Pour mettre en œuvre la transcription décrite dans ton IDE PyCharm, voici comment structurer le projet et intégrer les différentes fonctionnalités dans les fichiers appropriés.

**Étape 1 : Crée les fichiers dans PyCharm**

Dans PyCharm, crée un nouveau projet appelé **Voice Assistant** et dans ce projet, crée plusieurs fichiers Python pour diviser les différentes fonctionnalités.

1. **voice\_assistant.py** - fichier principal qui initialise le projet et inclut l’exécution des tâches générales.
2. **modules.py** - fichier pour contenir toutes les fonctions spécifiques (ex. recherche sur Google, envoi d’emails, etc.).
3. **setup.py** - fichier pour gérer l’installation des dépendances et initialiser l'assistant vocal.

**Étape 2 : Contenu de setup.py**

Ce fichier s'assurera que tous les modules nécessaires sont installés avant d'exécuter le projet.

python

Copier le code

import os

# Installe toutes les dépendances nécessaires

def install\_dependencies():

os.system('pip install pyttsx3 speechrecognition wikipedia gtts playsound pyaudio requests wolframalpha selenium')

if \_\_name\_\_ == "\_\_main\_\_":

install\_dependencies()

print("Toutes les dépendances sont installées.")

**Étape 3 : Contenu de voice\_assistant.py**

Le fichier principal pour exécuter les commandes et les réponses du voice assistant.

python

Copier le code

import pyttsx3

import speech\_recognition as sr

from modules import \*

import datetime

import wikipedia

import webbrowser

import os

# Initialise l'engine de texte à voix

engine = pyttsx3.init()

voices = engine.getProperty('voices')

engine.setProperty('voice', voices[1].id) # Définit la voix féminine

def talk(text):

engine.say(text)

engine.runAndWait()

def listen\_command():

r = sr.Recognizer()

with sr.Microphone() as source:

print("Listening...")

r.pause\_threshold = 1

audio = r.listen(source)

try:

print("Recognizing...")

command = r.recognize\_google(audio, language='en-in')

print(f"User said: {command}\n")

except Exception as e:

print("Say that again please...")

return "None"

return command.lower()

if \_\_name\_\_ == "\_\_main\_\_":

wish\_me()

while True:

command = listen\_command()

if 'wikipedia' in command:

talk('Searching Wikipedia...')

command = command.replace("wikipedia", "")

results = wikipedia.summary(command, sentences=2)

talk(f"According to Wikipedia {results}")

elif 'open youtube' in command:

webbrowser.open("youtube.com")

elif 'open google' in command:

webbrowser.open("google.com")

elif 'play music' in command:

music\_dir = 'C:\\Users\\User\\Music'

songs = os.listdir(music\_dir)

os.startfile(os.path.join(music\_dir, songs[0]))

elif 'the time' in command:

str\_time = datetime.datetime.now().strftime("%H:%M:%S")

talk(f"The time is {str\_time}")

elif 'email to' in command:

try:

talk("What should I say?")

content = listen\_command()

to = "someone@example.com" # Ajoute l'email du destinataire

send\_email(to, content)

talk("Email has been sent.")

except Exception as e:

talk("Sorry, I am not able to send this email.")

**Étape 4 : Contenu de modules.py**

Ce fichier contiendra les fonctions supplémentaires telles que la gestion des emails, de la météo, et des recherches sur Google.

python

Copier le code

import smtplib

import requests

import json

# Fonction pour envoyer un email

def send\_email(to, content):

server = smtplib.SMTP('smtp.gmail.com', 587)

server.ehlo()

server.starttls()

server.login('your-email@gmail.com', 'your-password') # Mets tes identifiants

server.sendmail('your-email@gmail.com', to, content)

server.close()

# Fonction pour récupérer la météo

def get\_weather(city):

api\_key = "your\_openweather\_api\_key"

base\_url = f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api\_key}"

weather\_data = requests.get(base\_url).json()

if weather\_data['cod'] != '404':

main = weather\_data['main']

weather\_description = weather\_data['weather'][0]['description']

temperature = main['temp'] - 273.15

humidity = main['humidity']

return f"Temperature: {temperature:.2f}°C\nHumidity: {humidity}%\nDescription: {weather\_description}"

else:

return "City not found"

# Fonction pour saluer l'utilisateur

def wish\_me():

hour = int(datetime.datetime.now().hour)

if 0 <= hour < 12:

talk("Good Morning!")

elif 12 <= hour < 18:

talk("Good Afternoon!")

else:

talk("Good Evening!")

talk("I am your assistant. Please tell me how can I help you.")

