
**Universidad Tecnológica de Santiago, UTESA Sistema
Corporativo**

**Facultad de Ingeniería y Arquitectura
Carrera de Ingeniería en Sistemas Computacionales**



Inteligencia Artificial

INF-503-001

Asignación:
Asignación Final

Por:
Starling Vasquez 1-17-0155

Para
Tomás F. Ramírez Pérez

Descripción de proyecto

El proyecto Fune Bot está enfocado en servir vía un chatbot a esas empresas que desean vender productos o servicios. El chatbot está enfocado en simplificar el proceso de compras para el cliente. El cliente podrá realizar todas las operaciones necesarias para adquirir los productos a través del chatbot.

Objetivo

Adquirir una buena reputación a través de un servicio de calidad, garantizando que el cliente tenga una experiencia de usuario agradable y sin problemas, así como recibir asistencia cuando sea necesario. Asegurándonos de que el usuario sienta que los servicios ofrecidos cumplen con las expectativas.

Características principales

- La aplicación cuenta con factores necesarios para que haya una buena experiencia de usuario, siendo amigable y fácil de usar.
- Identificación de palabras claves para responder al usuario.
- Permite identificar a los usuarios por medio de un id único.
- Los productos poseen la información necesaria para que el usuario pueda elegir
- Carrito de compras y proceso de pago.
- Envío de correo para confirmar la orden.
- Ventana de 40 min para cancelar su orden luego de realizada.

Comandos destacados utilizados en el bot

/start - Inicia el chat con el bot

/ayuda - Muestra esta ayuda

suscribirse: para recibir versículos y frases diarios.

desuscribirse: para anular la suscripcion.

flores/productos: para ver todas las flores disponibles de Alas de Paz.

agregar nombre_producto: para ver descripcion del arreglo floral y poder ser agregado al carrito.

correo correo@electrónico: configurar correo.

Teléfono: configurar tu número telefónico.

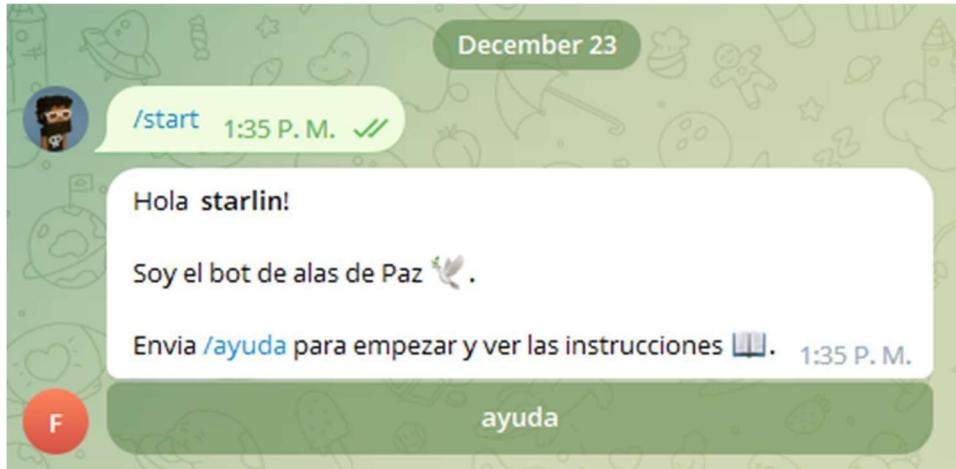
carrito: ver la orden actual.

