
152     f = open('eventos/{}.txt'.format(cid), 'a')
153     f.write(str(titulo)+"\n"+funcoes.datetime_para_texto(data)+"\n"+str(detalhes)+"\n"+lembreteTexto+"\n")
154     f.close()
155
156     for k in range(len(eventos[cid])):
157         for i in range(1, len(eventos[cid]) - k):
158             if eventos[cid][i][1] < eventos[cid][i-1][1]:
159                 eventos[cid][i], eventos[cid][i-1] = eventos[cid][i-1], eventos[cid][i]

```



```

107
108     else:
109         usuariosConhecidos.append(uid)
110         lembretes[uid] = {}
111         eventos[uid] = []
112         f = open('usuarios.txt', 'a')
113         f.write(str(uid)+"\n")
114         f.close()
115
116     try:
117         f = open('linhatemp/{}.txt'.format(uid), 'x')
118         f.close()
119     except FileExistsError:
120         pass
121     try:
122         f = open('eventos/{}.txt'.format(uid), 'x')
123         f.close()
124     except FileExistsError:
125         pass
126
127     passoUsuario[uid] = 0
128     v = open('passo/{}.txt'.format(uid), 'w')
129     v.write("0")
130     v.close()
131     return 0
132
133 # Essa função é para criar um teclado, ela é usada na lista e mostra ao usuário um teclado com números para cada evento e a opção de cancelar
134 def teclado_editar(quantidade_itens):
135     tecladoEditar = types.ReplyKeyboardMarkup()
136     tecladoEditar.add(*[str(n)+""] for n in range(1, quantidade_itens + 1)]+["CANCELAR"])
137     return tecladoEditar
138
139 # Salva no dicionário de eventos e no arquivo do usuário cid, o seu evento e organiza os eventos do dicionário por ordem de data.
140 def salvar_evento(eventos, cid, titulo, data, detalhes, lembrete, selecionarComando):
141     # O bot vai tentar colocar as informações dentro da matriz de eventos (#1), mas se no dicionário nada estiver salvo no cid, vai dar KeyError, nesse caso (#2) é criada uma matriz com o evento dentro.
142     try:#1
143         eventos[cid].append([titulo, data, detalhes, lembrete])
144     except KeyError:#2
145         eventos[cid] = [[titulo, data, detalhes, lembrete]]
146
147     if lembrete == 0:
148         lembreteTexto = "0"
149     else:
150         lembreteTexto = funcoes.datetime_para_texto(lembrete)
151
152     f = open('eventos/{}.txt'.format(cid), 'a')
153     f.write(str(titulo)+"\n"+funcoes.datetime_para_texto(data)+"\n"+str(detalhes)+"\n"+lembreteTexto+"\n")
154     f.close()
155
156     for k in range(len(eventos[cid])):
157         for i in range(1, len(eventos[cid]) - k):
158             if eventos[cid][i][1] < eventos[cid][i-1][1]:
159                 eventos[cid][i], eventos[cid][i-1] = eventos[cid][i-1], eventos[cid][i]

```

Arquivo Editar Formatar Exibir Ajuda

~time to sleep~

15/02/10 12:02

~guys, it's my time, i'm gonna~

0

anything

25/11/2018 16:09

hgryg

0

Something

25/11/2018 16:13

~Think have something here~

0

Prova da Ximena

25/11/2018 19:00

O que vai cair???

0

Sleep time

29/11/2018 11:50

~Sorry, it's my time to sleep, bye 0/~/

26/11/2018 11:50

Trabalho e apresentação de programação

28/11/2019 15:35

~Não sei o que mudar nos detalhes~

0

65453

17/01/2029 21:50

dasda

12/01/2029 21:50

```

159     eventos[cid][i][-1] < eventos[cid][i-1][-1]:
160         eventos[cid][i], eventos[cid][i-1] = eventos[cid][i-1], eventos[cid][i]
161
162     bot.send_message(cid, "Evento \"{}\" - {} salvo!\nO que deseja fazer?".format(titulo, funcoes.datetime_para_texto(data)), reply_markup=selecionarComando)
163
164 # Coloca em fila crescente todos os lembretes salvos e programa o envio.
165 fila = sched.scheduler(time.time, time.sleep)
166 def enviar_lembrete(usuario, titulo, data):
167     bot.send_message(usuario, "Você terá \"{}\" em {}".format(titulo, data))
168
169 for cid in usuariosConhecidos:
170     agora = datetime.datetime.now()
171     for i in range(len(eventos[cid])):
172         if eventos[cid][i][-1] != 0:
173             if eventos[cid][i][-1] < agora:
174                 eventos[cid][i][-1] = 0
175             else:
176                 st = (eventos[cid][i][-1] - agora).total_seconds()
177                 lembretes[cid][eventos[cid][i][0]:funcoes.datetime_para_texto(eventos[cid][i][1])] = fila.enter(st, 1, enviar_lembrete, argument=(cid, eventos[cid][i][0], funcoes.datetime_para_texto(eventos[cid][i][1])))
178 t = threading.Thread(target=fila.run)
179 t.start()
180
181 # Essa função é usada para mostrar na tela cada mensagem recebida num print do tipo: Nome [ID]: "mensagem"
182 def listener(mensagens):
183     # Usado somente para mostrar as mensagens recebidas no console.
184     for mensagem in mensagens:
185         if mensagem.content_type == 'text':
186             print(str(mensagem.chat.first_name) + " [" + str(mensagem.chat.id) + "]: " + mensagem.text)
187
188 bot = telebot.TeleBot(token)
189 bot.set_update_listener(listener) # a função anterior é usada aqui
190
191 # O que o bot faz quando recebe o comando '/start'
192 @bot.message_handler(commands=['start'])
193 def comando_start(mensagem):
194     cid = mensagem.chat.id # ID do usuário que mandou o comando
195     # Se o usuário não estiver na lista de usuários conhecidos será mandada a mensagem "Bem vindo [...]" para o usuário cid e vai mostrar o teclado de selecionarComando, isso é o reply_markup=selecionarComando que faz.
196     if cid not in usuariosConhecidos:
197         usuariosConhecidos.append(cid)
198         lembretes[cid] = {}
199         eventos[cid] = []
200         with open('usuarios.txt', 'a') as f:
201             f.write(str(cid)+"\n")
202         passoUsuario[cid] = 0
203         with open('passo/{}.txt'.format(cid), 'w') as v:
204             v.write("0")
205         bot.send_message(cid, "Bem vindo ao AgendaBot, gostaria de adicionar algum evento?", reply_markup=selecionarComando)
206
207     else:
208         passoUsuario[cid] = 0
209         with open('passo/{}.txt'.format(cid), 'w') as v:
210             v.write("0")
211         bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
212
213 # O que o bot responde quando o usuário seleciona a opção 'Adicionar evento'

```

