

## You

How can I create an application that intelligently extracts the text from a word document uploaded as a file and can insert the text into the relevant places in another word document in python using AI technologies and including the features defined in the attached screenshot (ignore the bits about the AI chat bot and document names summaries). I have access to Google Cloud workspace and Vertex AI.

## Copilot

Certainly! Let's break down the steps to create an application that extracts text from an uploaded Word document and inserts it into relevant places in another Word document using Python and AI technologies. I'll also address the features mentioned in the attached screenshot, excluding the chatbot-related functionality and document name summaries.

### 1. Text Extraction:

- First, you'll need to extract the text from the uploaded Word document. You can use the python-docx library to read the document and retrieve its content.
- Install the library using:
- `pip install python-docx`
- Extract the text from the document:

python

```
from docx import Document
```

```
def extract_text_from_docx(file_path):
```

```
    doc = Document(file_path)
```

```
    full_text = [para.text for para in doc.paragraphs]
```

```
    return "\n".join(full_text)
```

```
# Example usage:
```

```
uploaded_text = extract_text_from_docx("path_to_uploaded_document.docx")
```

### 2. AI Model Training:

- Next, you'll need to train an AI model to identify which parts of the extracted text correspond to relevant sections in the target Word document.
- Consider using Google Cloud's Vertex AI platform for model training. You can create a custom NLP model that learns from examples where manual placement of text has been done correctly.

### 3. Text Insertion:

- Once the model is trained, use it to predict where each piece of extracted text should go in the new Word document.
- Again, use the python-docx library to insert this text into the correct places in the new document.

#### 4. Integration with Google Cloud Workspace:

- Utilize Google Cloud APIs to integrate your application with Google Cloud Workspace for storing documents and managing access.

#### 5. Features (as per Screenshot):

- **Gradual Roll-out of Features:**
  - Start with the most relevant features first.
  - For example, implement text extraction and insertion for specific document types (e.g., referral forms, case notes) before expanding to other features.
- **Document Summarization:**
  - Summarize the contents of a folder based on file titles.
  - Count how many contacts are based on case notes.
  - Implement restrictions as needed.
- **AI Chatbot (Optional):**
  - Create an AI chatbot similar to GPT.
  - Allow users to ask specific functionality-related queries.

Remember that this is a high-level overview, and each step involves more detailed programming tasks. Ensure error handling, security considerations, and user-friendly interfaces as you build your application. Good luck! 🚀

Learn more: 1. [pythoninoffice.com](https://pythoninoffice.com) 2. [katemessner.com](https://katemessner.com) 3. [twitter.com](https://twitter.com) 4. [www.benchmarksixsigma.com](https://www.benchmarksixsigma.com) 5. [www.civildaily.com](https://www.civildaily.com) 6. [friscochronicles.com](https://friscochronicles.com) 7. [www.youtube.com](https://www.youtube.com) 8. [www.youtube.com](https://www.youtube.com) 9. [www.youtube.com](https://www.youtube.com) 10. [doc2lang.com](https://doc2lang.com) 11. [blog.finxter.com](https://blog.finxter.com) 12. [stackoverflow.com](https://stackoverflow.com)

### You

What is the best way to install the aforementioned python library (python-docx) in VS code? I'm using a laptop where access to command prompt and pip commands are blocked. Should I switch to Pycharm or is there a way in VS code?

### Copilot

Certainly! If you're using Visual Studio Code (VS Code) and don't have access to the command prompt or pip commands, you can still install Python libraries directly within VS Code. Here's how:

#### 1. Open VS Code:

- Launch VS Code on your laptop.

#### 2. Install Python Extension:

- Click on the Extensions tab (or use Ctrl+Shift+X) to open the Extensions section.

- Search for "python."
- The first result should display an extension named "Python" by Microsoft.
- Click on it and install the extension1.

### 3. Python Interpreter:

- You'll need to install a Python interpreter separately from the extension.
- For a quick installation, download Python from the official website (python.org) and install it.
- Once Python is installed, the VS Code extension will recognize it and allow you to write and run Python code directly in VS Code2.

### 4. Using python-docx:

- Now that you have the Python extension installed, you can use pip within VS Code to install python-docx.
- Open a Python file or create a new one.
- In the integrated terminal (accessible within VS Code), run the following command:
- `pip install python-docx`

### 5. Alternative Approach (Manual Installation):

- If pip is not available, you can manually download the distribution from PyPI:
  - Visit the python-docx documentation for the download link.
  - Unpack the tarball and navigate to the extracted folder.
  - Run `python setup.py install`.
- Note that python-docx depends on the lxml package, so ensure you have it installed as well.

You don't necessarily need to switch to PyCharm; VS Code is a powerful IDE for Python development, and with the right extensions, you can achieve your goals effectively. Happy coding! 📌🚀

Learn more: 1. [pythondex.com](https://pythondex.com) 2. [code.visualstudio.com](https://code.visualstudio.com) 3. [python-docx.readthedocs.io](https://python-docx.readthedocs.io)

**You**

QUESTION\_PLACEHOLDER 3

**Copilot**

Certainly! If you're working on **Windows** and don't have access to the command prompt, you can still extract files from a tarball (such as .tar.gz, .tgz, or .gz) using alternative methods:

#### 1. Using File Manager (Windows Explorer):

- Locate the tarball file (e.g., filename.tar.gz) in Windows Explorer.
- Right-click on the file.
- Choose "Extract" or "Extract All."
- Select the destination folder where you want the contents to be extracted.
- Click "Extract" to unzip and extract the files1.