Ejemplo de Interfaz

Telegram Bot

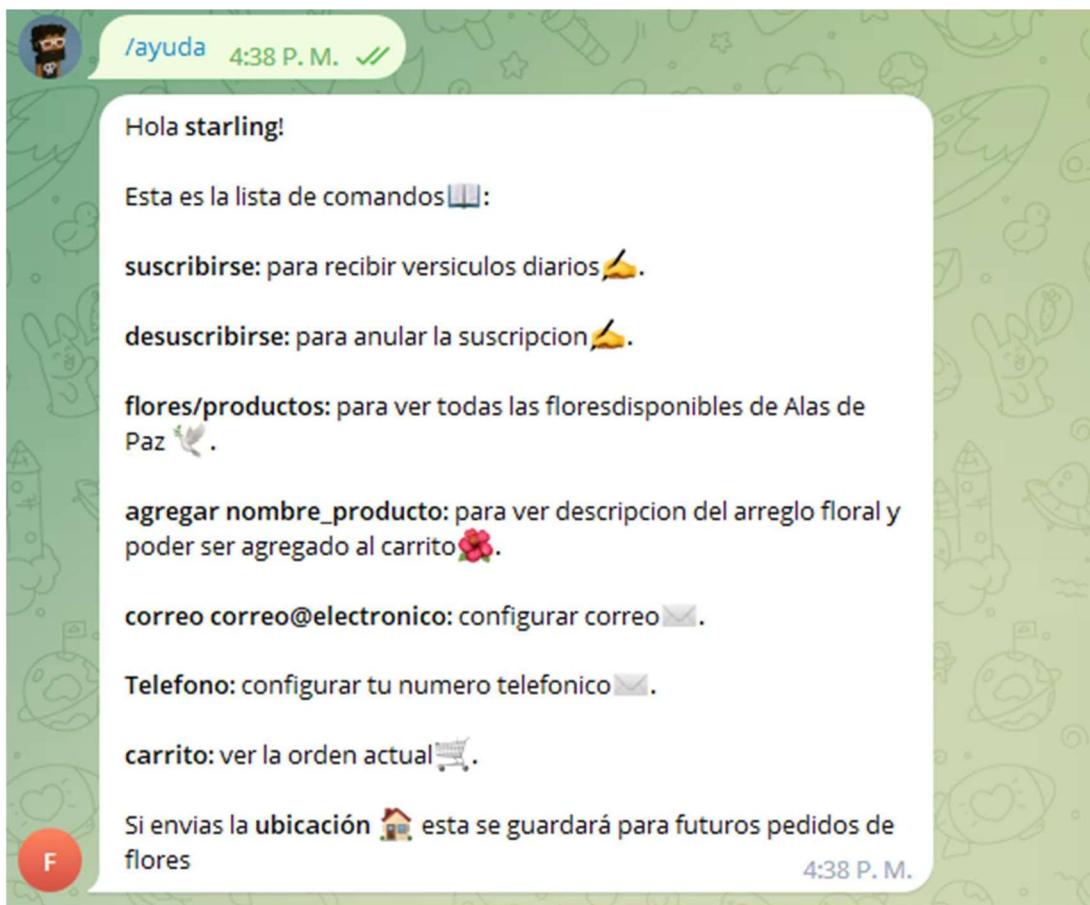
#=====

COMANDO DE INCIO



#=====

COMANDO DE AYUDA



| Write a message...

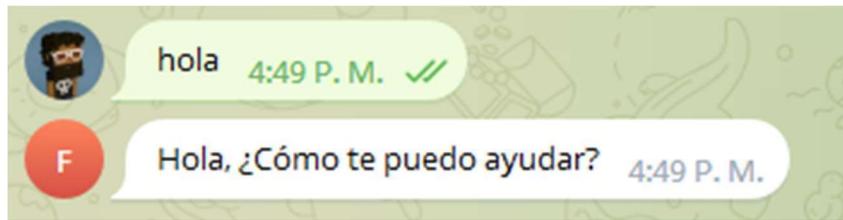
#=====

BOTONES PARA COMANDOS



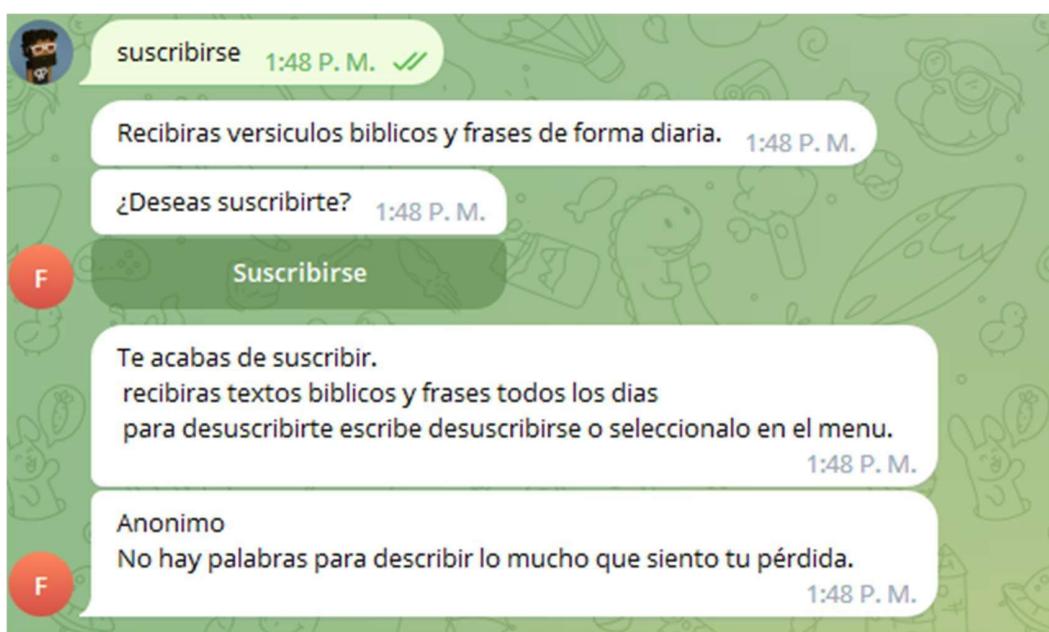
#=====

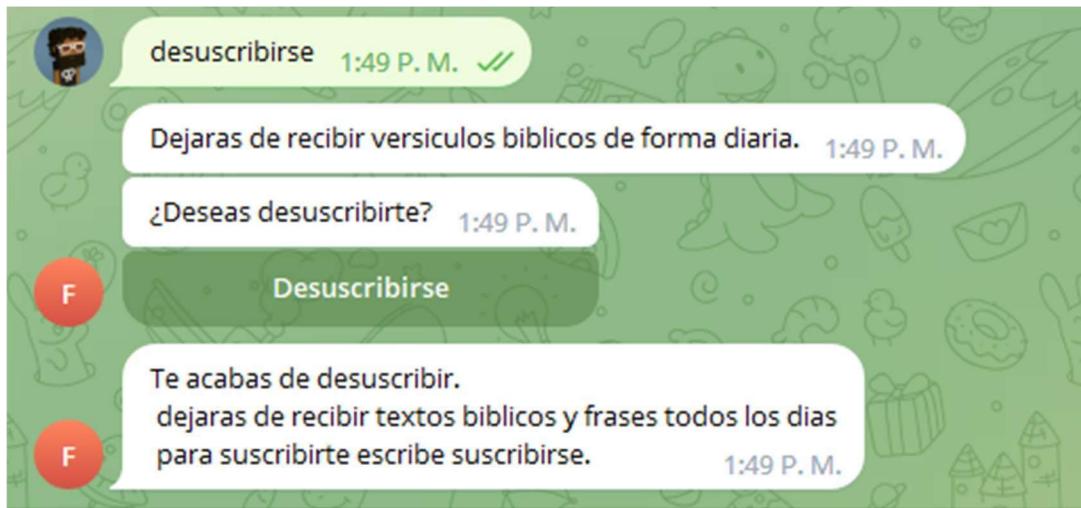
SALUDOS AL BOT



#=====

SUSCRIPCION Y DES SUSCRIPCION PARA RECIBIR VERSICULOS Y FRASES DE DUELO





#=====

LISTADO DE LOS ARREGLOS FLORALES DISPONIBLES



#=====

AGREGAR LOS ARREGLOS FLORALES AL CARRITO

agregar AMOROSSA 4:41 P.M. ✓



Arreglo Floral: AMOROSSA
Precio: RD\$500
Descripción: Hermoso arreglo de flores. Divinas para recordar

4:41 P.M.