```
:\\Users\\CHEN>Desktop\\python\\chatbot\\teste.py
hen [667218364]: /start
hen [667218364]: Adicionar evento
hen [667218364]: Prova
hen [667218364]: 98/11/10
hen [667218364]: 17/11/2018
hen [667218364]: 22:80
hen [667218364]: 22:50
hen [667218364]: Prova de programação
hen [667218364]: SIM
hen [667218364]: 0
hen [667218364]: Lista de eventos
hen [667218364]: 1)
hen [667218364]: Data
hen [667218364]: 10/11/2018
hen [667218364]: 21:20
hen [667218364]: Lista de eventos
hen [667218364]: 1)
hen [667218364]: Lembrete
hen [667218364]: Adicionar evento
hen [667218364]: Trabalho
hen [667218364]: 28/11/2018
hen [667218364]: 15:30
hen [667218364]: Trabalho de programação + apresentação #desesperada
hen [667218364]: NÃO
hen [667218364]: Adicionar evento
hen [667218364]: CANCELAR
hen [667218364]: Adicionar evento
hen [667218364]: Prova
hen [667218364]: CANCELAR
hen [667218364]: Adicionar evento
hen [667218364]: Prova
hen [667218364]: 17/11/2018
hen [667218364]: 18:50
hen [667218364]: CANCELAR
hen [667218364]: Lista de eventos
hen [667218364]: CANCELAR
hen [667218364]: Adicionar evento
hen [667218364]: Prova
hen [667218364]: 28/11/2018
hen [667218364]: 13:30
hen [667218364]: Prova de Geometria Analítica
hen [667218364]: CANCELAR
hen [667218364]: Lista de eventos
hen [667218364]: CANCELAR
hen [667218364]: Adicionar evento
hen [667218364]: Prova de GA e transformações
hen [667218364]: 11/11/2011
hen [667218364]: 17/02
hen [667218364]: 17:01
hen [667218364]: Prova da Ximena
hen [667218364]: Adicionar evento
hen [667218364]: Trabalho
hen [667218364]: 24/11/2018
hen [667218364]: 7:15
hen [667218364]: Gincana no IFPR
hen [667218364]: SIM
hen [667218364]: 5
```

```

159     eventos[cid][i][-1] < eventos[cid][i-1][-1]:
160         eventos[cid][i], eventos[cid][i-1] = eventos[cid][i-1], eventos[cid][i]
161
162     bot.send_message(cid, "Evento \"{}\" - {} salvo!\nO que deseja fazer?".format(titulo, funcoes.datetime_para_texto(data)), reply_markup=selecionarComando)
163
164 # Coloca em fila crescente todos os lembretes salvos e programa o envio.
165 fila = sched.scheduler(time.time, time.sleep)
166 def enviar_lembrete(usuario, titulo, data):
167     bot.send_message(usuario, "Você terá \"{}\" em {}".format(titulo, data))
168
169 for cid in usuariosConhecidos:
170     agora = datetime.datetime.now()
171     for i in range(len(eventos[cid])):
172         if eventos[cid][i][-1] != 0:
173             if eventos[cid][i][-1] < agora:
174                 eventos[cid][i][-1] = 0
175             else:
176                 st = (eventos[cid][i][-1] - agora).total_seconds()
177                 lembretes[cid][eventos[cid][i][0]:funcoes.datetime_para_texto(eventos[cid][i][1])] = fila.enter(st, 1, enviar_lembrete, argument=(cid, eventos[cid][i][0], funcoes.datetime_para_texto(eventos[cid][i][1])))
178 t = threading.Thread(target=fila.run)
179 t.start()
180
181 # Essa função é usada para mostrar na tela cada mensagem recebida num print do tipo: Nome [ID]: "mensagem"
182 def listener(mensagens):
183     # Usado somente para mostrar as mensagens recebidas no console.
184     for mensagem in mensagens:
185         if mensagem.content_type == 'text':
186             print(str(mensagem.chat.first_name) + " [" + str(mensagem.chat.id) + "]: " + mensagem.text)
187
188 bot = telebot.TeleBot(token)
189 bot.set_update_listener(listener) # a função anterior é usada aqui
190
191 # O que o bot faz quando recebe o comando '/start'
192 @bot.message_handler(commands=['start'])
193 def comando_start(mensagem):
194     cid = mensagem.chat.id # ID do usuário que mandou o comando
195     # Se o usuário não estiver na lista de usuários conhecidos será mandada a mensagem "Bem vindo [...]" para o usuário cid e vai mostrar o teclado de selecionarComando, isso é o reply_markup=selecionarComando que faz.
196     if cid not in usuariosConhecidos:
197         usuariosConhecidos.append(cid)
198         lembretes[cid] = {}
199         eventos[cid] = []
200         with open('usuarios.txt', 'a') as f:
201             f.write(str(cid)+"\n")
202         passoUsuario[cid] = 0
203         with open('passo/{}.txt'.format(cid), 'w') as v:
204             v.write("0")
205         bot.send_message(cid, "Bem vindo ao AgendaBot, gostaria de adicionar algum evento?", reply_markup=selecionarComando)
206
207     else:
208         passoUsuario[cid] = 0
209         with open('passo/{}.txt'.format(cid), 'w') as v:
210             v.write("0")
211         bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
212
213 # O que o bot responde quando o usuário seleciona a opção 'Adicionar evento'

```

domingo, 25 de novembro de 2018



Chen

/start

07:35:21



Eru

Bem vindo ao AgendaBot, gostaria de adicionar algum evento?

07:35:23



Escreva uma mensagem...



ENVIAR

Adicionar evento

Lista de eventos

```
211 # O que o bot responde quando o usuário seleciona a opção 'Adicionar evento'
212 @bot.message_handler(func=lambda message: message.text == comandos[0] and passo_do_usuario(message.chat.id) == 0)
213 def adicionar_evento(mensagem):
214     cid = mensagem.chat.id
215     bot.send_message(cid, "Digite um título, por favor", reply_markup=cancelar) # Manda uma mensagem pedindo o título e mostra o teclado com a opção cancelar
216     passoUsuario[cid] = 1
217     with open('passo/{}.txt'.format(cid), 'w') as f:
218         f.write("1")
219
220 # Salva o título e pergunta a data
221 @bot.message_handler(func=lambda message: passo_do_usuario(message.chat.id) == 1) # O que o bot vai fazer quando receber mensagem de um usuário que
222 # está no passo 1
223 def adicionar_evento1(mensagem):
224     cid = mensagem.chat.id
225     if mensagem.text == "CANCELAR": # Se a mensagem for "CANCELAR"
226         passoUsuario[cid] = 0
227         with open('passo/{}.txt'.format(cid), 'w') as v:
228             v.write("0")
229         bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
230         # Basicamente vai voltar para ao começo, colocando passo = 0 e mostrando a mensagem "O que deseja fazer?" e o teclado
231     else:
232         linhatemp[str(cid)+"titulo"] = mensagem.text # Salva o título na linhatemp
233         with open('linhatemp/{}.txt'.format(cid), 'a') as f:
234             f.write(mensagem.text+'\n')
235         bot.send_message(cid, "Qual é a data? (digite no formato DD/MM/AAAA)", reply_markup=cancelar)
236         # pergunta a data mostrando a opção de cancelar
237         passoUsuario[cid] = 2
238         with open('passo/{}.txt'.format(cid), 'w') as v:
239             v.write("2")
240
241 # Salva a data, pergunta o horário
242 @bot.message_handler(func=lambda message: passo_do_usuario(message.chat.id) == 2) # O que o bot vai fazer quando receber mensagem de um usuário que
243 # está no passo 2
244 def adicionar_evento2(mensagem):
245     cid = mensagem.chat.id
246     if mensagem.text == "CANCELAR":
247         passoUsuario[cid] = 0
248         with open('passo/{}.txt'.format(cid), 'w') as v:
249             v.write("0")
250         del linhatemp[str(cid)+"titulo"]
251         f = open('linhatemp/{}.txt'.format(cid), 'w')
252         f.writelines([""])
253         f.close()
254         bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
255     else:
256         try:
257             data = mensagem.text
258             data = [int(n) for n in data.split("/")]
259             if funcoes.validar_data(data[0], data[1], data[2]) == False or len(data) != 3:
260                 raise ValueError("Data inválida")
261             linhatemp[str(cid)+"data"] = datetime.datetime(data[-1], data[-2], data[-3])
262             with open('linhatemp/{}.txt'.format(cid), 'a') as f:
263                 f.write(funcoes.datetime_para_texto(linhatemp[str(cid)+"data"]))
264             bot.send_message(cid, "Qual é o horário? (digite no formato HH:MM)", reply_markup=cancelar) # Pergunta o horário
265             passoUsuario[cid] = 3
266             with open('passo/{}.txt'.format(cid), 'w') as v:
```

domingo, 25 de novembro de 2018



Chen

/start

07:35:21



Eru

Bem vindo ao AgendaBot, gostaria de adicionar algum evento?

07:35:23



Chen

Lista de eventos

07:36:47



Eru

Não há eventos salvos

07:36:48

O que deseja fazer?

07:36:48



Escreva uma mensagem...



ER



ENVIAR

Adicionar evento

Lista de eventos

