A white background with red and black text

Description automatically generated

**Code Refactoring and Bug Fixing**

Suswarah

A white background with black text

Description automatically generated

**Note taking Application  
 Outline**

**Problem Statement:**

A team of enthusiastic data scientists embarked on a mission to develop a Note Taking Application using Python, Flask, and HTML. However, their lack of experience in backend development has led to challenges in making the application fully functional. Recognizing your proficiency in backend development, you have been tasked with fixing the broken code and ensuring the application works seamlessly.

**Objective:**

Refactor the existing codebase and ensure the proper functioning of the Note Taking Application. Document all identified bugs during the debugging process. Remember, the task is not about recreating the app from scratch. Your goal is to fix the already existing codebase and make the application work as intended.

**Summary of the Code:**

The application's home route contains a text field and a button. Users can add a note, and all the notes should be displayed as an unordered list below the text field on the same page.

A white background with black text

Description automatically generatedA white background with black text

Description automatically generated

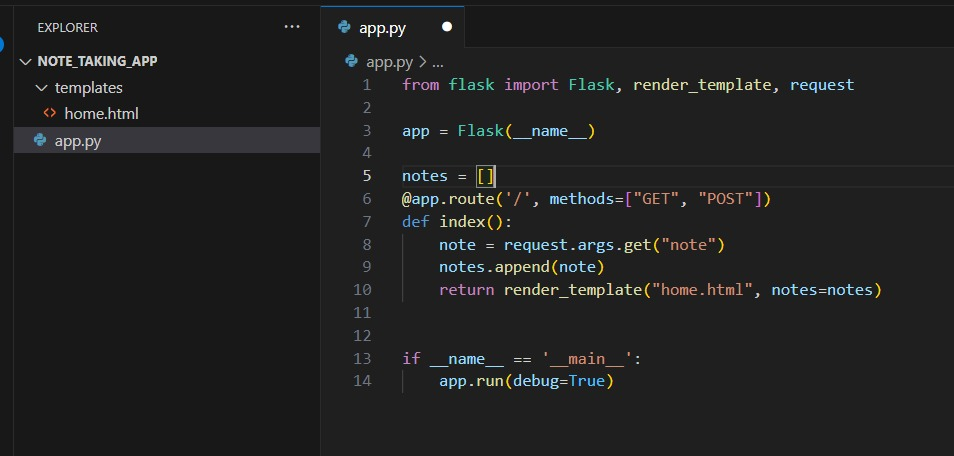
**Handling Bugs**

**Bug 1:**

After running the **app.py** code accompanied by the **home.html**, we end up with a message “Method Not Allowed The method is not allowed for the requested URL”.

**Cause:**   
This error indicates that the method specified in the primary route @app.route('/'), ‘method argument’ consists only POST method leading to the error.

**Handling:**   
Included GET method, in order to handle GET requests from the URL.



A white background with black text

Description automatically generated

**Handling Bugs**

**Bug 2:**

After fixing the primary issue, we could see the application directly return a “None” list item immediately after running app.py. Also, occurs recursively on adding any notes to the text box.

**Cause:**   
The issue arises from the attempt to retrieve the "note" parameter using **request.arg.get("note")**, which looks for the parameter in the URL's query string (GET parameters). However, the "note" is actually transmitted in the request body (POST parameters) from the HTML form.

**Handling:**   
Changed the request.args.get(“note”) to request.form.get(“note”) to correctly retrieve the information taken from the text box. To avoid information from text box to the URL's query string parameters, added the POST method in the <form>.

A screenshot of a computer screen

Description automatically generated

A white background with black text

Description automatically generated

**Enhancing User Experience**

**To Enhance:**

Empty note list item appears if an empty text note is added.

**Cause:**   
This is caused due to there is no condition given for the notes added to the text box.

**Handling:**   
An “If else” condition is introduced to fix if any empty note is given. This fix gives a message to the user “No notes available” to display.

A screenshot of a computer program

Description automatically generated

A white background with black text

Description automatically generated

**Conclusion**

In conclusion, while this marks my first venture into backend development, grasping the underlying concepts has proven to be challenging. However, with time, every line of code began to make sense. As each bug was addressed, the comprehension of the code improved. By the completion of the application development process, I successfully identified and resolved the bugs.