MEDAC
Instituto Oficial de Formación Profesional

Manual bot telebram

Requisitos previos

Node.js instalado en tu sistema.

Un token de bot de Telegram obtenido de BotFather.

Pasos para utilizar el módulo:

1. Configuración inicial:

Instala las dependencias necesarias ejecutando npm install node-telegram-bot-api axios en tu terminal.

Obtén el token de tu bot de Telegram de BotFather y reemplázalo en el código.

2. Entendiendo la estructura del código:

El código está dividido en dos partes principales: la configuración del bot y los manejadores de eventos.

La configuración del bot incluye la creación de un nuevo bot utilizando el token y habilitando el polling para recibir actualizaciones de mensajes.

```
/**
 * Módulo para un bot de Telegram que proporciona recetas de cocina.
 * @module TelegramBot
 */

const TelegramBot = require('node-telegram-bot-api');
const axios = require('axios');

// Token de tu bot obtenido de BotFather
const token = '6436020466:AAGTR1mfWq1Ln7pqL2ASWLrQRLU1wX8CVYs';

// Crear un nuevo bot
const bot = new TelegramBot(token, { polling: true });
```

Los manejadores de eventos están definidos para manejar comandos específicos de Telegram, como /start y /receta.

```
bot.onText(/\/start/, (msg) => {
    const options = {
        reply_markup: JSON.stringify({
            keyboard: [D
                ['/receta Potato Salad'],
                ['/receta Bread omelette'],
                ['/receta Blini Pancakes']
            ]
        })
    };
    bot.sendMessage(msg.chat.id, '¡Hola! Soy un bot que te proporcionará recetas de cocina. Elige una opción:', options);
});

/**
 * Maneja las consultas de recetas.
 * Consulta la API de TheMealDB y muestra la receta del plato especificado.
 * @param {Object} msg - El objeto mensaje recibido.
 * @param {Array} match - Array con la cadena de texto que coincide con el patrón.
 */
bot.onText(/\/receta (.+)/, async (msg, match) => {
    const dish = match[1];
    try {
        const response = await axios.get(`https://www.themealdb.com/api/json/v1/1/search.php?s=${dish}`);
        const recipe = response.data.meals[0];
        const ingredients = [];
        for (let i = 1; i <= 20; i++) {
            const ingredient = recipe[`strIngredient${i}`];
            if (ingredient) {
                const measure = recipe[`strMeasure${i}`];
                ingredients.push(`${ingredient} (${measure})`);
            } else {
                break;
            }
        }
        const instructions = recipe.strInstructions;
        bot.sendMessage(msg.chat.id, `Ingredientes para ${dish}:\n${ingredients.join('\n')}\n\nInstrucciones:\n${instructions}`);
    } catch (error) {
        bot.sendMessage(msg.chat.id, 'Lo siento, no pude encontrar una receta para ese plato.');
    }
});
```

3. Comandos disponibles:

/start: Este comando muestra un mensaje de bienvenida y opciones de recetas utilizando un teclado personalizado.

```
bot.onText(/\/start/, (msg) => {
    const options = {
        reply_markup: JSON.stringify({
            keyboard: [D
                ['/receta Potato Salad'],
                ['/receta Bread omelette'],
                ['/receta Blini Pancakes']
            ]
        })
    };
    bot.sendMessage(msg.chat.id, '¡Hola! Soy un bot que te proporcionará recetas de cocina. Elige una opción:', options);
});
```

/receta <nombre_plato>: Este comando consulta la API de TheMealDB para obtener la receta del plato especificado y muestra los ingredientes y las instrucciones de preparación.

```
bot.onText(/\/receta (.+)/, async (msg, match) => {
    const dish = match[1];
    try {
        const response = await axios.get(`https://www.themealdb.com/api/json/v1/1/search.php?s=${dish}`);
        const recipe = response.data.meals[0];
        const ingredients = [];
        for (let i = 1; i <= 20; i++) {
            const ingredient = recipe[`strIngredient${i}`];
            if (ingredient) {
                const measure = recipe[`strMeasure${i}`];
                ingredients.push(`${ingredient} (${measure})`);
            } else {
                break;
            }
        }
        const instructions = recipe.strInstructions;
        bot.sendMessage(msg.chat.id, `Ingredientes para ${dish}:\n${ingredients.join('\n')}\n\nInstrucciones:\n${instructions}`);
    } catch (error) {
        bot.sendMessage(msg.chat.id, 'Lo siento, no pude encontrar una receta para ese plato.');
    }
});
```

4. Personalización:

Puedes personalizar las opciones de recetas en el comando /start modificando el teclado de opciones.

```
const options = {
    reply_markup: JSON.stringify({
        keyboard: [D
            ['/receta Potato Salad'],
            ['/receta Bread omelette'],
            ['/receta Blini Pancakes']
        ]
    })
};
```

Puedes mejorar la lógica de búsqueda y presentación de recetas en el comando /receta.

5. Ejecución del bot:

Ejecuta el bot ejecutando node <nombre_archivo.js> en tu terminal, donde <nombre_archivo.js> es el nombre de tu archivo que contiene el código del bot.

Consideraciones adicionales:

Asegúrate de mantener el token del bot de Telegram seguro y no compartirlo con nadie más.

Puedes agregar más funcionalidades y mejorar el bot según tus necesidades, como agregar manejo de errores más robusto, implementar comandos adicionales, o integrar otras APIs de recetas para obtener una variedad más amplia de resultados.

¡Espero que este manual te sea útil para comprender y utilizar el módulo TelegramBot para proporcionar recetas de cocina! Si tienes alguna pregunta adicional o necesitas más ayuda, no dudes en preguntar.

6.Comprobar el funcionamiento del bot

Para poder ver como funciona el bot has de buscar en telegram el bot:

RamirezBot

De @: @@nosenombrebot

Y el enlace a github es: https://github.com/Croniclus/RAMIREZBOT.git