```
264     passoUsuario[cid] = 3
265     with open('passo/{}.txt'.format(cid), 'w') as v:
266         v.write("3")
267     except (ValueError, IndexError):
268         bot.send_message(cid, "Data inválida! Digite a data no formato DD/MM/AAAA !\nExemplo: dia 17 de janeiro de 2029 corresponde a data 17/01/2029.", reply_markup=cancelar)
269
270 # Salva o horário e pede os detalhes
271 @bot.message_handler(func=lambda message: passo_do_usuario(message.chat.id) == 3)
272 def adicionar_evento3(mensagem):
273     cid = mensagem.chat.id
274     if mensagem.text == "CANCELAR":
275         passoUsuario[cid] = 0
276         with open('passo/{}.txt'.format(cid), 'w') as v:
277             v.write("0")
278         del linhatemp[str(cid)+"titulo"]
279         del linhatemp[str(cid)+"data"]
280         with open('linhatemp/{}.txt'.format(cid), 'w') as f:
281             f.writelines([''])
282         bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
283     else:
284         try:
285             horario = mensagem.text
286             horario = [int(n) for n in horario.split(":")]
287             if horario[0] < 0 or horario[0] > 23:
288                 raise ValueError("Hora inválida")
289             if horario[1] < 0 or horario[1] > 59:
290                 raise ValueError("Minuto inválido")
291             if len(horario) != 2:
292                 raise ValueError("Horário inválido")
293             linhatemp[str(cid)+"data"] = linhatemp[str(cid)+"data"].replace(hour=horario[0], minute=horario[1])
294             with open('linhatemp/{}.txt'.format(cid), 'w') as f:
295                 f.write(linhatemp[str(cid)+"titulo"]+"\n"+funcoes.datetime_para_texto(linhatemp[str(cid)+"data"])+"\n")
296             bot.send_message(cid, "Digite mais detalhes sobre o evento:", reply_markup=cancelar) # Pergunta os detalhes
297             passoUsuario[cid] = 4
298             with open('passo/{}.txt'.format(cid), 'w') as v:
299                 v.write("4")
300         except (ValueError, IndexError):
301             bot.send_message(cid, "Horário inválido! Digite o horário no formato HH:MM !\nExemplo: duas e quarenta e cinco da tarde são 14:45")
302
303 # Salva os detalhes e pergunta se quer adicionar um lembrete
304 @bot.message_handler(func=lambda message: passo_do_usuario(message.chat.id) == 4)
305 def adicionar_evento4(mensagem):
306     cid = mensagem.chat.id
307     if mensagem.text == "CANCELAR":
308         passoUsuario[cid] = 0
309         with open('passo/{}.txt'.format(cid), 'w') as v:
310             v.write("0")
311         del linhatemp[str(cid)+"titulo"]
312         del linhatemp[str(cid)+"data"]
313         with open('linhatemp/{}.txt'.format(cid), 'w') as f:
314             f.writelines([''])
315         bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
316     else:
317         linhatemp[str(cid)+"detalhes"] = mensagem.text
```



Mensagem

En 123

1 2 3 4 5 6 7 8 9 0

! @ # \$ % ^ & * ()

Q W E R T Y U I O P

[] +

A S D F G H J K L

Z X C V B N M

↑ ? /

Pt 1@# ,

En ↗ .

...

←

```

315     bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
316 else:
317     linhatemp[str(cid)+"detalhes"] = mensagem.text
318     with open('linhatemp/{}.txt'.format(cid), 'a') as f:
319         f.write(mensagem.text+"\n")
320     agora = datetime.datetime.now()
321     if agora < linhatemp[str(cid)+"data"]:
322         bot.send_message(cid, "Deseja adicionar lembrete?", reply_markup=sim_ou_não)
323         passoUsuario[cid] = 5
324         with open('passo/{}.txt'.format(cid), 'w') as v:
325             v.write("5")
326     else: #3
327         salvar_evento(eventos, cid, linhatemp[str(cid)+"titulo"], linhatemp[str(cid)+"data"], linhatemp[str(cid)+"detalhes"], 0, selecionarComando) #4
328         passoUsuario[cid] = 0
329         with open('passo/{}.txt'.format(cid), 'w') as v:
330             v.write("0")
331         del linhatemp[str(cid)+"titulo"]
332         del linhatemp[str(cid)+"data"]
333         del linhatemp[str(cid)+"detalhes"]
334         with open('linhatemp/{}.txt'.format(cid), 'w') as f:
335             f.writelines([''])
336
337 # o que o bot faz dependendo da resposta
338 @bot.message_handler(func=lambda message: passo_do_usuario(message.chat.id) == 5)
339 def lembrete(mensagem):
340     cid = mensagem.chat.id
341     if mensagem.text == "SIM":
342         agora = datetime.datetime.now()
343         if (linhatemp[str(cid)+"data"] - agora).days >= 1:
344             bot.send_message(cid, "Quantos dias antes do evento você deseja ser lembrado?\n(0 número máximo que você pode digitar é: {})".format((linhatemp[str(cid)+"data"]-agora).days), reply_markup=esconderTeclado)
345             linhatemp[str(cid)+"casolembrete"] = 1
346             with open('linhatemp/{}.txt'.format(cid), 'a') as f:
347                 f.write("1")
348             passoUsuario[cid] = 6
349         elif linhatemp[str(cid)+"data"] < agora:
350             bot.send_message(cid, "Não é mais possível criar um lembrete.")
351             salvar_evento(eventos, cid, linhatemp[str(cid)+"titulo"], linhatemp[str(cid)+"data"], linhatemp[str(cid)+"detalhes"], 0, selecionarComando)
352             passoUsuario[cid] = 0
353             with open('passo/{}.txt'.format(cid), 'w') as v:
354                 v.write("0")
355             del linhatemp[str(cid)+"titulo"]
356             del linhatemp[str(cid)+"data"]
357             del linhatemp[str(cid)+"detalhes"]
358     else:
359         bot.send_message(cid, "Quantos horas antes do evento você deseja ser lembrado?\n(0 número máximo que você pode digitar é: {})".format((linhatemp[str(cid)+"data"]-agora).total_seconds()//3600), reply_markup=esconderTeclado)
360         linhatemp[str(cid)+"casolembrete"] = 2
361         with open('linhatemp/{}.txt'.format(cid), 'a') as f:
362             f.write("1")
363         passoUsuario[cid] = 6
364         with open('passo/{}.txt'.format(cid), 'w') as v:
365             v.write("6")
366     elif mensagem.text == "NÃO":
367         salvar_evento(eventos, cid, linhatemp[str(cid)+"titulo"], linhatemp[str(cid)+"data"], linhatemp[str(cid)+"detalhes"], 0, selecionarComando)
368         del linhatemp[str(cid)+"titulo"]
369         del linhatemp[str(cid)+"data"]

```



Mensagem

En ☺ 123 ⌂ ⌂



