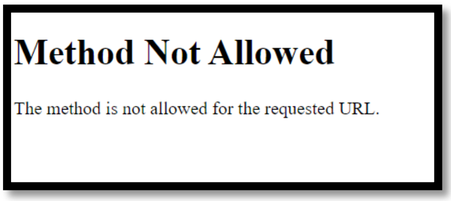
# Report On Code Refactoring and bug fixing

**Problem Statement:** A group of eager data scientists set out to create a Note Taking Application utilizing Python, Flask, and HTML. However, their limited expertise in backend development has resulted in difficulties in achieving full functionality for the application. Our responsibility is rectify the faulty code and guarantee smooth operation of the application.

## BUG 1 : GET METHOD USED IN POST REQUEST



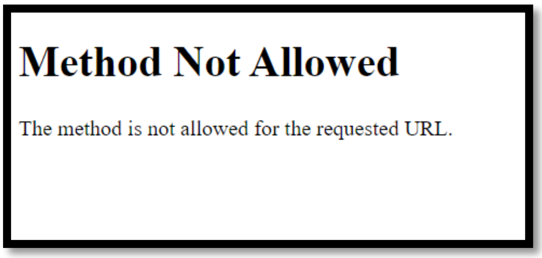
Method Not Allowed

The method is not allowed for the requested URL.

**Description:** The problem lies in the route definition of the index function. It specifies the method as "POST", but the route itself is defined using @app.route('/', methods=["POST"]). This setup restricts the route to only accept POST requests. However, the scenario suggests an attempt to handle a form submission, which conventionally utilizes POST requests, yet the form data appears to be transmitted as query parameters, a practice associated with GET requests.

**Resolution :** To fix this issue, we need to change **request.args.get("note")** to **request.form.get("note")**, as request.form is used to access form data sent with a POST request.

## BUG 2: GET REQUEST NOT ADDED IN ROUTE LIST



Method Not Allowed

The method is not allowed for the requested URL.

**Description:** When a user navigates to the root URL ("/") in their browser or sends a GET request to that URL, Flask will respond with a "Method Not Allowed" error because there is no route defined for handling GET requests at that endpoint.

**Resolution :** Adding GET request in the route method list in flask code

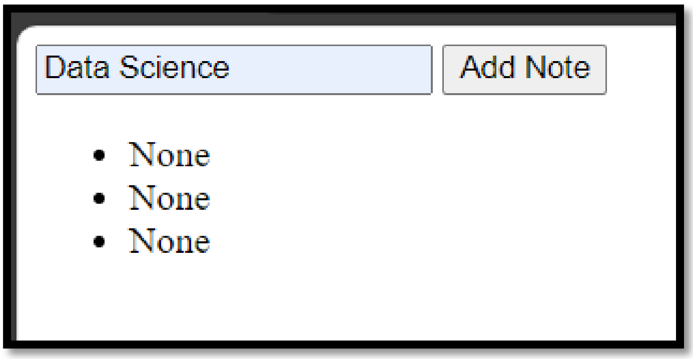
**@app.route('/', methods=["GET","POST"])**

## BUG 3: MISSING CONDITION TO HANDLE POST REQUEST SPECIFICALLY

**Description:** If the condition is not provided the index() function will execute the same code for both GET and POST requests. It means that any request, regardless of its method, will append the note to the notes list and render the template with the updated list of notes.

This might not be problematic as our application doesn't have specific requirements for handling different types of requests differently. However, it's generally good practice to differentiate between GET and POST requests, especially if there is different logic to execute for each type of request.

## BUG 4: ITEMS IN LIST ARE NOT ADDED IT IS SHOWING NONE



- None
- None
- None

**Description :** Here the problem lies in the html code. In the HTML code the form does not explicitly specify the method for data submission. When a method is not specified, the form defaults to using the GET method. Consequently, the form data will be appended to the URL as query parameters. However, the Flask route is configured to handle POST requests for processing the form submission.

**Resolution:**To fix this, we have to specify the method attribute in the form to POST

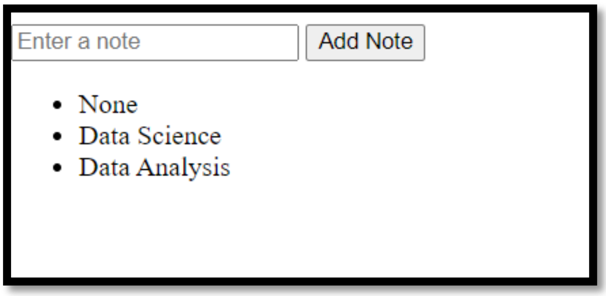
```
<form action="/" method="POST">
```

## BUG 5 : SUBMIT BUTTON MISSING

**Description:**Submit Button missing it can lead to  unexpected behaviour

**Resolution :** Add submit button in the html code

## BUG 6 : THE FIRST ITEM IS SHOWING AS  NONE

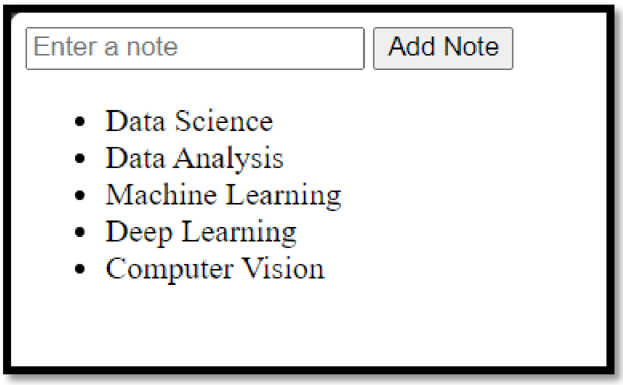


- None
- Data Science
- Data Analysis

**Description:** The problem arises when the form is submitted without any input in the text field, resulting in the transmission of an empty string or None value to the server.

**Resolution:** we  incorporate a condition to verify that the submitted note is not empty prior to appending it to the notes list.

**After fixing the Error- the app is working fine**



- Data Science
- Data Analysis
- Machine Learning
- Deep Learning
- Computer Vision

## CONCLUSION:

To summarize all the bug fixing

- Corrected the Flask route to accept both GET and POST requests.
- Modified the HTML form to specify the method as POST, ensuring alignment between form submission and route handling methods.
- Implemented a check in the Flask route to ensure only non-empty notes are appended to the notes list.
- Enhanced the template to conditionally render list items only when the note is non-empty, effectively addressing the issue of displaying empty bullet points when no notes are present.