

**Manual Técnico**

***Chatbot Comercial para messenger***

***Fecha: 09/03/2023***

***Elaborado por: Cristobal Valencia Hernandez***

**Contenido**  Pag.

Presentación ­\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 3

Resumen \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 3

Objetivo \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 3

Aspectos técnicos \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 4

Funcionalidad del proyecto \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 5

Funcionalidad del proyecto \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 6

Funcionalidad del proyecto \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 7

Resultados del proceso \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 9

Requerimientos de software mínimos \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 9

**Presentación**

El siguiente manual se creó con la finalidad de dar a conocer la información necesaria para realizar la instalación y adaptación del Reconocimiento facial.

Este ofrece la información necesaria de como se realizó el Reconocimiento facial para que se pueda, instalar, modificar en los diferentes clientes del proyecto Temp Tech

**Resumen**

El manual detalla los aspectos técnicos e informáticos del Reconocimiento facial con la finalidad de explicar la estructura del aplicativo al personal que quiera administrarlo, editarlo o configurarlo.

La siguiente guía se encuentra dividida en las herramientas que se usaron para la creación del software con una breve explicación paso a paso, El aplicativo web maneja diferentes funcionalidades el cual requieren de hardware y software el cual se explicará que funcionamiento realiza cada uno de ellos, dando sugerencias para el debido uso del sistema de información.

**Objetivo**

Dar a conocer el uso adecuado del **Reconocimiento facial** en aspectos técnicos de manera descriptiva e ilustrada sobre los componentes y funcionalidades que conforman el buen funcionamiento del sistema de información.

La finalidad de este manual técnico es instruir a la persona que quiera administrar, editar o configurar el **Reconocimiento facial** usando las debidas herramientas.

**Aspectos tecnicos**

**Herramientas utilizadas:**

**Visual Studio Code**

Visual Studio Code (VS Code) es un editor de código fuente desarrollado por Microsoft. Es software libre y multiplataforma, está disponible para Windows, GNU/Linux y macOS. VS Code tiene una buena integración con Git, cuenta con soporte para depuración de código, y dispone de un sinnúmero de extensiones, que básicamente te da la posibilidad de escribir y ejecutar código en cualquier lenguaje de programación.

**Librerias:**

**Express**

Es un framework web de Node.js que proporciona una seri de funciones y herramientas para simplificar la creación de servidores web. Permite la definición de rutas, middleware y vistas, así como la gestión de solicitudes HTTP y respuestas.

**Body-Parser**

Es un middleware que se utiliza para analizar el cuerpo de las solicitudes entrantes en formato JSON, texto plan o URL codificada, y los convierte en objetos de JavaScript que se pueden utilizar en el código de la aplicación.

**I18N**

Es una librería usada para la internacionalización en aplicaciones Node.JS. Permite la traducción de mensajes y textos de una aplicación en diferentes idiomas para adaptarse a la región o preferencias del usuario.

Con esta podemos definir archivos de idiomas que contienen claves y valores, y luego usar estas claves en su código para acceder a los textos traducidos. (Todo esto basado en la configuración regional del usuario)

**Request**

Es una librería popular para realizar solicitudes HTTP en Node.js. Proporciona una forma sencilla de enviar solicitudes HTTP con opciones personalizables, incluyendo configuraciones de tiempo de espera, autenticación y manejo de cookies

**Fetch**

Es una función global de JavaScript que también se puede usar en Node.js a través del paquete ‘Node-fetch’. Permite hacer solicitudes HTTP y obtener una respuesta como objeto Promise, lo que lo hace útil para trabajar con código asíncrono.

**URL**

Es un modulo integrado en Node.js que proporción herramientas para trabajar con URL y sus componentes, como el protocolo, el nombre de dominio, el camino y los parámetros de consulta.

**Tedious**

La librería tedious es una herramienta para trabajar con bases de datos SQL server en Node.js. Proporciona una API para realizar operaciones de base de datos como consultas, actualizaciones y eliminaciones. Esta utiliza el protocolo de comunicación TDS para conectarse y comunicarse con SQL Server.

**Herramientas/webs adicionales:**

**Meta for developers**

Es una pagina de ayuda y documentación proporcionada por Facebook para los desarrolladores que utilizan sus herramientas y API para construir aplicaciones y servicios en la plataforma de Facebook.

En particular, se centra en la información técnica y los metadatos que los desarrolladores pueden utilizar para mejorar la forma en que su contenido se muestra en la plataforma de Facebook.