```
368     del linhatemp[str(cid)+"titulo"]
369     del linhatemp[str(cid)+"data"]
370     del linhatemp[str(cid)+"detalhes"]
371     passoUsuario[cid] = 0
372     with open('passo/{}.txt'.format(cid), 'w') as v:
373         v.write("0")
374     else:
375         bot.send_message(cid, "Por favor selecione uma das opções", reply_markup=sim_ou_não)
376
377 # Verifica a data
378 @bot.message_handler(func=lambda message: passo_do_usuario(message.chat.id) == 6)
379 def lembrete1(mensagem):
380     cid = mensagem.chat.id
381     agora = datetime.datetime.now()
382     if linhatemp[str(cid)+"data"] < agora:
383         bot.send_message(cid, "Não é mais possível criar um lembrete.")
384         salvar_evento(eventos, cid, linhatemp[str(cid)+"titulo"], linhatemp[str(cid)+"data"], linhatemp[str(cid)+"detalhes"], 0, selecionarComando)
385         passoUsuario[cid] = 0
386         with open('passo/{}.txt'.format(cid), 'w') as v:
387             v.write("0")
388         del linhatemp[str(cid)+"titulo"]
389         del linhatemp[str(cid)+"data"]
390         del linhatemp[str(cid)+"detalhes"]
391         with open('linhatemp/{}.txt'.format(cid), 'w') as f:
392             f.write("")
393     else:
394         try:
395             tempo = int(mensagem.text)
396             if tempo < 0:
397                 raise ValueError
398             dataE = linhatemp[str(cid)+"data"]
399             titulo = linhatemp[str(cid)+"titulo"]
400             if linhatemp[str(cid)+"casolembrete"] == 1:
401                 lembrete = dataE - datetime.timedelta(days=tempo)
402             else:
403                 lembrete = dataE - datetime.timedelta(hours=tempo)
404             agora = datetime.datetime.now()
405             if lembrete < agora:
406                 if dataE < agora:
407                     bot.send_message(cid, "Não é mais possível criar um lembrete.")
408                     salvar_evento(eventos, cid, linhatemp[str(cid)+"titulo"], linhatemp[str(cid)+"data"], linhatemp[str(cid)+"detalhes"], 0, selecionarComando)
409                     passoUsuario[cid] = 0
410                     with open('passo/{}.txt'.format(cid), 'w') as v:
411                         v.write("0")
412                     del linhatemp[str(cid)+"titulo"]
413                     del linhatemp[str(cid)+"data"]
414                     del linhatemp[str(cid)+"detalhes"]
415                     with open('linhatemp/{}.txt'.format(cid), 'w') as f:
```



Chen

SIM

16:12:17



Eru

Quantos horas antes do evento você deseja ser lembrado?
(O número máximo que você pode digitar é: 0.0)

16:12:18



Chen

0

16:13:04



Eru

Não é possível criar um lembrete.

16:13:06

O que deseja fazer?

16:13:06



|Escreva uma mensagem...



ENVIAR

```

421     f.write(linhatemp[str(cid)+"titulo"]+"\n"+funcoes.datetime_para_texto(linhatemp[str(cid)+"data"])+"\n"+linhatemp[str(cid)+"detalhes"]+"1\n")
422 else:
423     bot.send_message(cid, "Quantos horas antes do evento você deseja ser lembrado?\n(0 número máximo que você pode digitar é: {}).format((linhatemp[str(cid)+"data"]-agora).total_seconds()//3600)")
424     linhatemp[str(cid)+"casoLembrete"] = 2
425     with open('linhatemp{}.txt'.format(cid), 'w') as f:
426         f.write(linhatemp[str(cid)+"titulo"]+"\n"+funcoes.datetime_para_texto(linhatemp[str(cid)+"data"])+"\n"+linhatemp[str(cid)+"detalhes"]+"2\n")
427 else: #2
428     bot.send_message(cid, "Lembrete definido para {}".format(funcoes.datetime_para_texto(lembrete)))
429     salvar_evento(eventos, cid, linhatemp[str(cid)+"titulo"], linhatemp[str(cid)+"data"], linhatemp[str(cid)+"detalhes"], lembrete, selecionarComando)
430     del linhatemp[str(cid)+"titulo"]
431     del linhatemp[str(cid)+"data"]
432     del linhatemp[str(cid)+"detalhes"]
433     with open('linhatemp{}.txt'.format(cid), 'w') as f:
434         f.write("")
435     passoUsuario[cid] = 0
436     with open('passo{}.txt'.format(cid), 'w') as v:
437         v.write("0")
438     st = (lembrete - agora).total_seconds()
439     lembretes[cid][titulo+funcoes.datetime_para_texto(dataE)] = fila.enter(st, 1, enviar_lembrete, argument=(cid, titulo, funcoes.datetime_para_texto(dataE)))
440     t = threading.Thread(target=fila.run)
441     t.start()
442 except ValueError:
443     agora = datetime.datetime.now()
444     if linhatemp[str(cid)+"data"] < agora:
445         bot.send_message(cid, "Não é mais possível criar um lembrete.")
446         salvar_evento(cid, eventos, linhatemp[str(cid)+"titulo"], linhatemp[str(cid)+"data"], linhatemp[str(cid)+"detalhes"], 0, selecionarComando)
447         passoUsuario[cid] = 0
448         with open('passo{}.txt'.format(cid), 'w') as v:
449             v.write("0")
450         del linhatemp[str(cid)+"titulo"]
451         del linhatemp[str(cid)+"data"]
452         del linhatemp[str(cid)+"detalhes"]
453         with open('linhatemp{}.txt'.format(cid), 'w') as f:
454             f.write("")
455     else:
456         bot.send_message(cid, "Valor inválido! Por favor digite um número não negativo")
457
458 # O que acontece quando o usuário seleciona a opção lista de eventos
459 @bot.message_handler(func=lambda message: message.text == comandos[1] and passo_do_usuario(message.chat.id) == 0)
460 def lista_de_eventos(mensagem):
461     cid = mensagem.chat.id
462     if len(eventos[cid]) == 0:
463         bot.send_message(cid, "Não há eventos salvos", reply_markup=esconderTeclado)
464         bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
465     else:
466         texto1 = "Selecione qual evento deseja editar:\n\n"
467         for i in range(len(eventos[cid])):
468             texto1 += "{}) {}:{}\n".format(i+1, eventos[cid][i][0], funcoes.datetime_para_texto(eventos[cid][i][1]))
469         bot.send_message(cid, texto1, reply_markup=teclado_editar(len(eventos[cid])))
470         passoUsuario[cid] = 10
471         with open('passo{}.txt'.format(cid), 'w') as v:
472             v.write("10")
473
474 opcoesEditar = ["Título", "Data", "Detalhes", "Lembrete", "Deletar", "CANCELAR"]
475 ocoesTeclado = tvoes.ReplyKeyboardMarkup()

```



Chen

/start

07:35:21



Eru

Bem vindo ao AgendaBot, gostaria de adicionar algum evento?

07:35:23



Chen

Lista de eventos

07:36:47



Eru

Não há eventos salvos

07:36:48

O que deseja fazer?

07:36:48



Escreva uma mensagem...



ER



ENVIAR

Adicionar evento

Lista de eventos

