**Handbuch**

**Adaptable Console Voice Assistent**

**"Willkommen beim Adaptable Console Voice Assistent – Dein kreativer Künstler für den ultimativen Digital-Lifestyle!"** 🚀

Bereit für ein atemberaubendes Feuerwerk der Funktionen? Der Adaptable Console Voice Assistent ist nicht nur ein Sprachassistent, sondern ein Regisseur, der ein beeindruckendes Zusammenspiel verschiedener Funktionen orchestriert – alles mit einem einzigen Sprachbefehl!

**Kreative Kombinationen – Ein Feuerwerk der Möglichkeiten:** Dein Assistent vereint die Vielseitigkeit der App-Interaktionen, den natürlichen Chat-Modus, die Entspannung auf Knopfdruck und die intelligenten Kombinationen zu einem atemberaubenden Gesamtkunstwerk. Sag einfach "Feuer frei", und erlebe:

1. 🎵 **Musikalische Magie:** Die Playlist deiner Stimmung, gesteuert von den Worten "musik playlist".
2. 📖 **Geschichten-Extravaganza:** Ein fesselnder Chat, wenn du sagst "unterhalten".
3. 🌌 **Entspannungsoase:** Lagerfeuerromantik, meditative Klänge durch die perfekte Playlist mit einem einfachen "entspannen".
4. 🔥 **Kreativer Zauber:** Deine individuellen Kombinationen, zum Beispiel "feuer" und "musik playlist meditation", für eine Reise durch deine Vorstellungskraft.

**Freiheit der Anpassung – Dein persönliches Feuerwerk:** Dieses Assistenten-Feuerwerk ist keine festgelegte Show, sondern ein persönlicher Ausdruck deiner Bedürfnisse und Ideen. Passe ihn an, lasse ihn wachsen und entfache jedes Mal ein neues Feuerwerk der Kreativität.

**Offline Spracherkennung und Text-to-Speech (TTS) – Sicherheit und Freiheit:** Dieses Feuerwerk der Funktionen leuchtet auch ohne permanente Internetverbindung. Mit erstklassiger offline Spracherkennung und TTS respektiert dein Assistent deine Privatsphäre und gibt dir die Freiheit, dich auszudrücken.

**App Funktion**

Wie alle Funktionen so hat auch die App Funktion 15 frei gestaltbare Presets die von dir kreativ genutzt und gestaltet werden können. Die App Funktion macht sich die Eigenschaft vieler Apps und Websites zu nutze mit Hilfe von Hotkeys und anderen Tasten und Kombinationen bedient werden zu können. So ist es möglich mit fast allen Apps und Websites viele spannende Dinge anzustellen und diese Funktionen per Sprache auszulösen. Lass deiner Kreativität freien Lauf!