**Pagina de Facebook**

Esta es usada para la instalación y ejecución del bot, también puedes realizar propuestas de marketing dentro de la misma, y usarla para interacción mas de cerca con tus clientes

**Paso a paso codigo**

**Paso 1:** Realizamos la importación de las librerías ‘express’ y ‘body-parser’



**Paso 2:** Se importan las funciones del archivo ‘Handling.js’



**Paso 3:** Se importa la configuración de traducción de texto desde el archivo ‘i18n.config.js’



**Paso 4:** Se importan las funciones de ‘tempGeneration.js’ y ‘graphApi.js’ para la creación de la pantalla de inicio



**Paso 5:** Se crea el app usando body-parser para que JavaScript pueda entender la sintaxis de las peticiones realizadas



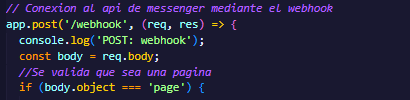
**Paso 6:** Se llama a la función que genera la pantalla de inicio en el chatbot



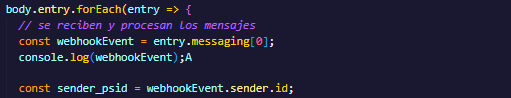
**Paso 6.1:** Se crea la lógica de la función anteriormente mencionada

Imagen que contiene Texto

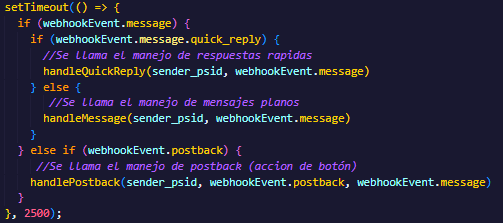
Descripción generada automáticamente

**Paso 7:** Se realiza la conexión al API con endpoint ‘webhook’  


**Paso 7.1:** Se revisa cada una de las peticiones y se registra el identificador del usuario interactuando



**Paso 7.2:** Se verifica el tipo de petición de entrada (Mensaje, postback, o quick reply)



**Paso 7.3:** Se envían las respuestas al servidor (evento recibido en caso de ser exitoso y error de servidor en caso de ser rechazado)  
Texto

Descripción generada automáticamente con confianza media

**Paso 8:** Se crea el endpoint de conexión con meta (Este cumple la función de realizar verificaciones de credencial desde meta for devolpers)

Texto

Descripción generada automáticamente

**Paso 9:** Se crea el listener del servidor al que va a responder nuestras peticiones

Texto

Descripción generada automáticamente

**Nota:** Se instancia el puerto 8080 en caso de que lo queramos subir a un servidor en la nube (Azure, Google cloud, AWS) si no es el caso puede ser cualquier puerto deseado

**Nota importante:** Todo el codigo mencionado anteriormente es codigo que esta siendo ejecutado en el archivo ‘index.js’ es decir que es todo lo que se ejecutara al inicializar el proyecto

**Archivo ‘Handling.js’**

**Paso 1:** Importamos todas las funcionalidades de otros archivos y llamamos la librería tedious para conexión a bases de datos

Texto

Descripción generada automáticamente

**Paso 1.1:** Se crea un objeto para ser rellenado con la información de cada usuario



**Nota:** Esto se realiza con la intención de que no haya problemas cuando varias personas interactúen con el bot

**Paso 2:** Creamos la función ‘Repetir’ cuya función es realizar consultas a la base datos con cada interacción de usuario



**Paso 2.1:** Se ingresa la función de consulta dentro de la función ‘Repetir’ para que esta se ejecute cada vez que haya una interacción



**Paso 3:** Se crea la función de manejo de respuestas ‘HandlePostback’ cuya función es recibir las peticiones que sean realizadas por interacciones con botones



**Paso 3.1:** Se crea un atributo “Flag” dentro de el objeto “UserPSID” con la intención de llevar un control de la conversación



**Paso 3.2:** Se crea la variable a la cual ira asignada cada respuesta del bot al usuario y se llama al api de información de usuario



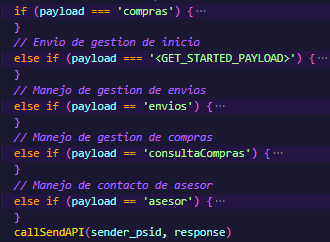
**Nota:** El api de información de usuario recolecta la información asociada a la cuenta de Facebook del usuario en cuestión, para ser mas exactos obtiene el primer nombre (Nombre de usuario) para así poder tener una conversación más dinámica con el usuario