```

475 opcoesTeclado = types.ReplyKeyboardMarkup()
476 opcoesTeclado.add(*opcoesEditar)
477
478 # Caso não consiga ler algo, só ignora
479 editartemp = {}
480 for cid in usuariosConhecidos:
481     try:
482         with open('editartemp/{}.txt'.format(cid), 'r') as f:
483             if f.readlines()[0] != "":
484                 editartemp[cid] = int(f.readlines()[0])
485     except FileNotFoundError:
486         f = open('editartemp/{}.txt'.format(cid), 'x')
487         f.close()
488     except IndexError:
489         pass
490
491 editartempcaso = {}
492 for cid in usuariosConhecidos:
493     try:
494         with open('editartempcaso/{}.txt'.format(cid), 'r') as f:
495             if f.readlines()[0] != "":
496                 editartemp[cid] = int(f.readlines()[0])
497     except FileNotFoundError:
498         f = open('editartempcaso/{}.txt'.format(cid), 'x')
499         f.close()
500     except IndexError:
501         pass
502
503 # Ao escolher um dos números, mostra todas as informações e pergunta qual delas deseja editar, caso escolha a opção 'cancelar' volta para o começo, e se não escolher nenhuma das opções, o bot mandará uma mensagem pedindo para escolher uma.
504 @bot.message_handler(func=lambda message: passo_do_usuario(message.chat.id) == 10)
505 def editar(mensagem):
506     cid = mensagem.chat.id
507     if mensagem.text in [str(n)+"\" for n in range(1, len(eventos[cid]) + 1)]:
508         edt = int(mensagem.text[-1]) - 1
509         editartemp[cid] = edt
510         with open('editartemp/{}.txt'.format(cid), 'w') as f:
511             f.write(str(edt))
512         if eventos[cid][edt][3] == 0:
513             texto2 = "Título:\n{}\\nData:\n{}\\nDetalhes:\n{}\\nLembrete:\nNão definido".format(eventos[cid][edt][0], funcoes.datetime_para_texto(eventos[cid][edt][1]), eventos[cid][edt][2])
514         else:
515             texto2 = "Título:\n{}\\nData:\n{}\\nDetalhes:\n{}\\nLembrete:\n{}".format(eventos[cid][edt][0], funcoes.datetime_para_texto(eventos[cid][edt][1]), eventos[cid][edt][2], funcoes.datetime_para_texto(eventos[cid][edt][3]))
516         bot.send_message(cid, texto2, reply_markup=opcoesTeclado)
517         passoUsuario[cid] = 12
518         with open('passo/{}.txt'.format(cid), 'w') as v:
519             v.write("12")
520     elif mensagem.text == "CANCELAR":
521         passoUsuario[cid] = 0
522         with open('passo/{}.txt'.format(cid), 'w') as v:
523             v.write("0")
524         bot.send_message(cid, "O que gostaria de fazer?", reply_markup=selecionarComando)
525     else:
526         bot.send_message(cid, "Por favor selecione uma das opções", reply_markup=teclado_editar(len(eventos[cid])))
527

```

ER

Eru

Selecione qual evento deseja editar:

16:18:31

- 1) ~time to sleep~: 15/02/10 12:02
- 2) Something: 25/11/2018 16:13
- 3) anything: 25/11/2018 16:09
- 4) Prova: 28/11/2018 13:30
- 5) Sleep time: 29/11/2018 11:50
- 6) Trabalho e apresentação de programação: 28/11/2019 15:35



Chen

16:18:31

CANCELAR

ER

Eru

16:18:33

O que gostaria de fazer?



Escreva uma mensagem...



ER



ENVIAR

[Adicionar evento](#)[Lista de eventos](#)

```
# Pede a informação nova caso seja possível, ou faz a ação de deletar ou cancelar.
529 @bot.message_handler(func=lambda message: passo_do_usuario(message.chat.id) == 12)
530 def editar2(mensagem):
531     cid = mensagem.chat.id
532     agora = datetime.datetime.now()
533     if mensagem.text == "Título":
534         bot.send_message(cid, "Digite o novo título:", reply_markup=esconderTeclado)
535         editartempcaso[cid] = 1
536         with open('editartempcaso/{}.txt'.format(cid), 'w') as f:
537             f.write("1")
538         passoUsuario[cid] = 13
539         with open('passo/{}.txt'.format(cid), 'w') as v:
540             v.write("13")
541     elif mensagem.text == "Data":
542         bot.send_message(cid, "Digite a nova data (no formato DD/MM/AAAA):", reply_markup=esconderTeclado)
543         editartempcaso[cid] = 2
544         with open('editartempcaso/{}.txt'.format(cid), 'w') as f:
545             f.write("2")
546         passoUsuario[cid] = 13
547         with open('passo/{}.txt'.format(cid), 'w') as v:
548             v.write("13")
549     elif mensagem.text == "Detalhes":
550         bot.send_message(cid, "Digite os novos detalhes:", reply_markup=esconderTeclado)
551         editartempcaso[cid] = 3
552         with open('editartempcaso/{}.txt'.format(cid), 'w') as f:
553             f.write("3")
554         passoUsuario[cid] = 13
555         with open('passo/{}.txt'.format(cid), 'w') as v:
556             v.write("13")
557     elif mensagem.text == "Lembrete":
558         if eventos[cid][editartemp[cid]][1] < datetime.datetime.now():
559             bot.send_message(cid, "Não é possível fazer isso! Esse evento já aconteceu!", reply_markup=esconderTeclado)
560             passoUsuario[cid] = 0
561             with open('passo/{}.txt'.format(cid), 'w') as v:
562                 v.write("0")
563             del editartemp[cid]
564             with open('editartemp/{}.txt'.format(cid), 'w') as f:
565                 f.write("")
566             bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
567         elif (eventos[cid][editartemp[cid]][1] - datetime.datetime.now()).days >= 1:
568             bot.send_message(cid, "Quantos dias antes do evento você deseja ser lembrado?\n(0 número máximo que você pode digitar é: {})".format((eventos[cid][editartemp[cid]][1]-agora).days), reply_markup=esconderTeclado)
569             editartempcaso[cid] = 4
570             with open('editartempcaso/{}.txt'.format(cid), 'w') as f:
571                 f.write("4")
572             passoUsuario[cid] = 13
573             with open('passo/{}.txt'.format(cid), 'w') as v:
574                 v.write("13")
575         else:
576             bot.send_message(cid, "Quantos horas antes do evento você deseja ser lembrado?\n(0 número máximo que você pode digitar é: {})".format((eventos[cid][editartemp[cid]][1]-agora).total_seconds()//3600), reply_markup=esconderTeclado)
577             editartempcaso[cid] = 5
578             with open('editartempcaso/{}.txt'.format(cid), 'w') as f:
579                 f.write("5")
580             passoUsuario[cid] = 13
581             with open('passo/{}.txt'.format(cid), 'w') as v:
582                 v.write("13")
```



5) Sleep time: 29/11/2018 11:50

4) Trabalho e apresentação de programação: 28/11/2019 15:35



Chen

08:06:23

3)



Eru

08:06:24

Título:

Sleep time

Data:

29/11/2018 11:50

Detalhes:

~Sorry, it's my time to sleep, bye O/~/

Lembrete:

Não definido



Chen

08:06:53

Lembrete



Eru

08:06:54

Quantos dias antes do evento você deseja ser lembrado?

(O número máximo que você pode digitar é: 4)



Escreva uma mensagem...



ER



ENVIAR

