■ Gmail AI Assistant

Complete Working Guide

This guide walks you through setting up, configuring, and using the Gmail AI Assistant — a Streamlit web application that lets you control your Gmail inbox using natural language voice or text commands, powered by Google Gemini AI.

Component	Technology
Frontend / UI	Streamlit
AI Engine	Google Gemini 2.5 Flash
Email Backend	Gmail API (OAuth2)
Voice Input	Web Speech API (browser)
Config	Python dotenv (.env file)

1. Prerequisites

Before installation, make sure you have the following:

Requirement	Details
Python	3.9 or higher
Google Account	Gmail-enabled account
Google Cloud Project	Gmail API enabled
Gemini API Key	From Google AI Studio (aistudio.google.com)
Browser	Chrome or Edge for voice commands (Firefox not supported)

2. Installation

Step 1 — Install Python dependencies

A requirements.txt file is included with the project. Install all dependencies with:

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

Package	Min Version	Purpose
streamlit	≥1.39.0	Web UI framework
google-api-python-client	≥2.111.0	Gmail API client
google-auth	≥2.35.0	OAuth2 authentication
google-auth-oauthlib	≥1.2.0	OAuth2 flow handling
google-generativeai	≥0.8.0	Gemini AI API
python-dotenv	≥1.0.0	.env file support
rich	≥13.0.0	Enhanced error display

Step 2 — Set up Google Cloud OAuth Credentials

1. Go to console.cloud.google.com and create (or select) a project — note your Project ID.

2. Navigate to APIs & Services → Library and enable the Gmail API.

3. Go to APIs & Services → Credentials → Create Credentials → OAuth 2.0 Client ID.

4. Select Desktop App as the application type.

5. Download the JSON file and rename it to credentials.json.

6. Place credentials.json in the project root directory.

The downloaded credentials.json will have this structure:

```
{ "installed": { "client_id": "YOUR_CLIENT_ID.apps.googleusercontent.com",
"project_id": "your-project-id", "auth_uri":
"https://accounts.google.com/o/oauth2/auth", "token_uri":
"https://oauth2.googleapis.com/token", "client_secret": "YOUR_CLIENT_SECRET",
"redirect_uris": ["http://localhost"] } }
```

■■ credentials.json contains your OAuth client secret. NEVER commit it to version control. Add it to .gitignore immediately. If exposed, revoke and regenerate via Google Cloud Console.

Step 3 — Configure Environment Variables

Create a .env file in the project root with the following content:

```
GEMINI_API_KEY=your_gemini_api_key_here
```

■■ Get your free Gemini API key at aistudio.google.com/apikey.

Step 4 — Create a .gitignore File

Protect sensitive files from being accidentally committed:

```
credentials.json token.pickle .env __pycache__/ *.pyc
```

Step 5 — Verify Project File Structure

Your project folder should look like this:

```
gmail-ai-assistant/ ■■■ app.py ■■■ ai_assistant.py ■■■ mail_service.py ■■■ test_models.py ■■■ credentials.json ← you provide this ■■■ token.pickle ← auto-generated on first run ■■■ .env ← you create this
```

3. Running the Application

Launch the app from your terminal:

```
streamlit run app.py
```

■■ On the very first run, a browser window will open automatically for Google OAuth2 authentication. Sign in with your Google account and grant Gmail access. A token.pickle file will be saved — subsequent runs will not require re-authentication.

The app will be available at http://localhost:8501 in your browser.

4. Using the Application

4.1 Interface Overview

The app has two main areas: a sidebar on the left for commands and navigation, and a main content panel on the right for the inbox, email detail, or compose view.

4.2 Voice Commands

Click the ■ Click & Speak button in the sidebar, then speak your command clearly. The app will recognize your speech, parse it with Gemini AI, and execute the action instantly. Voice input requires Chrome or Edge browser (Firefox is not supported).

4.3 Text Commands

Type a natural language command in the text box labeled ■ Or type command: and click the ■■ Execute button.

4.4 Command Reference

Command Example	Action Taken
"show unread emails"	Filters inbox to unread messages only
"show emails from alice"	Filters inbox by sender name or address
"show emails from last week"	Filters by date (last 7 days)
"show emails from last 10 days"	Filters by date (last 10 days)
"show emails about invoice"	Keyword search across inbox
"compose email"	Opens blank compose window
"compose email to bob@example.com"	Opens compose with To field prefilled
"open email from support"	Opens latest email from that sender
"reply"	Prepares reply to the currently open email

4.5 Manual Navigation

Use the sidebar navigation buttons to switch views directly: ■ Inbox, ✉■ Compose, ■ Sent, and ■ Refresh.

4.6 Command History

The last 10 executed commands are shown in the ■ Command History expander in the sidebar. Click Clear History to reset the log.

5. How It Works — Architecture

Step	Module	Description
1. Input	app.py	Voice (Web Speech API) or typed text command captured
2. AI Parse	ai_assistant.py	parse_command() sends command to Gemini; returns JSON action
3. Execute	app.py	execute_action_with_feedback() reads action, updates session state
4. Gmail API	mail_service.py	list_emails(), get_email_detail(), send_email() called as needed
5. UI Update	app.py	st.rerun() refreshes the Streamlit UI with new state

Supported AI Actions (from Gemini)

Action	Parameters	Description
compose	to, subject, body	Opens compose view with prefilled fields
filter_inbox	unread, sender, keyword, date_range	Queries Gmail and refreshes inbox list
open_email	sender	Finds latest email from sender and opens it
reply	(none)	Reads current email and opens reply compose view
unknown	(none)	Command not recognized; shows info message

6. Security & Permissions

• The app uses the gmail.modify OAuth2 scope, which grants read and send access but does not allow permanent email deletion.

• OAuth2 credentials are stored locally in token.pickle — this file is sensitive and should not be shared.

• Your Gemini API key is read from the .env file and is never transmitted to Google's Gmail servers.

• No email content is stored by the app beyond what is displayed in the current session.

• Add credentials.json, token.pickle, and .env to your .gitignore to avoid accidental exposure.

■■ To revoke access at any time, go to myaccount.google.com/permissions and remove the app.

7. Troubleshooting

Problem	Solution
GEMINI_API_KEY not found	Check that .env exists in the project root with the correct key name and value
OAuth popup does not open	Delete token.pickle and re-run the app to trigger re-authentication
Voice button not responding	Use Chrome or Edge. Check that microphone permissions are granted for localhost
HttpError 403 from Gmail	Ensure the Gmail API is enabled in your Google Cloud project console
Gemini model not found error	Run test_models.py to list available models and update ai_assistant.py accordingly
App shows blank inbox	Click ■ Refresh or check Gmail API quota limits in Cloud Console
token.pickle authentication fails	Delete token.pickle and re-authenticate; the token may have expired

8. Testing Gemini Models

If you encounter errors related to the Gemini model name, run the included test_models.py script to see all models available to your API key:

```
python test_models.py
```

Update the model name in ai_assistant.py if needed. The current default is gemini-2.5-flash.