¿Quieres añadirla al pedido 4:41 P.M.

Añadir

F

Arreglo floral añadido al carrito 🛒

F Para ver su pedido escriba carrito. 4:42 P.M.



| Write a message...

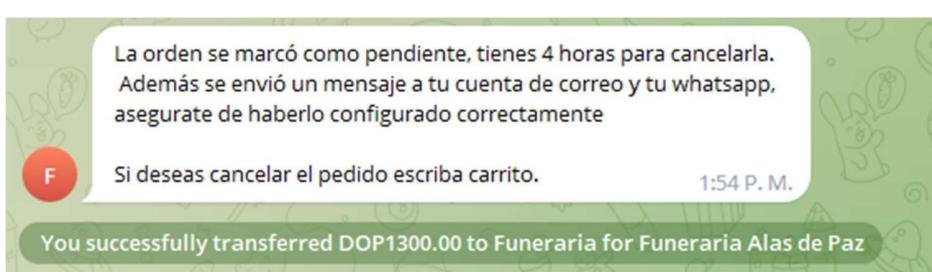
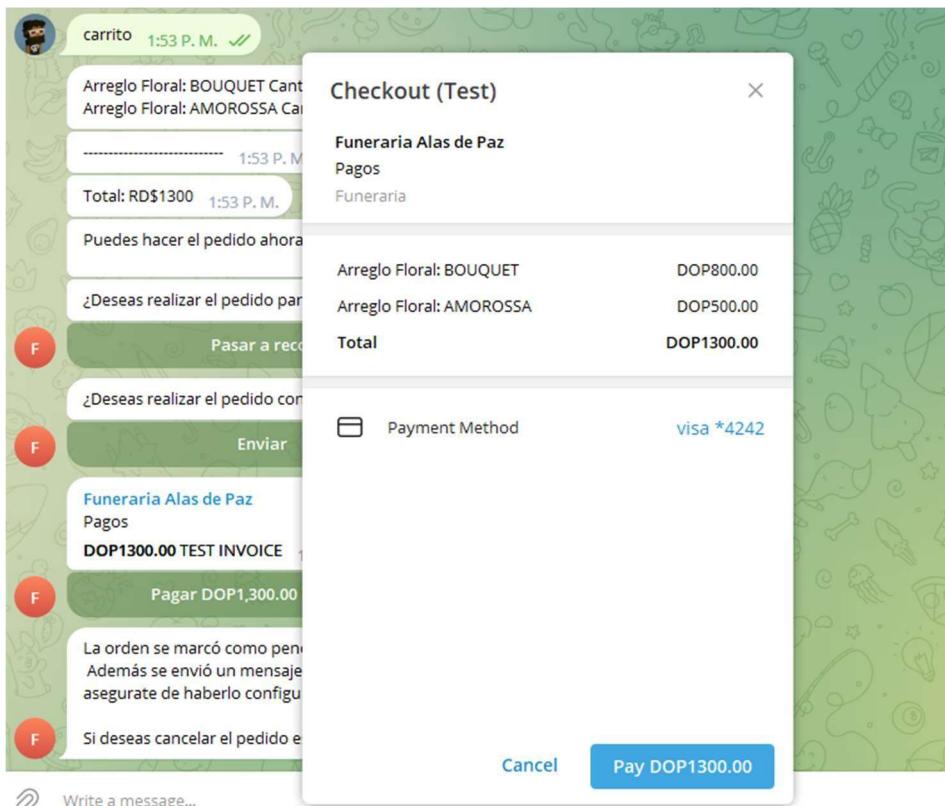
#=====

VISUALIZAR EL CARRITO PARA REALIZAR EL PEDIDO



#=====

REALIZAR EL PEDIDO PARA RECOLECCION (NO SE COBRA ENVIO) AL PAGAR SE COMPLETA LA TRANSACCION



#=====

REALIZAR EL PEDIDO PARA ENVIO
(SI EL ENVIO VA DIRIJIDO A UNA UBICACIÓN CON MENOS DE 5KM NO COBRA CARGO ADICIONAL)
AL PAGAR SE COMPLETA LA TRANSACCION

The screenshot shows a mobile application interface for a florist service. On the left, a green-themed chat window displays a conversation between a customer and a florist named 'Funeraria Alas de Paz'. The florist has sent a receipt for two items: 'Arreglo Floral: BOUQUET' and 'Arreglo Floral: AMOROSSA', both at a price of RD\$800 and RD\$500 respectively, totaling RD\$1300. The customer asks if they want to pick up or send the order. The florist responds with a payment link: 'DOP1300.00 TEST INVOICE'. The customer then sends a message confirming they will pay. A red circle with a white 'F' icon is visible next to each message from the florist.

carrito 1:53 P. M. ✓

Arreglo Floral: BOUQUET Cantidad: 1 Precio: RD\$800
Arreglo Floral: AMOROSSA Cantidad: 1 Precio: RD\$500 1:53 P. M.

----- 1:53 P. M.

Total: RD\$1300 1:53 P. M.