```

580     passouUsuario[cid] = 13
581     with open('passo/{}.txt'.format(cid), 'w') as v:
582         v.write("13")
583 elif mensagem.text == "Deletar":
584     if eventos[cid][editartemp[cid]][3] != 0:
585         if datetime.datetime.now() < eventos[cid][editartemp[cid]][3]:
586             fila.cancel(lembrete[cid][eventos[cid][editartemp[cid]][0]]+funcoes.datetime_para_texto(eventos[cid][editartemp[cid]][1]))
587         del eventos[cid][editartemp[cid]]
588         funcoes.salvar_edicao(eventos, cid)
589         del editartemp[cid]
590         with open('editartemp/{}.txt'.format(cid), 'w') as f:
591             f.write("")
592         passouUsuario[cid] = 0
593         with open('passo/{}.txt'.format(cid), 'w') as v:
594             v.write("0")
595         bot.send_message(cid, "Evento deletado com sucesso!", reply_markup=esconderTeclado)
596         bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
597 elif mensagem.text == "CANCELAR":
598     passouUsuario[cid] = 0
599     with open('passo/{}.txt'.format(cid), 'w') as v:
600         v.write("0")
601     bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
602 else:
603     bot.send_message(cid, "Por favor selecione uma das opções", reply_markup=opcoesTeclado)
604
605 editartempdata = {}
606 for cid in usuariosConhecidos:
607     try:
608         with open('editartempdata/{}.txt'.format(cid), 'r') as f:
609             data = [int(n) for n in f.readline().split("/")]
610             if len(data) == 3:
611                 editartempdata[cid] = datetime.datetime(data[-1], data[-2], data[-3])
612     except FileNotFoundError:
613         f = open('editartempdata/{}.txt'.format(cid), 'x')
614         f.close()
615     except ValueError:
616         pass
617
618 # Faz a edição de cada caso (título, data, detalhes, lembrete, deletar e cancelar); a alteração da data é feita em três etapas, nesse def é a primeira.
619 @bot.message_handler(func=lambda message: passo_do_usuario(message.chat.id) == 13)
620 def editar2(m):
621     cid = m.chat.id
622     if editartempcaso[cid] == 1:
623         tituloNovo = m.text
624         if eventos[cid][editartemp[cid]][3] != 0:
625             if datetime.datetime.now() < eventos[cid][editartemp[cid]][3]:
626                 fila.cancel(lembrete[cid][eventos[cid][editartemp[cid]][0]]+funcoes.datetime_para_texto(eventos[cid][editartemp[cid]][1]))
627                 st = (eventos[cid][editartemp[cid]][3] - datetime.datetime.now()).total_seconds()
628                 lembretes[cid][tituloNovo+funcoes.datetime_para_texto(eventos[cid][editartemp[cid]][1])] = fila.enter(st, 1, enviar_lembrete, argument=(cid, tituloNovo, funcoes.datetime_para_texto(eventos[cid][editartemp[cid]][1])))
629                 t = threading.Thread(target=fila.run)
630                 t.start()
631             eventos[cid][editartemp[cid]][0] = tituloNovo
632             funcoes.salvar_edicao(eventos, cid)
633             bot.send_message(cid, "Título editado com sucesso!")

```



Não definido



Chen

Título

07:50:39



Eru

Digite o novo título:

07:50:40



Chen

Trabalho e apresentação de programação

07:51:33



Eru

Título editado com sucesso!

07:51:35

O que deseja fazer?

07:51:35



Escreva uma mensagem...



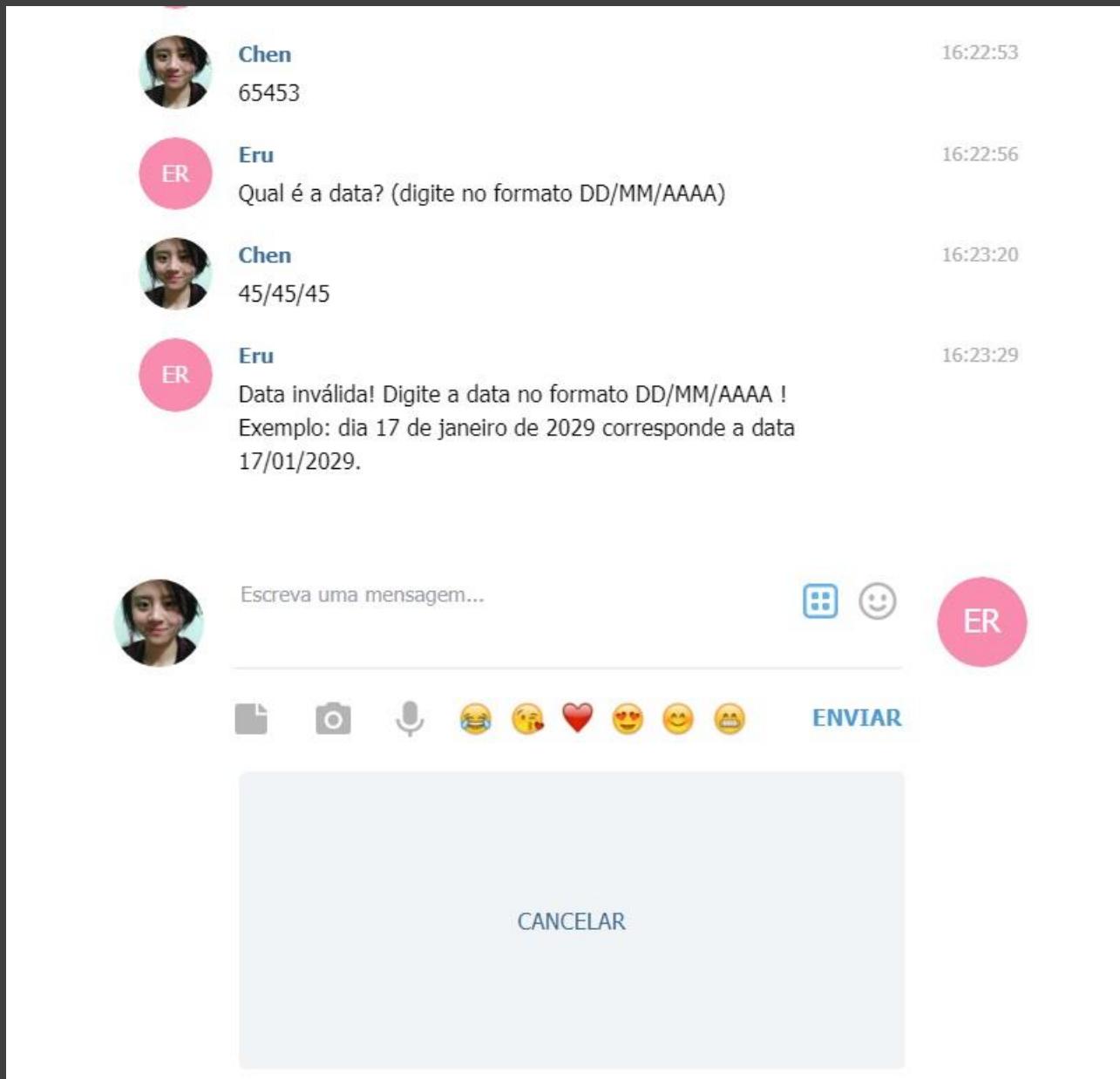
ENVIAR



Adicionar evento

Lista de eventos

