✦ Lumin - Advanced Gemini AI Discord Bot

Lumin is a professional, multimodal Discord bot powered by Google's **Gemini 2.5** and **Imagen 3** models. It features long-term vector memory (RAG), sophisticated file handling, customizable personalities, and a robust settings dashboard.
Built with discord.js, @google/genai, and MongoDB.

✨ Key Features

🧠 AI Capabilities

- **State-of-the-Art Models:** Supports gemini-2.5-flash, gemini-2.5-pro, and gemini-2.5-flash-lite.
- **Multimodal Understanding:** Capable of processing:
  - **Images:** (PNG, JPG, WebP)
  - **Audio:** (MP3, WAV, OGG, etc.) - *Automatically converts formats like OGG to MP3.*
  - **Video:** (MP4, MOV, WebM) - *Automatically converts GIFs and animated stickers to MP4.*
  - **Documents:** (PDF, TXT, Code files, CSV, etc.) - *Extracts text automatically.*
- **Image Generation:** Uses imagen-3.0-fast-generate-001 via the /imagine command.
- **Code Execution:** Can generate and execute Python code internally.

💾 Memory & Context (RAG)

- **Vector Database:** Uses MongoDB to store vector embeddings of conversations.
- **Context Retrieval:** Automatically retrieves relevant past messages based on the current conversation topic (Retrieval-Augmented Generation).
- **Summarization:** Automatically compresses long conversation histories to maintain context without hitting token limits.

⚙️ Customization & Control

- **Dual Settings System:** Separate settings for **Users** (Global) and **Servers** (Guild-specific).
  - **Server Overrides:** Admins can enforce server-wide settings (e.g., enforcing a specific model or disabling embedded responses).
- **Personalities:** Users and Servers can set custom system instructions (Personalities) for the bot.
- **Response Styles:** Choose between "Normal" (text) or "Embedded" (rich formatting with metadata) responses.
- **Continuous Reply Mode:** The bot can engage in conversation without being mentioned every time.

🛠️ Technical Features

- **Poll Analysis:** Can read and analyze Discord Poll results.
- **Rate Limiting:** Built-in protection against spam and image generation abuse.
- **Thread Safety:** Uses Mutex locks to prevent race conditions during chat history updates.
- **Auto-Cleanup:** Automatically manages temporary files to keep the host system clean.

📋 Prerequisites

Before running the bot, ensure you have the following installed:

1. **Node.js** (v18.0.0 or higher)
2. **MongoDB** (Local instance or Atlas URI)
3. **FFmpeg** (Required for media conversion, e.g., converting GIFs/Stickers to Video for Gemini)

🚀 Installation

1. **Clone the repository:**
   ```
   git clone
   [https://github.com/yourusername/lumin-bot.git](https://github.com/yourusername/lumin-bot.git)
   cd lumin-bot
   ```

2. **Install dependencies:**
   ```
   npm install
   ```

3. **Configure Environment Variables:** Create a .env file in the root directory:
   ```
   # Discord Configuration
   DISCORD_BOT_TOKEN=your_discord_bot_token_here

   # Google AI Configuration
   GOOGLE_API_KEY=your_google_gemini_api_key_here

   # Database Configuration
   MONGODB_URI=mongodb://localhost:27017/gemini-discord-bot

   # Optional
   PORT=3000
   ```

4. **Start the bot:**
   ```
   # For development (with auto-reload)
   npm run dev

   # For production
   npm start
   ```

🎮 Commands

Command	Description
/settings	Opens the interactive dashboard to configure User or Server settings.
/search	Perform a search or query with the AI. Supports text prompts and file attachments.
/imagine	Generate images using Google's Imagen 3 model.

💬 Usage Guide

Interaction Modes

1. **Mention:** Simply tag @Lumin (or your bot's name) to chat.
2. **Reply:** Reply to a bot's message to continue the conversation.
3. **Continuous Mode:** Enable this in /settings to have the bot respond to every message in a channel (or DM) without mentions.

File Attachments

You can upload almost any file type.

- **Code/Text:** Upload .js, .py, .txt, etc., and ask the bot to analyze or debug it.
- **PDFs:** Ask for summaries of uploaded PDF documents.
- **Videos/GIFs:** Ask the bot to describe what is happening in the video.

Managing Context

- **Time Awareness:** The bot knows how much time has passed between messages.
- **User Info:** It knows who is speaking (Username/Display Name).
- **Downloads:** You can download your conversation history via the /settings menu.

🏗️ Project Structure

- **index.js**: Entry point. Handles Discord events, message processing, file conversion, and response streaming.
- **botManager.js**: Singleton manager for the Discord Client, Gemini Instance, and global state/settings management.
- **memorySystem.js**: Handles Embedding generation, Vector search, and RAG logic using MongoDB.
- **database.js**: MongoDB wrapper for saving settings, history, and memory entries.
- **config.js**: Static configuration (models, colors, default prompts).

📜 License

Developed by unmuted / _imgeno