Puedes hacer el pedido ahora o seguir añadiendo productos 1:53 P. M.

¿Deseas realizar el pedido para recoger? 1:53 P. M.

F Pasar a recojer

¿Deseas realizar el pedido con envio? 1:53 P. M.

F Enviar

Funeraria Alas de Paz
Pagos
DOP1300.00 TEST INVOICE 2:19 P. M.

F Pagar DOP1,300.00

La orden se marcó como pendiente, se enviara a la ubicacion seleccionada **sin cargo adicional**, tienes 4 horas para cancelarla. Además se envió un mensaje a tu cuenta de correo y tu whatsapp, asegurate de haberlo configurado correctamente

F Si deseas cancelar el pedido escribe carrito. 2:19 P. M.

Checkout (Test) X

Funeraria Alas de Paz
Pagos
Funeraria

Item	Price
Arreglo Floral: BOUQUET	DOP800.00
Arreglo Floral: AMOROSSA	DOP500.00
Total	DOP1300.00

Payment Method: visa *4242

Shipping Address: viale vede, calle 10, sant...

Name: starling

Email: jose@dagjng.com

Phone: +1809 471 7008

Cancel Pay DOP1300.00

La orden se marcó como pendiente, se enviara a la ubicacion seleccionada **sin cargo adicional**, tienes 4 horas para cancelarla. Además se envió un mensaje a tu cuenta de correo y tu whatsapp, asegurate de haberlo configurado correctamente

F Si deseas cancelar el pedido escribe carrito. 2:19 P. M.

You successfully transferred DOP1300.00 to Funeraria for Funeraria Alas de Paz

#=====

REALIZAR EL PEDIDO PARA ENVIO

(SI EL ENVIO VA DIRIJIDO A UNA UBICACIÓN CON MAS DE 5KM SE COBRA UN CARGO ADICIONAL)

AL PAGAR SE COMPLETA LA TRANSACCION

carrito 1:53 P. M. ✓

Arreglo Floral: BOUQUET Cantidad: 1 Precio: RD\$800
Arreglo Floral: AMOROSSA Cantidad: 1 Precio: RD\$500 1:53 P. M.

----- 1:53 P. M.

Total: RD\$1300 1:53 P. M.

Puedes hacer el pedido ahora o seguir añadiendo productos 1:53 P. M.

¿Deseas realizar el pedido para recojer? 1:53 P. M.

F Pasar a recojer

¿Deseas realizar el pedido con envío? 1:53 P. M.

F Enviar

Funeraria Alas de Paz
Pagos
DOP1300.00 TEST INVOICE 2:13 P. M.

F Pagar DOP1,300.00

La orden se marcó como pendiente y *se aplicara un cargo adicional por el envio*, tienes 4 horas para cancelarla.
Además se envió un mensaje a tu cuenta de correo y tu whatsapp, asegurate de haberlo configurado correctamente

F Si deseas cancelar el pedido escribe carrito. 2:13 P. M.

Checkout (Test)

X

Funeraria Alas de Paz
Pagos
Funeraria

Arreglo Floral: BOUQUET	DOP800.00
Arreglo Floral: AMOROSSA	DOP500.00
Envio	DOP200.00
Total	DOP1500.00

Payment Method visa *4242

Shipping Address vialle vede, calle 10, sant...

Shipping Method Envio

Name starling

Cancel Pay DOP1500.00

La orden se marcó como pendiente y *se aplicara un cargo adicional por el envio*, tienes 4 horas para cancelarla.
Además se envió un mensaje a tu cuenta de correo y tu whatsapp, asegurate de haberlo configurado correctamente

F Si deseas cancelar el pedido escribe carrito. 2:13 P. M.

You successfully transferred DOP1500.00 to Funeraria for Funeraria Alas de Paz

#=====

REALIZAR EL PEDIDO PARA ENVIO

(SI EL ENVIO VA DIRIJIDO A UNA UBICACIÓN CON MAS DE 10KM, SE ENVIA MENSAJE QUE NO ES POSIBLE REALIZAR EL ENVIO A ESA UBICACION)

¿Deseas realizar el pedido para recojer? 1:53 P. M.

F Pasar a recojer

¿Deseas realizar el pedido con envío? 1:53 P. M.

F Enviar

F No se puede enviar el pedido a tu ubicacion, estas a 12.18 KM de distancia 2:34 P. M.

#=====

ENVIO DE INFORMACION AL CORREO Y AL WHATTSAPP DEL CLIENTE CON INFORMACION DEL PEDIDO

Inboxes > hey > Orden Pedido Floral

S starlin jose vasquez peña

Search...

Orden Pedido Floral
to: <tatielunik@gmail.com> 8 minutes ago

Orden Pedido Floral
to: <tatielunik@gmail.com> 10 minutes ago

Orden Pedido Floral
to: <tatielunik@gmail.com> 12 minutes ago

Orden Pedido Floral
to: <tatielunik@gmail.com> 14 minutes ago

Orden Pedido Floral
to: <tatielunik@gmail.com> 15 minutes ago

Orden Pedido Floral
to: <tatielunik@gmail.com> 38 minutes ago

Orden Pedido Floral
to: <tatielunik@gmail.com> 43 minutes ago

Orden Pedido Floral
to: <tatielunik@gmail.com> an hour ago

Orden Pedido Floral
to: <default@gmamil.com> 2 hours ago

Orden Pedido Floral
From: <funebot21@gmail.com>
To: <tatielunik@gmail.com>
Show Headers

HTML HTML Source Text Raw Spam Analysis Tech Info

La orden ha sido realizada, tienes 4 Horas para cancelarla
Contenido del pedido
Arreglo Floral: BOUQUET Cantidad: 1 Precio: RD\$800
Arreglo Floral: AMOROSA Cantidad: 1 Precio: RD\$500

Subtotal: RD\$1300.0
Envio: RD\$200
Total: RD\$1500.0

+1 (415) 523-8886

HOY

join active-surprise 2:48 p.m. ✓

Twilio Sandbox: You are all set! The sandbox can now send/receive messages from whatsapp:+14155238886. Reply stop to leave the sandbox any time.