```
633 bot.send_message(cid, "Título editado com sucesso!")
634 del editartempcaso[cid]
635 with open('editartempcaso/{}.txt'.format(cid), 'w') as f:
636     f.write("")
637 del editartemp[cid]
638 with open('editartemp/{}.txt'.format(cid), 'w') as f:
639     f.write("")
640 passoUsuario[cid] = 0
641 with open('passo/{}.txt'.format(cid), 'w') as v:
642     v.write("0")
643 bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
644
645 elif editartempcaso[cid] == 2:
646     try:
647         dataNova = [int(n) for n in m.text.split("/")]
648         validar = funcoes.validar_data(dataNova[0], dataNova[1], dataNova[2])
649         if validar == False or len(dataNova) != 3:
650             raise ValueError("Data inválida")
651         with open('editartempdata/{}.txt'.format(cid), 'w') as f:
652             f.write(m.text)
653         editartempdata[cid] = datetime.datetime(dataNova[-1], dataNova[-2], dataNova[-3])
654         bot.send_message(cid, "Digite o novo horário (no formato HH:MM): ")
655         passoUsuario[cid] = 14
656         with open('passo/{}.txt'.format(cid), 'w') as v:
657             v.write("14")
658     except (ValueError, IndexError):
659         bot.send_message(cid, "Data inválida! Digite a data no formato DD/MM/AAAA !\nExemplo: dia 17 de janeiro de 2029 corresponde a data 17/01/2029.")
660
661 elif editartempcaso[cid] == 3:
662     detalhesNovo = m.text
663     eventos[cid][editartemp[cid]][2] = detalhesNovo
664     funcoes.salvar_edicao(eventos, cid)
665     bot.send_message(cid, "Detalhes editados com sucesso!")
666     del editartempcaso[cid]
667     with open('editartempcaso/{}.txt'.format(cid), 'w') as f:
668         f.write("")
669     del editartemp[cid]
670     with open('editartemp/{}.txt'.format(cid), 'w') as f:
671         f.write("")
672     passoUsuario[cid] = 0
673     with open('passo/{}.txt'.format(cid), 'w') as v:
674         v.write("0")
675     bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
676
677 elif editartempcaso[cid] == 4 or editartempcaso[cid] == 5:
678     try:
679         tempo = int(m.text)
680         if tempo < 0:
681             raise ValueError
682         titulo = eventos[cid][editartemp[cid]][0]
683         if editartempcaso[cid] == 4:
684             lembrete = eventos[cid][editartemp[cid]][1] - datetime.timedelta(days=tempo)
685         else:
686             lembrete = eventos[cid][editartemp[cid]][1] - datetime.timedelta(hours=tempo)
687         agora = datetime.datetime.now()
```



```

686 lembrete = eventos[cid][editartemp[cid]][1] - datetime.timedelta(hours=tempo)
687 agora = datetime.datetime.now()
688 if lembrete < agora:
689     if eventos[cid][editartemp[cid]][1] < agora:
690         bot.send_message(cid, "Não é possível criar um lembrete.")
691         bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
692         passoUsuario[cid] = 0
693         with open('passo/{}.txt'.format(cid), 'w') as v:
694             v.write("0")
695     elif (eventos[cid][editartemp[cid]][1] - agora).days >= 1:
696         bot.send_message(cid, "Número inválido!\nQuantos dias antes do evento você deseja ser lembrado?\n(O número máximo que você pode digitar é: {})".format((eventos[cid][editartemp[cid]][1] - agora).days))
697         editartempcaso[cid] == 4
698         with open('editartempcaso/{}.txt'.format(cid), 'w') as f:
699             f.write("4")
700     else:
701         bot.send_message(cid, "Número inválido!\nQuantos horas antes do evento você deseja ser lembrado?\n(O número máximo que você pode digitar é: {})".format((eventos[cid][editartemp[cid]][1]-agora).total_seconds()//3600))
702         editartempcaso[cid] == 5
703         with open('editartempcaso/{}.txt'.format(cid), 'w') as f:
704             f.write("5")
705     else:
706         try:
707             fila.cancel(lembretes[cid][eventos[cid][editartemp[cid]][0]+funcoes.datetime_para_texto(eventos[cid][editartemp[cid]][1])])
708         except KeyError:
709             pass
710         st = (lembrete - datetime.datetime.now()).total_seconds()
711         eventos[cid][editartemp[cid]][3] = lembrete
712         lembretes[cid][eventos[cid][editartemp[cid]][0]+funcoes.datetime_para_texto(eventos[cid][editartemp[cid]][1])] = fila.enter(st, 1, enviar_lembrete, argument=(cid, eventos[cid][editartemp[cid]][0],
713                                         funcoes.datetime_para_texto(eventos[cid][editartemp[cid]][1])))
714         t = threading.Thread(target=fila.run)
715         t.start()
716         bot.send_message(cid, "Lembrete definido para {}".format(funcoes.datetime_para_texto(lembrete)))
717         funcoes.salvar_edicao(eventos, cid)
718         passoUsuario[cid] = 0
719         with open('passo/{}.txt'.format(cid), 'w') as v:
720             v.write("0")
721         bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
722
723     except ValueError:
724         agora = datetime.datetime.now()
725         if eventos[cid][editartemp[cid]][1] < agora:
726             bot.send_message(cid, "Não é possível criar um lembrete.")
727             bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
728             passoUsuario[cid] = 0
729             with open('passo/{}.txt'.format(cid), 'w') as v:
730                 v.write("0")
731         else:
732             bot.send_message(cid, "Valor inválido! Por favor digite um número não negativo")
733
734 # Segunda parte da data vai perguntar se quer um lembrete
735 @bot.message_handler(func=lambda message: passo_do_usuario(message.chat.id) == 14)
736 def editar_horario(mensagem):
737     try:
738         cid = mensagem.chat.id
739         horarioNovo = mensagem.text
740

```





Chen

21:50

16:25:19



Eru

16:25:22

Digite mais detalhes sobre o evento:



Chen

16:25:25

dasda



Eru

16:25:35

Deseja adicionar lembrete?



Chen

16:25:45

SIM



Eru

16:25:47

Quantos dias antes do evento você deseja ser lembrado?
(O número máximo que você pode digitar é: 3706)



Chen

16:25:51

lkb



Eru

16:25:54

Valor inválido! Por favor digite um número não negativo



|Escreva uma mensagem...



ER



ENVIAR

