

---

## Computer Voice Assistant - VS Code Setup Guide

**Projekt:** Computer Voice Assi

**Datum:** 06. Dezember 2025

**Erstellt von:** Manus AI

**Seite:** {page}

---



# Voice Assistant in VS Code starten - Schritt für Schritt

---

Hallo Chef! Hier ist deine Anleitung, um den Voice Assistant direkt in Visual Studio Code zu starten und zu testen. Folge einfach diesen Schritten.

---



## Vorbereitung: Was du brauchst

---

1. **Visual Studio Code:** [Hier herunterladen](#)
  2. **Python:** Version 3.9, 3.10 oder 3.11. [Hier herunterladen](#)
  3. **Git:** [Hier herunterladen](#)
  4. **GitHub Account:** Du hast bereits einen.
  5. **Picovoice Account:** Kostenlos erstellen auf [Picovoice Console](#)
- 

## Schritt 1: Projekt in VS Code öffnen

---

Zuerst klonen wir das Projekt von GitHub direkt in VS Code.

1. **Öffne VS Code.**
2. Öffne die **Befehlspalette** mit `Strg+Shift+P` (oder `Cmd+Shift+P` auf Mac).
3. Tippe `Git: Clone` und drücke Enter.

4. Füge die **Repository URL** ein:

```
https://github.com/KoMMb0t/Computer-Voice-Assi.git
```

5. Wähle einen **lokalen Ordner**, in dem du das Projekt speichern möchtest (z.B. `C:\Users\DeinName\Documents\GitHub` ).

6. Nach dem Klonen fragt VS Code: **“Möchten Sie das geklonte Repository öffnen?”** Klicke auf **“Öffnen”** .

**Ergebnis:** Das Projekt ist jetzt in VS Code geöffnet und du siehst alle Dateien im Explorer auf der linken Seite.



VS Code Explorer

<– Platzhalter für Screenshot

---

## Schritt 2: Python-Umgebung einrichten

---

Wir erstellen eine virtuelle Umgebung, um die Projekt-Abhängigkeiten sauber zu halten.

1. **Öffne ein neues Terminal** in VS Code mit `Strg+Shift+Ö` (oder `Terminal > New Terminal` ).

2. **Erstelle eine virtuelle Umgebung:**

```
python -m venv .venv
```

3. **Aktiviere die virtuelle Umgebung:**

- **Windows (PowerShell):**

```
.\.venv\Scripts\Activate.ps1
```

- **Windows (CMD):**

```
.venv\Scripts\activate.bat
```

- **macOS/Linux:**

```
source .venv/bin/activate
```

**Wichtig:** Dein Terminal sollte jetzt `(.venv)` am Anfang der Zeile anzeigen.

#### 4. Python-Interpreter in VS Code auswählen:

- Drücke `Strg+Shift+P` und tippe `Python: Select Interpreter`.
- Wähle den Interpreter aus, der `' .venv '` im Pfad hat. Er wird meist als **‘Recommended’** angezeigt.

---

## Schritt 3: Abhängigkeiten installieren

---

Jetzt installieren wir alle Python-Pakete, die das Projekt benötigt.

1. **Stelle sicher, dass deine virtuelle Umgebung aktiv ist** (siehe `(.venv)` im Terminal).
2. **Installiere die Pakete** mit diesem Befehl:

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

*Hinweis: Eine `requirements.txt` ist im Repo. Falls nicht, hier die manuelle Installation:*

```
pip install pvporcupine vosk sounddevice numpy openai edge-tts
```

3. **Warte, bis die Installation abgeschlossen ist.** Das kann ein paar Minuten dauern.
- 

## Schritt 4: Konfiguration anpassen ( `config.ini` )

---

Bevor wir starten, musst du deine persönlichen API-Keys eintragen.

1. Öffne die Datei `config.ini` im VS Code Explorer.

### 2. Trage deine API-Keys ein:

```
[WakeWord]
# Dein Picovoice Access Key von der Picovoice Console
access_key = "DEIN_PICOVOICE_ACCESS_KEY"
# Der Pfad zum "Computer" Wake-Word Modell (.ppn Datei)
# Lade es von der Picovoice Console herunter und platziere es im
Projektordner
keyword_path = "computer_windows.ppn"

[LLM]
# Dein OpenAI API Key
api_key = "DEIN_OPENAI_API_KEY"
```

### 3. Wake-Word Modell herunterladen:

- Gehe zur [Picovoice Console](#).
  - Gehe zum Tab **“Porcupine”** .
  - Erstelle das Wake-Word **“Computer”** .
  - Wähle als Plattform **“Windows”** .
  - Lade die `.ppn` -Datei herunter.
  - Benenne sie um (z.B. `computer_windows.ppn` ) und kopiere sie in den Hauptordner deines Projekts.
-

## Schritt 5: Voice Assistant starten & testen

---

Jetzt ist alles bereit für den ersten Start!

1. **Öffne die Haupt-Datei:** `15_voice_assistant_configurable.py` .
2. **Starte das Skript** mit dem “Play” -Button oben rechts in VS Code oder drücke `F5` .

 VS Code Run Button

*<- Platzhalter für Screenshot*

3. **Beobachte das Terminal.** Du solltest folgende Ausgaben sehen:

```
✓ Porcupine initialisiert.  
✓ Vosk Model geladen.  
✓ Edge-TTS initialisiert.
```

```
🎤 Höre auf 'Computer'...
```

4. **Sage laut und deutlich: “Computer”** .

- Der Assistent sollte mit “Ja?” oder einem Sound antworten.
- Im Terminal siehst du: `✓ Wake-Word 'Computer' erkannt!`

5. **Gib einen Befehl:**

- Sage: “Öffne YouTube” .
- Der Browser sollte sich öffnen.

6. **Teste die LLM-Integration:**

- Sage: “Computer, wie hoch ist der Mount Everest?”
  - Der Assistent sollte die Antwort von ChatGPT vorlesen.
-

# Troubleshooting: Was tun, wenn es nicht klappt?

---

- **Fehler:** `pyporcupine.PorcupineActivationError`
  - **Lösung:** Dein `access_key` in `config.ini` ist falsch. Kopiere ihn nochmal von der Picovoice Console.
- **Fehler:** `PorcupineInvalidArgumentError`
  - **Lösung:** Der Pfad zur `.ppn`-Datei in `config.ini` ist falsch oder die Datei fehlt. Überprüfe den Dateinamen und den Speicherort.
- **Fehler:** `sounddevice.PortAudioError`
  - **Lösung:** Dein Mikrofon wird nicht erkannt. Überprüfe die Windows-Sound-Einstellungen. Manchmal hilft es, das Standard-Mikrofon neu festzulegen.
- **Wake-Word wird nicht erkannt:**
  - **Lösung 1:** Sprich lauter und deutlicher.
  - **Lösung 2:** Erhöhe die `sensitivity` in `config.ini` (z.B. auf `0.7`).
  - **Lösung 3:** Teste dein Mikrofon mit dem Skript `03_record_wake_word.py` ( `--mode test` ).
- **Befehle werden nicht verstanden:**
  - **Lösung:** Sprich deutlicher. Überprüfe die Vosk-Modell-Sprache (sollte `de` sein).

---

## **Geschafft!**

---

Wenn du diese Schritte befolgt hast, sollte dein Computer Voice Assistant jetzt in VS Code laufen.

Viel Spaß beim Testen und Weiterentwickeln! 💪

---