**Anleitung zur Konfiguration einer App-Funktion für den **Adaptable-Console-Voice-Assistent**:**

1. **App-Konfiguration:**
   * app = Pfad zu Spotify: Hier wird der Pfad der App festgelegt, für die die Funktion konfiguriert wird. In diesem Beispiel ist es Spotify.
   * wait\_app = 5: Die Wartezeit in Sekunden, die der Assistent auf das Öffnen der Anwendung wartet bevor die weiteren Schritte ablaufen. Hier sind es 5 Sekunden.
   * app\_signal\_words = musica: Die Schlüsselwörter, die den Assistenten dazu auffordern, die App-Funktion auszuführen. Hier lautet das Signalwort "musica". Zu beachten ist, das Umlaute bei den Signalwörtern nicht genutzt werden dürfen, da sie zu Problemen führen. Anstelle von ä einfach a schreiben usw.
   * app\_and\_signal\_words = playlist: Zusätzliche Schlüsselwörter, die mit den App-Signalwörtern kombiniert werden müssen. Hier ist das kombinierte Signalwort zum auslösen der Funktion "musica playlist".
2. **Hotkeys, User Eingaben, Text und Tastenkombinationen:**
   * hotkey\_1 =: Hier wird die erste Tastenkombination festgelegt. In diesem Beispiel ist es leer, was bedeutet, dass keine Tastenkombination für diesen Schritt erforderlich ist. Diese müssen folgendermaßen eingetragen werden z.b. ctrl,alt,F. Leerzeichen dürfen nicht vorkommen.
   * wait\_1\_1 = 0.5: Die Wartezeit nach der Tastenkombination, bevor der nächste Schritt ausgeführt wird.
   * user\_input\_1 = False: Gibt an, ob Benutzereingabe verarbeitet werden sollen. False bedeutet nein. Bei True wird das was vom Benutzer nach den Signalwörtern gesagt wird beispielsweise in ein zuvor mit einem Hotkey aufgerufenes Suchfeld in der App eingetragen.
   * text\_1 =: Hier kann optional ein Text eingegeben werden, der an die App gesendet wird.
   * wait\_1\_2 = 0.5: Wartezeit nach dem Text, bevor der nächste Schritt ausgeführt wird.
   * keys\_1 =: Tasten, die nach dem Text gedrückt werden sollen. Diese müssen folgendermaßen eingetragen werden z.b. "tab", "tab", "tab", "down". Auch Leerzeichen sind zu beachten.
   * wait\_1\_3 = 0.5: Wartezeit nach den Tasten, bevor der nächste Schritt ausgeführt wird.
   * Das ist die Beschreibung eines Schrittes zur Bedienung der App.
   * Es gibt insgesamt sechs Schritte.
   * In jedem Schritt darf nur eine Funktion genutzt werden. Also entweder eine Hotkey Kombination gedrückt werden oder ein User\_input oder ein Text an die App übergeben werden oder die keys Funktion genutzt werden. Diese stellt eine Ausnahme da, da hier einer oder beliebig viele Tasten nach einander gedrückt oder abgearbeitet werden können(Schreibweise beachten).
3. **Hinweis:**
   * Achte darauf, dass die Wartezeiten realistisch gewählt werden, um sicherzustellen, dass die App-Funktion korrekt ausgeführt wird. Sollte ein folgender Schritt nicht erledigt werden liegt es meist an einer zu kurzen Wartezeit.
   * Tastenkombinationen und Benutzereingaben sollten entsprechend den Anforderungen der App korrekt festgelegt werden. Da dies bei jeder App unterschiedlich ist sollte man sich entsprechend informieren oder in der Hilfe der App unter Hotkeys und Tasten nachsehen.
   * Es ist ratsam, die Konfiguration mit einfachen Schritten zu beginnen und sie dann nach Bedarf zu erweitern und anzupassen.
   * Um Mißverständnissen bei der Spracherkennung entgegenzuwirken ist es sinnvoll als User\_Input nur Wörter der gewählten Sprache zu nutzen und deren Erkennung vorher einmal auszuprobieren. z.b. macht es bei Spotify Sinn als Namen für seine Playlist´s bei deutscher Sprache deutsche Begriffe zu nutzen. Will man viele Englischsprachige Künstler wählen können sollte man diese Funktion in englischer Sprache festlegen damit Englischen Namen richtig erkannt und weitergegeben werden.

**Chat Funktion**

**Anleitung zur Konfiguration einer Chat-Funktion für den **Adaptable-Console-Voice-Assistent**:**

1. **Chat-Konfiguration:**
   * active\_chat\_engine = huggingface: Hier wird die Chat-Engine ausgewählt, die für die Konversation verwendet wird. In diesem Beispiel ist es Hugging Face.
   * signal\_words = erzahl: Die Schlüsselwörter, die den Assistenten dazu auffordern, die Chat-Funktion zu starten. Hier lautet das Signalwort "erzahl". Zu beachten ist, das Umlaute bei den Signalwörtern nicht genutzt werden dürfen, da sie zu Problemen führen. Anstelle von ä einfach a schreiben usw.
   * and\_signal\_words = geschichte: Zusätzliche Schlüsselwörter, die mit den Signalwörtern kombiniert werden müssen. Hier ist das kombinierte Signalwort zum auslösen der Funktion "erzahl geschichte".
2. **Benutzereingabe-Konfiguration:**

**Die folgenden Eingaben sind als eine Strukturierungs- bzw. Erstellungshilfe für den an den Chatbot bzw. GPT zu übergebenden Promt anzusehen und enthalten alle zum Zeitpunkt der Erstellung als sinnvoll erachteten Bestandteile eines guten Promts. Sie werden aneinandergereiht an den Chatbot bzw. GPT übergeben und müssen nicht zwangsweise genutzt werden. Es ist auch möglich den ganzen Promt an einer Stelle einzutragen oder nur den User\_input zu nutzen.**