```

736 def editar_horario(mensagem):
737     try:
738         cid = mensagem.chat.id
739         horarioNovo = mensagem.text
740         horarioNovo = [int(n) for n in horarioNovo.split(":")]
741         if horarioNovo[0] < 0 or horarioNovo[0] > 23:
742             raise ValueError("Hora inválida")
743         if horarioNovo[1] < 0 or horarioNovo[1] > 59:
744             raise ValueError("Minuto inválido")
745         if len(horarioNovo) != 2:
746             raise ValueError("Horário inválido")
747         editartempdata[cid] = editartempdata[cid].replace(hour=horarioNovo[0], minute=horarioNovo[1])
748         newDate = editartempdata[cid]
749         if eventos[cid][editartemp[cid]][3] != 0:
750             if datetime.datetime.now() < eventos[cid][editartemp[cid]][3]:
751                 fila.cancel(lembretes[cid][eventos[cid][editartemp[cid]][0]+funcoes.datetime_para_texto(eventos[cid][editartemp[cid]][1])])
752             eventos[cid][editartemp[cid]][1] = newDate
753             del editartempdata[cid]
754             with open('editartempdata/{}.txt'.format(cid), 'w') as f:
755                 f.write("")
756             eventos[cid][editartemp[cid]][3] = 0
757             if newDate < datetime.datetime.now():
758                 for k in range(len(eventos[cid])):
759                     for i in range(1, len(eventos[cid]) - k):
760                         if eventos[cid][i][1] < eventos[cid][i-1][1]:
761                             eventos[cid][i], eventos[cid][i-1] = eventos[cid][i-1], eventos[cid][i]
762             funcoes.salvar_edicao(eventos, cid)
763             bot.send_message(cid, "Data editada com sucesso!")
764             passoUsuario[cid] = 0
765             with open('passo/{}.txt'.format(cid), 'w') as v:
766                 v.write("0")
767             bot.send_message(cid, "O que deseja fazer?", reply_markup=selecionarComando)
768         else:
769             bot.send_message(cid, "Data editada com sucesso!\nDeseja colocar um lembrete?", reply_markup=sim_ou_não)
770             passoUsuario[cid] = 15
771             with open('passo/{}.txt'.format(cid), 'w') as v:
772                 v.write("15")
773     except (ValueError, IndexError):
774         bot.send_message(cid, "Horário inválido! Digite o horário no formato HH:MM !\nExemplo: duas e quarenta e cinco da tarde são 14:45")
775
776 # Terceira parte da data o que o bot faz caso a resposta seja 'sim' ou 'não'
777 @bot.message_handler(func=lambda message: passo_do_usuario(message.chat.id) == 15)
778 def editar_horario2(m):
779     cid = m.chat.id
780     if m.text == "SIM":
781         agora = datetime.datetime.now()
782         if (eventos[cid][editartemp[cid]][1]-agora).days >= 1:
783             bot.send_message(cid, "Quantos dias antes do evento você deseja ser lembrado?\n(0 número máximo que você pode digitar é: {})".format((eventos[cid][editartemp[cid]][1]-agora).days), reply_markup=esconderTeclado)
784             editartempcaso[cid] = 4
785             with open('editartempcaso/{}.txt'.format(cid), 'w') as f:
786                 f.write("4")
787             passoUsuario[cid] = 13
788             with open('passo/{}.txt'.format(cid), 'w') as v:
789                 v.write("13")
790

```

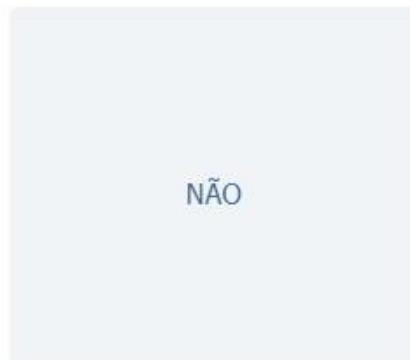
-  **Eru** 07:53:07
Digite a nova data (no formato DD/MM/AAAA):
-  **Chen** 07:53:55
28/11/2019
-  **Eru** 07:53:56
Digite o novo horário (no formato HH:MM):
-  **Chen** 07:54:36
15:35
-  **Eru** 07:54:38
Data editada com sucesso!
Deseja colocar um lembrete?



Escreva uma mensagem...



ENVIAR



SIM

NÃO

```
889     v.write("13")
890 elif eventos[cid][editartemp[cid]][1] < agora:
891     bot.send_message(cid, "Não é mais possível criar um lembrete.")
892     for k in range(len(eventos[cid])):
893         for i in range(1, len(eventos[cid]) - k):
894             if eventos[cid][i][1] < eventos[cid][i-1][1]:
895                 eventos[cid][i], eventos[cid][i-1] = eventos[cid][i-1], eventos[cid][i]
896     funcoes.salvar_edicao(eventos, cid)
897     bot.send_message(cid, "Edição concluída!\n0 que deseja fazer?", reply_markup=selecionarComando)
898     passoUsuario[cid] = 0
899     with open('passo/{}.txt'.format(cid), 'w') as v:
900         v.write("0")
901 else:
902     bot.send_message(cid, "Quantos horas antes do evento você deseja ser lembrado?\n0 número máximo que você pode digitar é: {}".format((eventos[cid][editartemp[cid]][1]-agora).total_seconds()//3600), reply_markup=esconderTeclado)
903     editartempcaso[cid] = 5
904     with open('editartempcaso/{}.txt'.format(cid), 'w') as f:
905         f.write("5")
906     passoUsuario[cid] = 13
907     with open('passo/{}.txt'.format(cid), 'w') as v:
908         v.write("13")
909 elif m.text == "NÃO" or eventos[cid][editartemp[cid]][1] < datetime.datetime.now():
910     for k in range(len(eventos[cid])):
911         for i in range(1, len(eventos[cid]) - k):
912             if eventos[cid][i][1] < eventos[cid][i-1][1]:
913                 eventos[cid][i], eventos[cid][i-1] = eventos[cid][i-1], eventos[cid][i]
914     funcoes.salvar_edicao(eventos, cid)
915     bot.send_message(cid, "Edição concluída!\n0 que deseja fazer?", reply_markup=selecionarComando)
916     passoUsuario[cid] = 0
917     with open('passo/{}.txt'.format(cid), 'w') as v:
918         v.write("0")
919 else:
920     bot.send_message(cid, "Por favor selecione uma das opções", reply_markup=sim_ou_não)
921
922 bot.polling() # Executa o bot
```

Exemplo: data é quarenta e cinco da tarde São 14:15



Chen

19:00

16:29:15



Eru

Data editada com sucesso!
Deseja colocar um lembrete?

16:29:17



Chen

NÃO

16:29:31



Eru

Edição concluída!
O que deseja fazer?

16:29:37



Escreva uma mensagem...



ER



ENVIAR

Adicionar evento

Lista de eventos



Obrigado & Obrigada!