learnwithbhawana

Python JSON

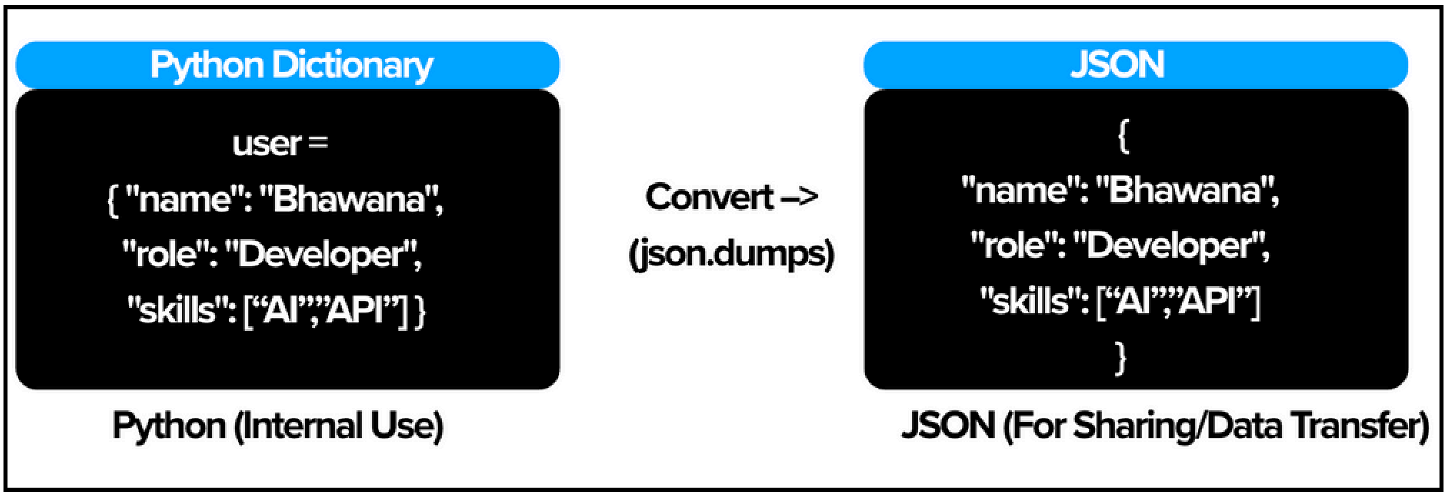
▢ What is JSON?

- JSON stands for *JavaScript Object Notation.*
- It is a lightweight format used to store and share data.
- It looks similar to a Python dictionary and is commonly used in APIs, web apps, and data transfer.

▢ Example Usage of JSON in Real Life:

- When you log in to a website, your profile details are sent in JSON format.
- When you call an API (like weather data or stock prices), the response comes in JSON format.
- Mobile apps use JSON to send/receive data from servers.

▢ Visual Diagram — JSON vs Python Dictionary



```
user =
{ "name": "Bhawana",
"role": "Developer",
"skills": [“AI”,”API”] }
```

```
{
"name": "Bhawana",
"role": "Developer",
"skills": [“AI”,”API”]
}
```

▢ Code + Explanation :

Here’s a beginner-friendly version of your code, saved as a **complete code file** with **line-by-line explanation** so even someone new to Python can understand it easily.

- Code :

```python
# Step 1: Import the json module.
# This module helps us convert Python data into JSON format and vice versa.
import json

# Step 2: Create a Python dictionary named 'user'.
# A dictionary stores data in key-value pairs.
user = {
    "name": "Bhawana",              # Key: "name", Value: "Bhawana" (a string)
    "role": "Python Developer",     # Key: "role", Value: "Python Developer"
    "skills": ["AI", "Automation", "APIs"]  # Key: "skills", Value: List of strings
}

# Step 3: Convert the Python dictionary into a JSON string using json.dumps()
json_data = json.dumps(user,indent = 4)

# Step 4: Print the converted JSON string
print("JSON Output:\n", json_data)

# Step 5: Open a file named 'user.json' in write mode ("w") to save the JSON string.
# The 'with' block automatically closes the file after writing.
with open("user.json", "w") as file:
    json.dump(json_data, file)  # Save the JSON string into the file

# Step 6: Open the same 'user.json' file in read mode ("r") to load the data back.
with open("user.json", "r") as file:
    data = json.load(file)  # Read the file content and load it as Python data
    print("\nLoaded from file:", data)  # Print the data read from file
```

- Output:

```
JSON Output:
 {
    "name": "Bhawana",
    "role": "Python Developer",
    "skills": [
        "AI",
        "Automation",
        "APIs"
    ]
}

Loaded from file: {
    "name": "Bhawana",
    "role": "Python Developer",
    "skills": [
        "AI",
        "Automation",
        "APIs"
    ]
}
```

- **Quick Breakdown for Beginners:**

Concept	**Explanation**
Dictionary	A data format in Python like a mini database with key: value.
JSON	A universal format used to share data between websites, apps, APIs, etc.
json.dumps( )	Converts Python data → JSON string
json.dump( )	Saves JSON data into a file
json.load( )	Reads JSON data back from a file