* + user\_input\_enabled = True: Gibt an, ob die Benutzereingabe aktiviert ist und der Text der nach den Signalwörtern gesprochen wird als Aufgabe an den Chatbot bzw. GPT weitergegeben wird.
  + user\_input\_task =: Hier kann der Benutzer die Aufgabe festzulegen die an an den Chatbot bzw. GPT weitergegeben wird. z.b. in diesem Fall erzähl eine Geschichte.
  + user\_input\_context =: Der Benutzer kann hier den Kontext für die Konversation angeben. Um so umfangreicher umso besser die Ergebnisse.
  + user\_input\_example =: Hier kann der Benutzer Beispiele für die Konversation angeben. z.b. eine Geschichte die einen ähnlichen Aufbau hat wie das was man sich wünscht.
  + user\_input\_persona =: Der Benutzer kann hier die Persona für den Chat festlegen. z.b. Märchenerzähler
  + user\_input\_format =: Hier kann der Benutzer das Format für die Chat-Antworten angeben.
  + user\_input\_tone =: Der Ton, den der Assistent in der Konversation verwenden soll. z.b. Jugendsprache, förmlich, usw.

1. **Dialog-Chat-Konfiguration:**

**Während die obere Chat-Funktion eine für sich stehende Anfrage an den Chatbot bzw. GPT stellt ohne das anschließende weitere sich darauf beziehende Nachfragen oder Anmerkungen möglich sind, kann mit der folgenden Einstellung eine Dialog-Funktion mit all den genannten Möglichkeiten initiiert werden. Beenden kann man den Dialog übrigens mit den Signalwort Stop.**

* + dialog\_chat\_enabled = False: Gibt an, ob der Dialog-Chat aktiviert ist. In diesem Beispiel ist er deaktiviert.

1. **Hinweise:**
   * Die Benutzereingabe kann je nach den Anforderungen und dem Verhalten der Chat-Engine angepasst werden.
   * Es ist ratsam, die Chat-Funktion zunächst mit grundlegenden Einstellungen zu testen und sie dann nach Bedarf zu erweitern.
   * Stelle sicher, dass die Signalwörter der gewählten Sprache entsprechen und möglichst klar und einfach gesprochen werden können um Missverständnisse zu verringern.

**Combination Funktion**

**Anleitung zur Konfiguration einer Kombinationsfunktion für den **Adaptable-Console-Voice-Assistent**:**

**Die Kombinationsfunktion macht es möglich mit einem Sprachbefehl hintereinander mehrere der zuvor kreierten Chat-, App- oder Tapo-Funktionen ablaufen zu lassen und diese so beliebig zu kombinieren.**

1. **Kombinations-Konfiguration:**
   * combination\_signal\_words = entspannen: Hier werden die Haupt-Signalwörter festgelegt, die den Assistenten dazu auffordern, die Kombinationsfunktion auszuführen. In diesem Beispiel ist es "entspannen".
   * combination\_and\_signal\_words =: Zusätzliche Signalwörter, die mit den Haupt-Signalwörtern kombiniert werden können. Hier sind keine zusätzlichen Wörter festgelegt.
2. **Benutzereingabe-Konfiguration:**
   * user\_combination\_input\_1 = feuer:

Der Benutzer kann hier die Signalwörter der Funktion eintragen die als erstes ablaufen soll. Es müssen natürlich die gleichen sein, die er auch für diese Funktion bereits zuvor gewählt hatte. In diesem Fall ist es „feuer“. Dies löst eine App Funktion aus die bei einem Youtubefork ein Kaminfeuervideo startet.

* + user\_combination\_input\_2 = musik playlist meditation:

Der Benutzer kann hier die Signalwörter der Funktion eintragen die als zweites ablaufen soll. Es müssen natürlich die gleichen sein, die er auch für diese Funktion bereits zuvor gewählt hatte. In diesem Fall ist es „musik playlist meditation“. Dies löst eine App Funktion aus die bei Spotify die Playlist Meditationsmusik startet.

* + Analog dazu können die weiteren user\_combination\_input´s genutzt werden.

1. **Hinweise:**
   * Stelle sicher, dass die Signalwörter der gewählten Sprache entsprechen und möglichst klar und einfach gesprochen werden können um Missverständnisse zu verringern.

**Tapo Funktion**