**Paso 3.3:** Se guarda el payload dentro de una variable para tener constancia de cual fue la interacción del usuario

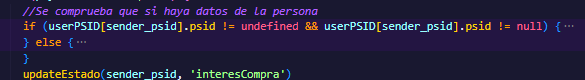


**Nota:** El payload es un tipo de palabra clave la cual sirve para identificar la interacción del usuario

**Paso 3.4:** Se hace una cadena de if’s para entender cuál debería ser la respuesta que del bot según la interacción del usuario



**Paso 3.5 (Compras):** Se verifica si esta persona tiene datos registrados en la base de datos y si los tiene se realiza envio de menú de opciones



**Nota:** También se actualiza el estado con la función “updateEstado” para llevar una constancia del proceso que sigue cada persona con el chat

**Paso 3.6 (Get started):** Se registra si el usuario tiene una primera interacción con el botón de la pantalla de inicio y si es así se envían los mensajes de bienvenida

Texto

Descripción generada automáticamente

**Paso 3.7 (Envíos):** Se registra si la interacción es respecto al área de envíos y se envía un texto de comprobación de datos

Texto

Descripción generada automáticamente

**Paso 3.8 (Consulta Compras):** Se registra si la interacción es respecto a consulta de compras y se envida texto de comprobación de datos para el envio de las ultimas compras de dicho cliente

**Texto

Descripción generada automáticamente**

**Paso 3.9:** Se registra si la interacción es respecto a contacto de asesor y se verifica si dentro de los datos del usuario hay un numero de teléfono registrado sino se le pide el numero al mismo

**Texto

Descripción generada automáticamente**

**Paso 4:** Se crea la función de manejo de mensajes



**Paso 4.1:** Se crea una cadena de if’s para poder ver el seguimiento de la conversación (Esto con el atributo “Flag” del objeto “UserPSID”)

Pantalla de computadora con letras

Descripción generada automáticamente con confianza media

**Paso 4.2:** Se recibe la cedula de el usuario se guarda dentro del objeto y se aumenta el atributo “Flag” para pedir el nombre completo del usuario

Pantalla de un video juego

Descripción generada automáticamente con confianza media

**Paso 4.3:** Se recibe el nombre del usuario y se pide el numero de contacto del mismo

Pantalla de un video juego

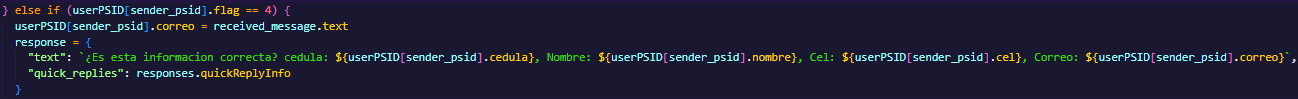
Descripción generada automáticamente con confianza media

**Paso 4.4:** Se recibe el numero del usuario y se pide el correo

Pantalla de un video juego

Descripción generada automáticamente con confianza media

**Paso 4.5:** Se recibe el correo y se realiza la confirmación de cada uno de los datos



**Paso 4.6:** Se recibe la orden de pedido y se verifica si existe dentro de las compras del usuario

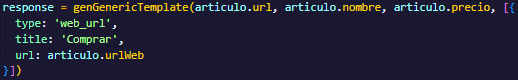
Texto

Descripción generada automáticamente

**Paso 4.7:** Se verifica si existe o no una guía de envíos para dicha persona, si existe se envia su guía con el respectivo link de seguimiento**Texto

Descripción generada automáticamente**

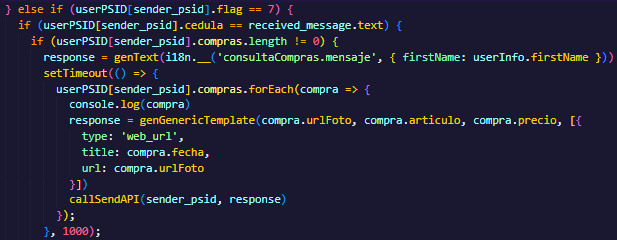
**Paso 4.8:** Se crea la instancia de envio de productos



En este paso también se verifica que si exista el producto que el cliente esta ingresando, de no ser así se le informara al usuario

**Nota:** Este funciona de manera que cuando el usuario escriba el nombre o el id del producto el chatbot responda con la imagen, el precio y el link del producto

**Paso 4.9:** Se genera la instancia de consulta de compras en la cual se verifica la cedula de el usuario y se envían sus ultimas 5 compras (Siendo el caso de que tenga mas de estas)