**Étape 5 : Lancement du projet**

1. Exécute d'abord le fichier setup.py pour installer les dépendances.
2. Ensuite, lance voice\_assistant.py pour démarrer l'assistant vocal.

**Conclusion**

Avec ces fichiers, tu auras une structure de projet bien organisée dans PyCharm. Chaque fichier gère une partie spécifique du projet, facilitant l'ajout et la gestion des fonctionnalités.

4o

**Vous avez dit :**

il y a des code a faire sur le terminal ?

**ChatGPT a dit :**

ChatGPT

Oui, il y a quelques commandes à exécuter dans le terminal de PyCharm pour installer des bibliothèques nécessaires et gérer certaines fonctionnalités du projet. Voici les étapes à suivre pour bien démarrer :

**Étape 1 : Installer les dépendances**

Tu devras exécuter certaines commandes dans le terminal pour installer les bibliothèques utilisées dans le projet, comme pyttsx3, speech\_recognition, wikipedia, et d'autres.

**Ouvre le terminal dans PyCharm et exécute les commandes suivantes une par une :**

bash

Copier le code

pip install pyttsx3

pip install SpeechRecognition

pip install wikipedia

pip install gtts

pip install playsound==1.2.2 # Cette version est compatible

pip install pyaudio

pip install requests

pip install wolframalpha

pip install selenium

je suis ici

**Étape 2 : Installation du WebDriver pour Selenium**

Si tu utilises **Selenium** pour automatiser la navigation, tu devras aussi installer **ChromeDriver** (ou un autre WebDriver selon ton navigateur).

**Télécharge ChromeDriver (ou EdgeDriver/GeckoDriver pour Firefox)**

* Va sur https://chromedriver.chromium.org/downloads et télécharge la version correspondant à ta version de Chrome.
* Après avoir téléchargé **chromedriver.exe**, place-le dans un dossier facile à trouver, comme ton bureau ou dans le dossier de ton projet.

**Étape 3 : Initialiser chromedriver dans le code**

Après avoir téléchargé **ChromeDriver**, tu devras le lier dans ton code Selenium.

**Par exemple, dans ton fichier voice\_assistant.py ou modules.py, ajoute ceci :**

python

Copier le code

from selenium import webdriver

# Indique l'emplacement de ChromeDriver

driver = webdriver.Chrome(executable\_path='C:/path\_to\_chromedriver/chromedriver.exe')

# Ouvre le site Domino's (exemple)

driver.get("https://www.dominos.com")

**Étape 4 : Utilisation de l'API OpenWeatherMap**

Si tu utilises des APIs comme OpenWeatherMap, tu devras créer un compte et obtenir une **clé API**.

**Pour OpenWeatherMap :**

1. Va sur https://home.openweathermap.org/users/sign\_up et crée un compte.
2. Récupère ta clé API dans ton tableau de bord.
3. Remplace "your\_openweather\_api\_key" dans ton code par ta clé API.

**Étape 5 : Vérifier et exécuter le projet**

Une fois que tu as installé toutes les dépendances et configuré ton WebDriver, tu peux exécuter ton fichier voice\_assistant.py dans PyCharm pour démarrer l'assistant vocal.

bash

Copier le code

python voice\_assistant.py

**Résumé des commandes à exécuter dans le terminal :**

1. Installer les bibliothèques :
   * pip install pyttsx3 SpeechRecognition wikipedia gtts playsound pyaudio requests wolframalpha selenium
2. Télécharge et configure ChromeDriver.
3. Obtiens ta clé API OpenWeatherMap et configure-la dans le code.

Avec ces étapes, ton environnement sera prêt pour exécuter l'assistant vocal dans PyCharm !

4o

Haut du formulaire

Bas du formulaire

ChatGPT peut faire des erreurs. Envisagez de vérifier les inf