Telegram AI Chatbot Documentation

Overview:

This project implements a Telegram AI chatbot using Telethon, MongoDB, Google Gemini API, and Google Translate. The bot supports user registration, AI-powered chat responses, image analysis, web search, and auto-translation.

Technologies Used:

- **Python**: Main programming language to create the bot
- **Telethon**: Telegram Bot API client
- **MongoDB**: Database for user and chat history storage
- **Google Generative AI (Gemini API)**: For chatbot responses and image analysis
- **Google Translate**: For multilingual support
- **Requests**: For web search queries
- **Asyncio**: For handling asynchronous tasks

Features & Functionality:

1. User Registration:

- The bot registers users when they send the /start command.
- If a new user joins, their details (chat ID, username, first name) are stored in MongoDB.
- Users are prompted to share their phone number, which is saved in the database.

2. AI-Powered Chat (Gemini API):

- The bot detects the user's language and translates it into English for processing.
- It queries the Gemini AI model for a response and translates it back into the user's language before responding.
- Chat history is stored in MongoDB.
- If the user doesn't respond within 5 minutes, the bot sends a follow-up message.

3. Image & File Analysis:

- The bot accepts images and uses gemini-pro-vision to generate a description.
- The description is sent back to the user.
- Error handling ensures failed image downloads do not crash the bot.

4. Web Search:

- Users can search the web using /search <query>.
- The bot fetches results from Google Custom Search API and returns the top 5 links.

5. Translation Feature:

- Users can translate text to English using /translate <text>.
- The bot processes the translation using Gemini AI.

Code Breakdown:

1. Initialization:

- Environment variables (API keys, database URIs) are loaded using dotenv.
- Telethon initializes the bot.
- MongoDB connections are established.
- Gemini AI API is configured.
- Google Translate API is set up.

2. User Registration:

- /start: Registers new users and requests their phone number.
- Phone number is stored upon sharing.

3. Chat Processing:

- Detects the user's language.
- Uses gemini-pro for generating responses.
- Translates the response back into the user’s language.
- Stores chat history in MongoDB.
- Triggers a follow-up message if the user remains inactive.

4. Image & File Analysis:

- Downloads images sent by users.
- Uses gemini-pro-vision to generate a description.
- Handles errors and sends an appropriate response if the analysis fails.

5. Web Search:

- /search <query> triggers a Google Custom Search API request.
- Returns the top 5 results with titles and links.

6. Translation:

- /translate <text> triggers text translation using Gemini AI.

Error Handling & Debugging:

- Checks if API responses are valid before processing.
- Logs errors to help with debugging.
- Provides user-friendly error messages in case of failures.

Running the Bot:

1. Set up environment variables (.env file required).
2. Install dependencies using:

```
pip install -r requirements.txt
```

3. Run the bot:

```
python bot.py
```

Future Enhancements

- Improve multilingual support.
- Implement advanced image/file analysis for large files.
- Add support for video and audio processing.
- Implement a more advanced web search engine.