2:48 p.m.

1 MENSAJE NO LEÍDO

La orden ha sido realizada, tienes 4 Horas para cancelarla
Contenido del pedido
Arreglo Floral: BOUQUET Cantidad: 1 Precio: RD\$800
Arreglo Floral: AMOROSA Cantidad: 1 Precio: RD\$500

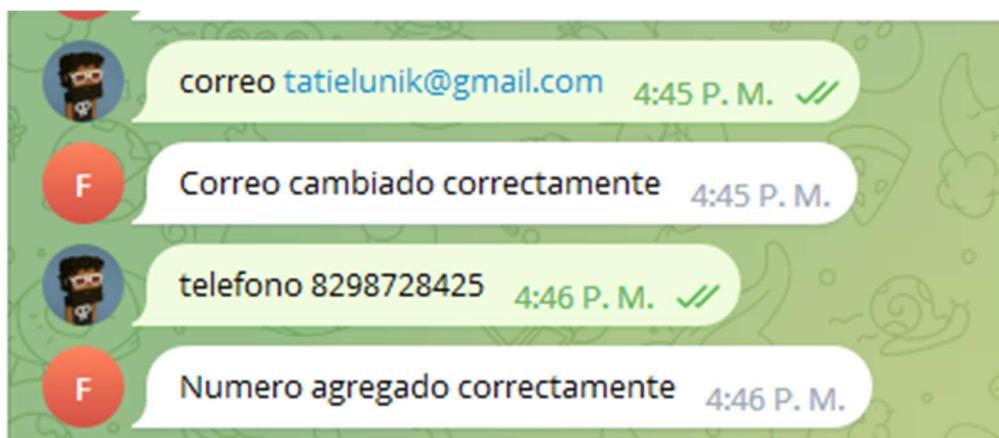
Subtotal: RD\$1300.0
Envio: RD\$200
Total: RD\$1500.0

2:48 p.m.

Escribe un mensaje aquí

#=====

OTROS COMANDOS Y TEXTOS EJECUTADOS



Funciones del Código

Saludos

```
In [5]: 1 from sklearn.feature_extraction.text import TfidfVectorizer
2 from sklearn.metrics.pairwise import cosine_similarity
3 from nltk.corpus import stopwords
4
5 #=====
6 # FUNCION PARA DETERMINAR LA SIMILITUD DEL TEXTO INSERTADO Y EL CORPUS
7 # =====
8
9 def response(user_response):
10    robo_response = ''
11    # Añade al corpus la respuesta de usuario al final
12    sent_tokens.append(user_response)
13    TfIdfVec = TfidfVectorizer(
14        tokenizer=LemNormalize, stop_words=stopwords.words('spanish'))
15    tfidf = TfIdfVec.fit_transform(sent_tokens)
16    # 3 EVALUAR SIMILITUD DE COSENO ENTRE MENSAJE USUARIO (tfidf[-1]) y el CORPUS (tfidf)
17    vals = cosine_similarity(tfidf[-1], tfidf)
18    idx = vals.argsort()[0][-2]
19    flat = vals.flatten()
20    flat.sort()
21    req_tfidf = flat[-2]
22
23    if(req_tfidf == 0):
24        robo_response = robo_response + \
25            "Lo siento, no he entendido tu mensaje. Trata de utilizar otras palabras"
26        return robo_response
27
28    else:
29        robo_response = robo_response+sent_tokens[idx]
30    return robo_response
31
32
33 SALUDOS_INPUTS = ("hola", "buenas", "saludos", "qué tal", "hey", "buenos dias", 'klk')
34 SALUDOS_OUTPUTS = ["Hola", "Hola, ¿Qué tal?", "Hola, ¿Cómo te puedo ayudar?", "Hola, encantado de hablar contigo"]
35
36 def saludos(sentence):
37     for word in sentence.split():
38         if word.lower() in SALUDOS_INPUTS:
39             return random.choice(SALUDOS_OUTPUTS)
40
41 def verificarTexto(msg):
42     respuesta = ''
43     if(saludos(msg) != None):
44         respuesta = saludos(msg)
45
46     else:
47         respuesta = response(msg)
48         sent_tokens.remove(msg)
49
50     return respuesta
```

Importacion XML biblia

```
In [6]: 1 #=====
2 # GENERADOR DE VERSICULOS Y FRASES
3 # =====
4 arbol = ET.parse('Bible_Espanol_oso.xml')
5 def generarVersiculo():
6     root = arbol.getroot()
7
8     partesXML=arbol.findall('BIBLEBOOK')
9     partes = randint(0, len(partesXML)-1)
10
11    capituloXML=root[partes].findall('CHAPTER')
12    capitulo = randint(0, len(capituloXML)-1)
13
14    versiculoXML= root[partes][capitulo].findall('VERS')
15    versiculo=randint(0, len(versiculoXML)-1)
16
17    nombre_=root[partes].attrib['bname']
18    cap_=root[partes][capitulo].attrib['cnumber']
19    ver_=root[partes][capitulo][versiculo].attrib['vnumber']
20    versiculo=root[partes][capitulo][versiculo].text
21    formato="{} {} {}\n{}".format(nombre_,cap_,ver_,versiculo)
22
23    return formato
```