## 2. Using 7-Zip (Third-Party Software):

- Download and install 7-Zip if you don't have it already.
- Right-click on the tarball file.
- Choose "7-Zip" from the context menu.
- Select "Extract Here" to extract the contents to the current folder or choose "Extract to [folder name]" to specify a destination folder<sup>2</sup>.

Remember that these methods allow you to extract files from tarballs without using the command prompt.

Happy extracting! 📁 🔗

Learn more: 1. [windowsloop.com](https://www.windowsloop.com) 2. [wiki.haskell.org](https://wiki.haskell.org) 3. [howtogeek.com](https://howtogeek.com) 4. [thewindowsclub.com](https://thewindowsclub.com)  
5. [youtube.com](https://youtube.com) 6. [linuxhandbook.com](https://linuxhandbook.com) 7. [linuxize.com](https://linuxize.com) 8. [cyberciti.biz](https://cyberciti.biz)

## You

My python file has file path "C:\Users\HasanAkh1\OneDrive - bradford.gov.uk\Markdown and Python\wordDocTester.py" and the module I want to import has file path "C:\Users\HasanAkh1\OneDrive - bradford.gov.uk\Markdown and Python\python\_docx-1.1.2\src\docx\document.py", how can I import the module in my python file?

## Copilot

To import a module from a specific file path in Python, you can use the `importlib.util` module. Here's how you can do it:

### 1. Using `importlib.util.spec_from_file_location()`:

- Specify the full path to the module you want to import (e.g., "C:\\Users\\HasanAkh1\\OneDrive - bradford.gov.uk\\Markdown and Python\\python\_docx-1.1.2\\src\\docx\\document.py").
- Create a module specification using `importlib.util.spec_from_file_location(module_name, module_path)`.
- Create a new module object based on the specification using `importlib.util.module_from_spec(spec)`.
- Execute the module in its own namespace using `spec.loader.exec_module(module)`.

Here's an example of how you can import the `document.py` module from the specified path:

```
python
```

```
import importlib.util
```

```
# Specify the full path to the module
```

```
module_path = "C:\\Users\\HasanAkh1\\OneDrive - bradford.gov.uk\\Markdown and Python\\python_docx-1.1.2\\src\\docx\\document.py"
```

```
# Create a module specification
```

```
spec = importlib.util.spec_from_file_location("document", module_path)
```

```
# Create a new module object
```

```
module = importlib.util.module_from_spec(spec)
```

```
# Execute the module
```

```
spec.loader.exec_module(module)
```


```
# Now you can use the module in your code
```

```
# For example:
```

```
from document import some_function
```

```
# Call the function from the imported module
```

```
some_function()
```

Replace "some\_function" with the actual function or class you want to use from the document.py module. Make sure to adjust the paths according to your system. 

Learn more: 1. [geeksforgeeks.org](https://www.geeksforgeeks.org) 2. [python-code.dev](https://python-code.dev) 3. [ioflood.com](https://ioflood.com) 4. [geeksforgeeks.org](https://www.geeksforgeeks.org)

## You

Are you sure this works? This is my python code:

```
import importlib.util
```

```
# Specify the full path to the module
```

```
module_path = "C:\\Users\\HasanAkh1\\OneDrive - bradford.gov.uk\\Markdown and Python\\python_docx-1.1.2\\src\\docx"
```

```
# Create a module specification
```

```
spec = importlib.util.spec_from_file_location("document", module_path)
```

```
# Create a new module object
```

```
module = importlib.util.module_from_spec(spec)
```

```
# Execute the module
```

```
spec.loader.exec_module(module)
```

```
from document import Document
```

```
def extract_text_from_docx(file_path):
```

```
doc = Document(file_path)
```

```
spec.loader.exec module(module)
```

# Now you can use the classes/functions from the imported module

```
from document import Document
```

```
def extract_text_from_docx(file_path):
```

```
    doc = Document(file_path)
```

```
    full_text = [para.text for para in doc.paragraphs]
```

```
    return "\n".join(full_text)
```

# Example usage:

```
uploaded_text = extract_text_from_docx("C:/Users/HasanAkh1/OneDrive - bradford.gov.uk/Word  
docs/Report about AI in BCFT - Draft 2.docx")
```

Make sure to replace "document" with the actual module name if it's different. This should resolve the issue, and you'll be able to use the Document class from the document.py module. Let me know if you encounter any further issues! 🚀

Learn more: 1. [geeksforgeeks.org](https://www.geeksforgeeks.org) 2. [python-code.dev](https://python-code.dev) 3. [ioflood.com](https://ioflood.com) 4. [geeksforgeeks.org](https://www.geeksforgeeks.org)  
5. [stackoverflow.com](https://stackoverflow.com) 6. [docs.python.org](https://docs.python.org)

---

[Conversation by Fifa Utuber with Copilot on Tue, Aug 20 and Wed, Aug 21 2024](#)