To update an existing Jupyter notebook (xyz.ipynb), appending to the file directly using open("xyz.ipynb", "a", encoding="utf-8") is not the correct approach. This is because a Jupyter notebook file is a JSON document, and appending to it as if it were a text file would corrupt the JSON structure.

**Correct Approach to Update a Notebook**

To update an existing notebook, you should:

1. **Read the existing notebook.**
2. **Modify the notebook object.**
3. **Write the modified notebook back to the file.**

Here’s how you can do it:

import nbformat

# Read the existing notebook

with open("xyz.ipynb", "r", encoding="utf-8") as f:

nb = nbformat.read(f, as\_version=4)

# Add new content to the notebook (example: adding a new markdown cell)

new\_text = """

## Adding a New Section

This section is added as an update to the existing notebook.

"""

# Add a new markdown cell

nb.cells.append(nbformat.v4.new\_markdown\_cell(new\_text))

# Add a new code cell (example)

new\_code = """

# New code cell added during the update

print("Hello from the updated notebook!")

"""

nb.cells.append(nbformat.v4.new\_code\_cell(new\_code))

# Write the updated notebook back to the file

with open("xyz.ipynb", "w", encoding="utf-8") as f:

nbformat.write(nb, f)

print("xyz.ipynb Updated")

**Steps Explained:**

1. **Reading the Existing Notebook:**  
   You open the existing xyz.ipynb file in read mode, parse it using nbformat.read, and load it into a notebook object.
2. **Modifying the Notebook:**  
   You can append new cells (markdown or code) to the notebook object.
3. **Writing Back the Updated Notebook:**  
   You write the updated notebook object back to the xyz.ipynb file, overwriting the existing content with the new content.

This approach ensures that the notebook remains a valid JSON document and allows you to safely add new content.