Lista de mensajes

```
In [8]: 1 INICIO_MSG = (
2     "Hola {first_name}!\\n\\n"
3     "Soy el bot de alas de Paz 🌸.\\n\\n"
4     "Envia /ayuda para empezar y ver las instrucciones □."
5 )
6
7 AYUDA_MSG = (
8     "Hola {first_name}!\\n\\n"
9     "Esta es la lista de comandos □:\\n\\n"
10    "**suscribirse**: para recibir versiculos diariosVerse.\\n\\n"
11    "**desuscribirse**: para anular la suscripcionUnsubscribe.\\n\\n"
12    "**flores/productos**: para ver todas las flores disponibles de Alas de Paz 🌸.\\n\\n"
13    "**agregar nombre_producto**: para ver descripcion del arreglo floral y poder ser agregado al carritoCart.\\n\\n"
14    "**correo correo@electrónico**: configurar correoEmail.\\n\\n"
15    "**Telefono**: configurar tu numero telefonicoPhone.\\n\\n"
16    "**carrito**: ver la orden actualCart.\\n\\n"
17    "Si envias la *ubicación* 🏷 esta se guardará para futuros pedidos de flores\\n"
18 )
19
20
```

Funciones Generales

```
In [11]: 1 =====
2 # FUNCION PARA ENVIAR EMAIL
3 # =====
4 #resive el mensaje que se desea enviar y el email del usuario
5 def enviaremail(mensaje, email):
6
7     message = EmailMessage()
8
9     # Configure email headers
10    message['Subject'] = "Orden Pedido Floral"
11    message['From'] = "funebot21@gmail.com"
12    message['To'] = email
13
14    # Set email body text
15    message.set_content(mensaje)
16
17    # Set smtp server and port
18    server = smtplib.SMTP("smtp.mailtrap.io", '2525')
19
20    # Identify this client to the SMTP server
21    server.ehlo()
22
23    # Secure the SMTP connection
24    server.starttls()
25
26    # Login to email account
27    server.login("07dc4f467bde8f", "1634f4420b12ec" )
28
29    # Send email
30    server.send_message(message)
31
32    # Close connection to server
33    server.quit()
34
35 =====
36 # FUNCION PARA ENVIAR MSJ A WHATSAPP
37 # =====
38 def enviarmsjwhatsapp(mensaje, user_tel):
39     #Enviar msj del pedido a whatsapp
40     from twilio.rest import Client
41     account_sid = 'AC588f0503e61c35a576bfe860dba7a3ea'
42     auth_token = 'f594f2c6ab8f11b548dd771468bfc9e1'
43     client = Client(account_sid, auth_token)
44     message = client.messages.create(
45         from_='whatsapp:+14155238886',
46         body=mensaje,
47         to=f'whatsapp:+1{user_tel}'
48     )
49
50 =====
51 # FUNCION PARA CALCULAR LA DISTANCIA ENTRE DOS PUNTOS
52 # =====
53 #recibe las longitudes y latitudes como parametros
54
55 def haversine(lat1, lon1, lat2, lon2):
56     rad=math.pi/180
57     dlat=lat2-lat1
58     dlon=lon2-lon1
59     R=6372.795477598
60     a=(math.sin(rad*dlat/2))**2 + math.cos(rad*lat1)*math.cos(rad*lat2)*(math.sin(rad*dlon/2))**2
61     distancia=2*R*math.asin(math.sqrt(a))
62     return distancia
```

```

1 #=====
2 # FUNCION PARA VER CARRITO
3 # =====
4 def verCarrito(update, message):
5     print("carrito")
6     print(update.message)
7     userTelegram = update.message.from_user
8     username = update.message.chat.username
9
10    # Get Customerid
11    query = f'SELECT cliente_id FROM cliente where cliente_nombreUsuario = "{username}"'
12    user = execute_read_query(connection, query)
13    user_id = user[0][0]
14
15    # 1. Buscar si existe una orden con tipo carrito de este cliente
16    query = f'SELECT * from orden where cliente_id = {user_id} AND orden_estatus = 4'
17    exists = execute_read_query(connection, query)
18    id_orden = 0
19
20    if(len(exists) < 1):
21        query = f'SELECT * from orden where cliente_id = {user_id} AND orden_estatus = 2'
22        existsPending = execute_read_query(connection, query)
23        if(len(existsPending) > 0 ):
24            update.message.reply_text('Tienes una orden pendiente')
25            buttons = [[InlineKeyboardButton("Cancelar orden", callback_data=f'cancelar_orden {existsPending[0][0]}')]]
26            userTelegram.send_message(reply_markup=InlineKeyboardMarkup(buttons), text='¿Deseas cancelarla?')
27        else:
28            update.message.reply_text('No tienes ningún carrito')
29        return
30    else:
31        id_orden = exists[0][0]
32
33    print("verCarrito id_orden id: ",id_orden)
34    queryprice = f'SELECT SUM(oi.precio) FROM orden o INNER JOIN ordenitem oi ON o.orden_id = oi.ordenItem_id WHERE o.orden_id'
35    queryFlorList = f'SELECT flor_nombre, cantidad, flor_precio from (SELECT o.orden_id, oi.flor_id, oi.cantidad FROM orden o'
36    price = execute_read_query(connection, queryprice)
37    floresList = execute_read_query(connection, queryFlorList)
38
39    # Mostrar el listado de artículos del carrito
40    listado = ''
41    for flor in floresList:
42        listado+= f'Arreglo Floral: {flor[0]} Cantidad: {flor[1]} Precio: RD${flor[2]}\n'
43    update.message.reply_text(listado)
44    update.message.reply_text('-----')
45    update.message.reply_text(f'Total: RD${price[0][0]}')
46    update.message.reply_text('Puedes hacer el pedido ahora o seguir añadiendo productos')
47    buttons = [[InlineKeyboardButton("Pasar a recoger", callback_data=f'hacer_pedido_recojer {id_orden} {user_id}')]]
48    userTelegram.send_message(reply_markup=InlineKeyboardMarkup(buttons), text='¿Deseas realizar el pedido para recojer?')
49    buttons = [[InlineKeyboardButton("Enviar", callback_data=f'hacer_pedido_envio {id_orden} {user_id}')]]
50    userTelegram.send_message(reply_markup=InlineKeyboardMarkup(buttons), text='¿Deseas realizar el pedido con envío?')
51
```

```

1 #=====
2 # FUNCION PARA SUSCRIBIRSE
3 # =====
4 def suscribirse(update, message):
5     userTelegram = update.message.from_user
6     username = update.message.chat.username
7
8     query = f'SELECT cliente_suscripto FROM cliente where cliente_nombreUsuario = "{username}"'
9     user_suscripcion = execute_read_query(connection, query)
10    suscripcion = user_suscripcion[0][0]
11    if (suscripcion == 0):
12        #no esta suscripto
13
14        update.message.reply_text('Recibiras versiculos biblicos y frases de forma diaria.')
15        buttons = [[InlineKeyboardButton("Suscribirse", callback_data=f'suscribirse {username}')]]
16        userTelegram.send_message(reply_markup=InlineKeyboardMarkup(buttons), text='¿Deseas suscribirte?')
17    if (suscripcion == 1):
18        update.message.reply_text('Ya estas suscrito.')
19
20
21 #=====
22 # FUNCION PARA DESUSCRIBIRSE
23 # =====
24 def desuscribirse(update, message):
25     userTelegram = update.message.from_user
26     username = update.message.chat.username
27
28     query = f'SELECT cliente_suscripto FROM cliente WHERE cliente_nombreUsuario = "{username}"'
29     user_suscripcion = execute_read_query(connection, query)
30     suscripcion = user_suscripcion[0][0]
31     if (suscripcion == 1):
32         #esta suscripto
33
34         chatt_id=update.message.chat.id
35         update.message.reply_text('Dejaras de recibir versiculos biblicos de forma diaria.')
36         buttons = [[InlineKeyboardButton("Desuscribirse", callback_data=f'des_uscribirse {username} {chatt_id}')]]
37         userTelegram.send_message(reply_markup=InlineKeyboardMarkup(buttons), text='¿Deseas desuscribirte?')
38     if (suscripcion == 0):
39         #no esta suscripto
40         update.message.reply_text('No estas suscrito.')
41
42
43
```

```

1 =====
2 # FUNCION PARA VERIFICAR SUSCRIPCION
3 # =====
4 def verificarSuscripcion(update , context):
5     username = update.message.chat.username
6     queryE = f'SELECT client_id from cliente c WHERE c.cliente_nombreUsuario = "{username}"'
7     data = execute_read_query(connection, queryE)
8     user_ssuscripto = data[0][0]
9
10 =====
11 # FUNCION PARA ENVIAR LOS VERSICULOS
12 # =====
13 def enviarVersiculo(context):
14     try:
15         for user_id, user_data in context.dispatcher.user_data.items():
16             print("user_id:: ",user_id)
17             print("user_data:: ", user_data['username'])
18             if 'username' in user_data.keys():
19                 print("olita if username: ",user_data['username'])
20                 username = user_data['username']
21                 queryE = f'SELECT cliente_suscripto from cliente c WHERE c.cliente_nombreUsuario = "{username}"'
22                 data = execute_read_query(connection, queryE)
23                 user_suscripto = data[0][0]
24                 print("user_suscripto: ",user_suscripto)
25                 if(user_suscripto==1):
26                     id_index = user_data['usser_id']
27                     context.bot.send_message(chat_id=id_index, text=generarVersiculo())
28
29 except TypeError as e:
30     print(e)
31     print("handled successfully")
32
33
34