**Nota:** Este proceso solo se realiza si la persona ya tiene compras dentro de la base de datos, de no ser así se le informara al usuario dicho caso

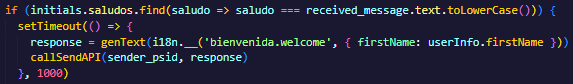
**Paso 4.10:** Se genera la instancia de contactar asesor cuyo fin es obtener el numero de teléfono y notificar al encargado de dicha solicitud de asesoría

Texto

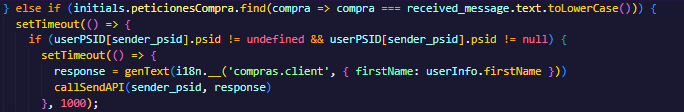
Descripción generada automáticamente

**Nota:** El proceso descrito se realiza con un flujo en power automate cloud

**Paso 4.11:** Hechas todas las comprobaciones anteriores ahora se verifica si la petición de el usuario es un saludo



**Paso 4.12:** Se verifica cuando la petición es respecto al área de compras y se verifica de igual manera si el usuario que hizo la petición es alguien de quien ya tenemos información para dinamizar la conversación



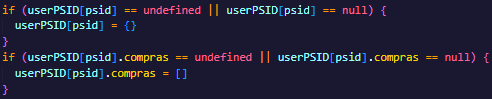
**Nota:** Se verifica si es un cliente de veces anteriores para darle la bienvenida de vuelta o si no lo es para poder tomar datos y llevar constancia de quien interactúa con el chatbot

**Paso 5 (Funciones de SQL):** Se crea la función de consulta para poder conocer la información de los usuarios que llegan al chat

**Texto

Descripción generada automáticamente**

**Paso 5.1:** En caso de que el usuario no haya interactuado nunca con el chatbot se rellena su objeto identificador vacío, dicho proceso se realiza también con el objeto de compras de cada usuario

****

**Paso 5.2:** Se crean las configuraciones de la conexión



**Paso 5.3:** Se realiza la conexión con el servidor

Texto

Descripción generada automáticamente

**Paso 5.4:**  Se realiza la petición de consulta a la base de datos

Texto

Descripción generada automáticamente

**Paso 5.5:** Se reciben los datos y se le asignan a cada usuario

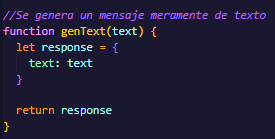
Texto

Descripción generada automáticamente

**Archivo ‘TempGeneration.js’**

La funcionalidad de este archivo es generar plantillas de mensaje con la estructura dada por Meta

**Plantilla de texto:**

****

**Plantilla de tarjetas:**

**Texto

Descripción generada automáticamente**

**Plantilla de respuestas rápidas:**

**Texto

Descripción generada automáticamente**

**Archivo ‘GraphApi.js’**

La funcionalidad de este archivo es enviar todas las peticiones tanto de mensaje, como de información de usuario a la plataforma de meta

**Peticiones de mensaje:**

**Texto

Descripción generada automáticamente**

**Petición de información de usuario:**

**Pantalla de computadora con letras

Descripción generada automáticamente con confianza media**

**Creación de pantalla de inicio:**

**Texto

Descripción generada automáticamente**

**Creación de menú persistente:**

**Texto

Descripción generada automáticamente**

**Nota importante:** Cualquier cambio que se quiera realizar en el flujo de la conversación se debe realizar en los archivos .JSON los cuales contienen las respuestas predeterminadas del chatbot

Interfaz de usuario gráfica

Descripción generada automáticamente

**Ejemplos:**

Texto

Descripción generada automáticamente

**Resultados del proceso**

**Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente**

**Imagen que contiene Patrón de fondo

Descripción generada automáticamente**

**Requerimientos de software mínimos**

* Visual Studio Code (O el ambiente de código que prefieras)
* Node.js

**Requerimientos de hardware mínimos**

* 4/8 GB Ram mínimo (Esto para la ejecución correcta del reconocimiento)
* Espacio disponible en el disco duro (Esto debido a la ejecución local de la base de datos)
* Conexión a internet

**Requerimientos técnicos mínimos**

* Conocimiento (Medio/Avanzado) en javascript
* Conocimiento medio en manejo de listas
* Conocimiento medio en manejo y creación de base datos

Hacemos **TECH** a las empresas

Transformar Evolucionar Conectar Humanizar