▢ Real Example — Fetching JSON from GitHub API

- Code :

```python
import requests  # Used to call APIs

# Step 1: API endpoint of a GitHub user profile (public data)
url = "https://api.github.com/users/pythonessdatadiaries"  # Replace with any GitHub username

# Step 2: Send GET request to fetch data
response = requests.get(url)

# Step 3: Convert response to JSON (dictionary format)
data = response.json()

# Step 4: Print some useful information
print("GitHub User Info (JSON):")
print("Name:", data["name"])
print("Public Repos:", data["public_repos"])
print("Followers:", data["followers"])
```

- Output:

```
GitHub User Info (JSON):
Name: Bhawana Saxena
Public Repos: 3
Followers: 2
```

- **Quick Breakdown for Beginners:**

Step	Action	Format
Send request to API	requests.get()	JSON text from server
Convert to Python	.json()	Dictionary
Access Values	data["name"]	Easy to use

▢ Real Example : AI Based model (Sentiment analysis):

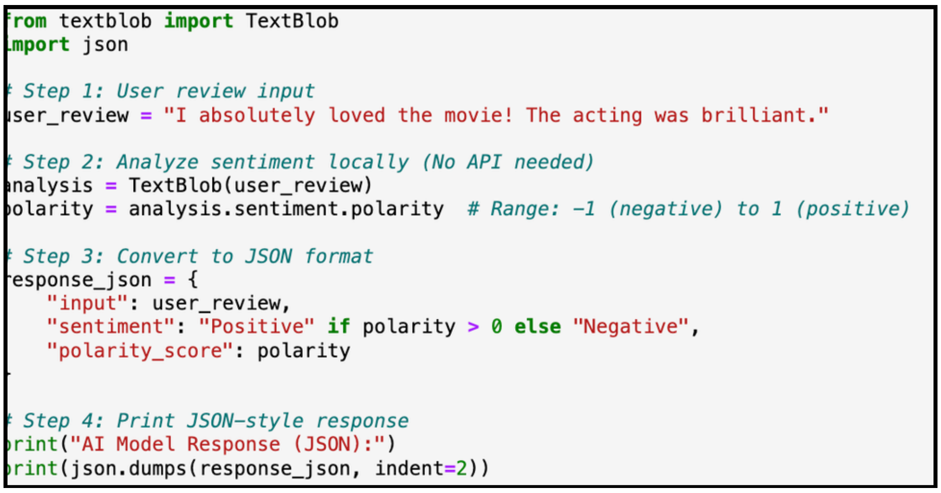
We'll use a **lightweight AI model installed locally** called **TextBlob**. It can analyze sentiment directly.

- **Installation (Run Once):**

pip install textblob

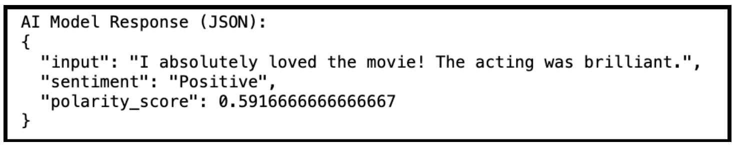
python -m textblob.download_corpora

- Code :



```python
from textblob import TextBlob
import json

# Step 1: User review input
user_review = "I absolutely loved the movie! The acting was brilliant."

# Step 2: Analyze sentiment locally (No API needed)
analysis = TextBlob(user_review)
polarity = analysis.sentiment.polarity  # Range: -1 (negative) to 1 (positive)

# Step 3: Convert to JSON format
response_json = {
    "input": user_review,
    "sentiment": "Positive" if polarity > 0 else "Negative",
    "polarity_score": polarity
}

# Step 4: Print JSON-style response
print("AI Model Response (JSON):")
print(json.dumps(response_json, indent=2))
```

- Output:



```
AI Model Response (JSON):
{
  "input": "I absolutely loved the movie! The acting was brilliant.",
  "sentiment": "Positive",
  "polarity_score": 0.5916666666666667
}
```

- **Quick Breakdown for Beginners:**

Line of Code	**What it Does**	**Beginner-Friendly Explanation**
from textblob import TextBlob	Imports TextBlob library	TextBlob is a simple AI tool that can understand text, check sentiment, etc.
import json	Imports JSON library	Allows us to convert Python data into JSON format (like a dictionary for sharing data).
user_review = "I absolutely loved the movie! The acting was brilliant."	Stores the review	This is the text we want the AI to analyze.
analysis = TextBlob(user_review)	Creates a TextBlob object	TextBlob will analyze the text for things like sentiment, grammar, etc.
polarity = analysis.sentiment.polarity	Gets sentiment polarity	Returns a number between -1 and 1: negative → -1, positive → 1, neutral → 0.
response_json = { ... }	Creates a JSON-style dictionary	We store the review, the sentiment (Positive/Negative), and polarity score in a dictionary.
"sentiment": "Positive" if polarity > 0 else "Negative"	Checks if polarity is positive	If the score > 0 → Positive, otherwise Negative.
print("AI Model Response (JSON):")	Prints a header	Just shows the output clearly.
print(json.dumps(response_json, indent=2))	Converts Python dictionary to JSON string and prints nicely	JSON format is easy to read and used everywhere in real-life AI applications.