```

```

1 =====
2 # MENU DE OPCIONES
3 # =====
4 def respuestas(message, update):
5     if message in ('suscribirse'):
6         return suscribirse(update, message)
7
8     if message in ('desuscribirse'):
9         return desuscribirse(update, message)
10    if message in ('flores', 'productos'):
11        return floresLista(update)
12
13    if(message.find('agregar') > -1):
14        return descripcionDeArregloFloral(update, message)
15
16    if(message.find('correo') > -1):
17        return cambiarCorreo(update, message)
18
19    if message in ('carrito'):
20        print(message)
21        return verCarrito(update, message)
22
23
24    if(message.find('telefono') > -1 ):
25        return cambiarTel(update, message)
26
27
28    return verificarTexto(message)

```

```

1 #=====
2     # FUNCION PARA COMANDO INICIAL
3     # ====="""
4 def start_comando(update, context):
5     first_name = update.message.from_user.first_name
6     username = update.message.from_user['username']
7     buttons = [[InlineKeyboardButton("ayuda", callback_data='ayuda')]]
8     chat_id = update.effective_chat.id
9     id = update.effective_chat.id
10    context.bot.send_message(reply_markup=InlineKeyboardMarkup(buttons),
11        chat_id=update.message.chat_id,
12        text=INICIO_MSG.format(first_name=first_name, chat_id=chat_id),parse_mode= 'Markdown'
13    )
14 #=====
15     # FUNCION PARA COMANDO AYUDA
16     # ====="""
17 def ayuda_comando(update: Update, context: CallbackContext) -> None:
18     first_name = update.message.from_user.first_name
19     chat_id = update.effective_chat.id
20
21
22     context.bot.send_message(
23         chat_id=update.message.chat_id,
24         text=AYUDA_MSG.format(first_name=first_name, chat_id=chat_id),parse_mode= 'Markdown'
25     )
26
27 #=====
28     # FUNCION PARA ENVIO DE MENSAJES
29     # ====="""
30 def handle_message(update, context):
31     text = str(update.message.text).lower()
32     user = update.message.from_user
33     ''' Verificar si el usuario está guardado '''
34     verificarUsuario(user)
35     #verificarSuscripcion()
36     response = respuestas(text, update)
37     if(response):
38         update.message.reply_text(response)
39
40 #=====
41     # FUNCION PARA RECIBIR LA LOCALIZACION DEL CHAT
42     # ====="""
43 def LocationMessageEvent(update, context):
44     username = update.message.chat.username
45     latitude = update.message.location.latitude
46     longitude = update.message.location.longitude
47     changeLocation(username, latitude, longitude, update)

```

```

1 #=====
2     # FUNCION PARA GUARDAR USUARIOS
3     # ====="""
4 def verificarUsuario(user):
5     query = f'SELECT * FROM cliente where cliente_nombreUsuario = "{user.username}"'
6     user_exists = execute_read_query(connection, query)
7     if(len(user_exists)):
8         print(f'Existe el usuario: {user.username}')
9     else:
10        print('Guardando el usuario en la DB')
11        query = f'INSERT INTO cliente VALUES(NULL, "{user.first_name}", "default@gmamil.com","", "{user.username}", "",0)'
12        execute_query(connection, query)
13        print(query)
14

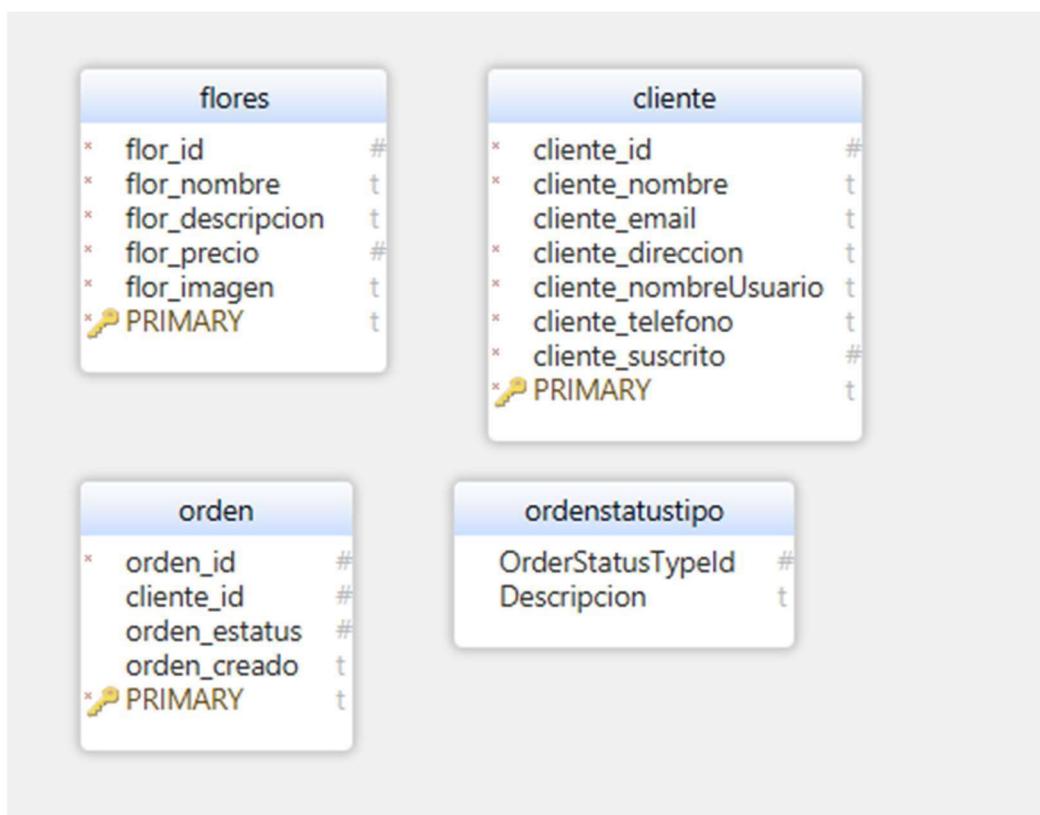
```

Listado de procesos de negocios

ChatBot

- **Suscripción**
 - Registro
- **Desuscripción**
- **Flores**
 - Registro
 - Consulta general
 - Agregar
 - Nombre
 - Pagar
 - Pasar a recoger
 - Con envío
- **Correo**
 - Registro
- **Teléfono**
 - Registro
- **Ubicación**

Modelo de base de datos



Manual de usuario Requisitos:

- Editor de código
- Cuenta en telegram
- MySQL Server
- Pytohn 3.9.6.0 o
- Paquetes:
 - datetime,
 - lxml
 - beautifulsoup4
 - mysqlclient
 - tornado==6.1
 - python-telegram-bot
 - nltk
 - smtplib
 - U scikit-learn scipy matplotlib
 - pywhatkit
 - twilio

Pasos para correr el proyecto

1. Obtén todos los archivos del proyecto de la plataforma nube utesa, abre la carpeta en un editor de código.
 - a. Necesitarás las variables de entorno.
 - b. Instalar todos los paquetes y software.
 - c. Clonar la base de datos.
 - d. Cambiar el string de conexión a la base de datos por el de su equipo.
2. Corre el proyecto utilizando el bot de Telegram.
 - a. Desde la terminal accede a la carpeta **telegram**.
 - b. Ejecutar el archivo **main.py**
 - c. Dirigirse al bot por medio del link (https://t.me/fune_bot) o el usuario del bot (@**fune_bot**)