INNOMATICS®
RESEARCH LABS

INNOVATION. AUTOMATION. ANALYTICS

PROJECT ON

Code Refactoring and Bug Fixing for Scribble

By
Hannah Igboke

About me

I am **Hannah Igboke**, and one of my dreams is to work at **NASA** one day. Alongside my degree in Chemical Engineering and my interest in material science, I am a data analyst with a proven track record as a dedicated and lifelong learner, contributing to individual and organizational success. I am excited to explore the power of data to address domain-specific issues and aid decision-making.

To do this, I have built and keep expanding my skill-set in **problem solving** and analytical thinking, **writing** and communication, **SQL** database systems like Microsoft SQL Server and MySQL, **Python** libraries (Pandas, Numpy, Seaborn, etc.), and **Power BI**. I also have working technical knowledge of backend systems and tools, including **Flask** and **HTML**. All in all, I love leveraging technology to tackle complex problems and deliver impactful solutions.

Connect with me on:

INNOMATICS RESEARCH LABS

Report contents

Background

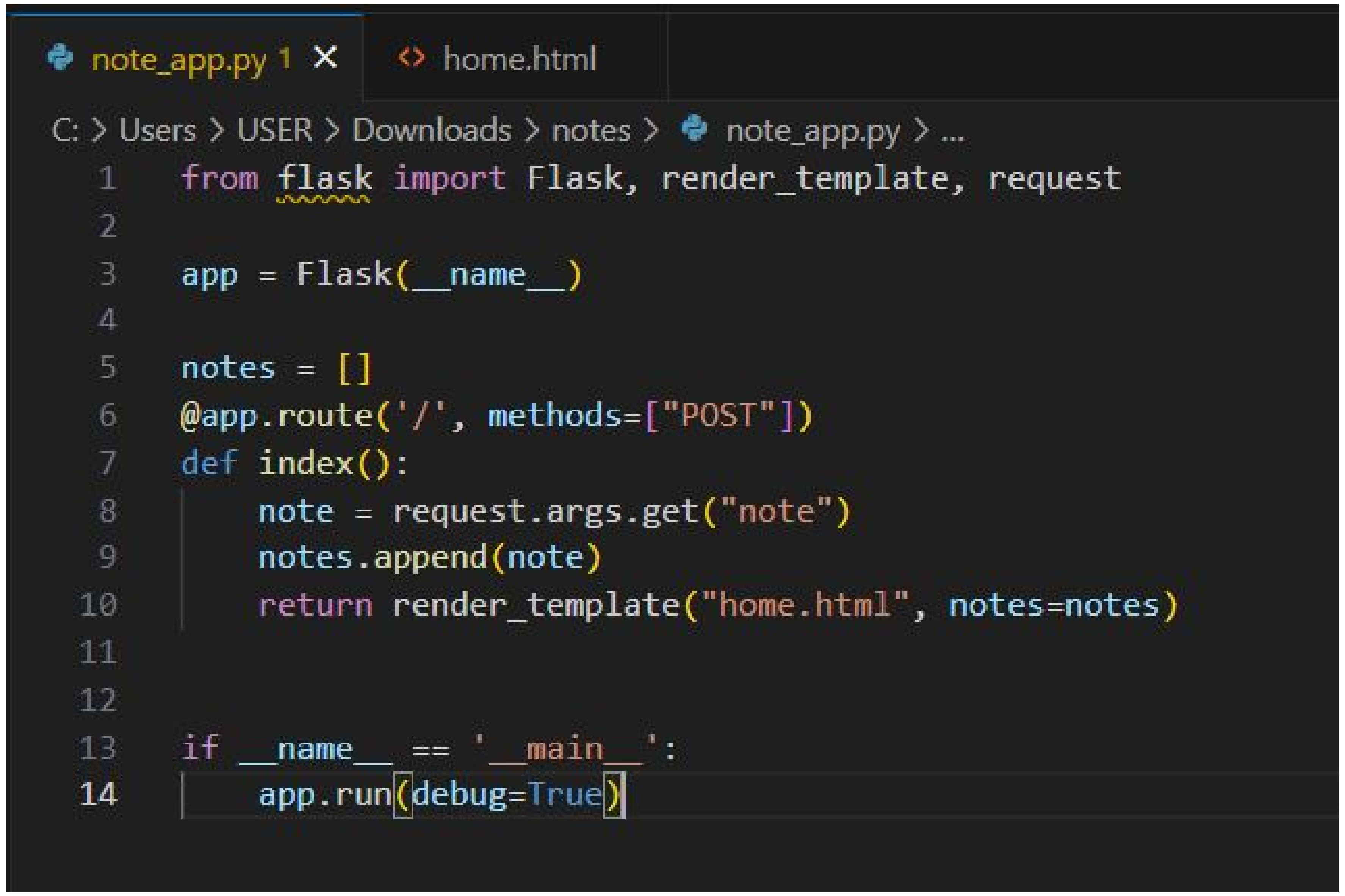
A team of enthusiastic data scientists identified a need to develop a personalized Note Taking Application called **Scribble**. To build this, they have to utilize Python, HTML and Flask framework to create the application server. While at the development stage, the team ran into some **challenges** with the backend which prevented it from being fully functional. Recognizing my proficiency in backend development, I have been tasked with fixing the broken code to ensure the application works seamlessly.

How it is supposed to work

The Scribble application is such that the home route or page contains a text field and a button. And users can add a note, and all the notes will be displayed as an unordered list below the text field on the same page.

Project description

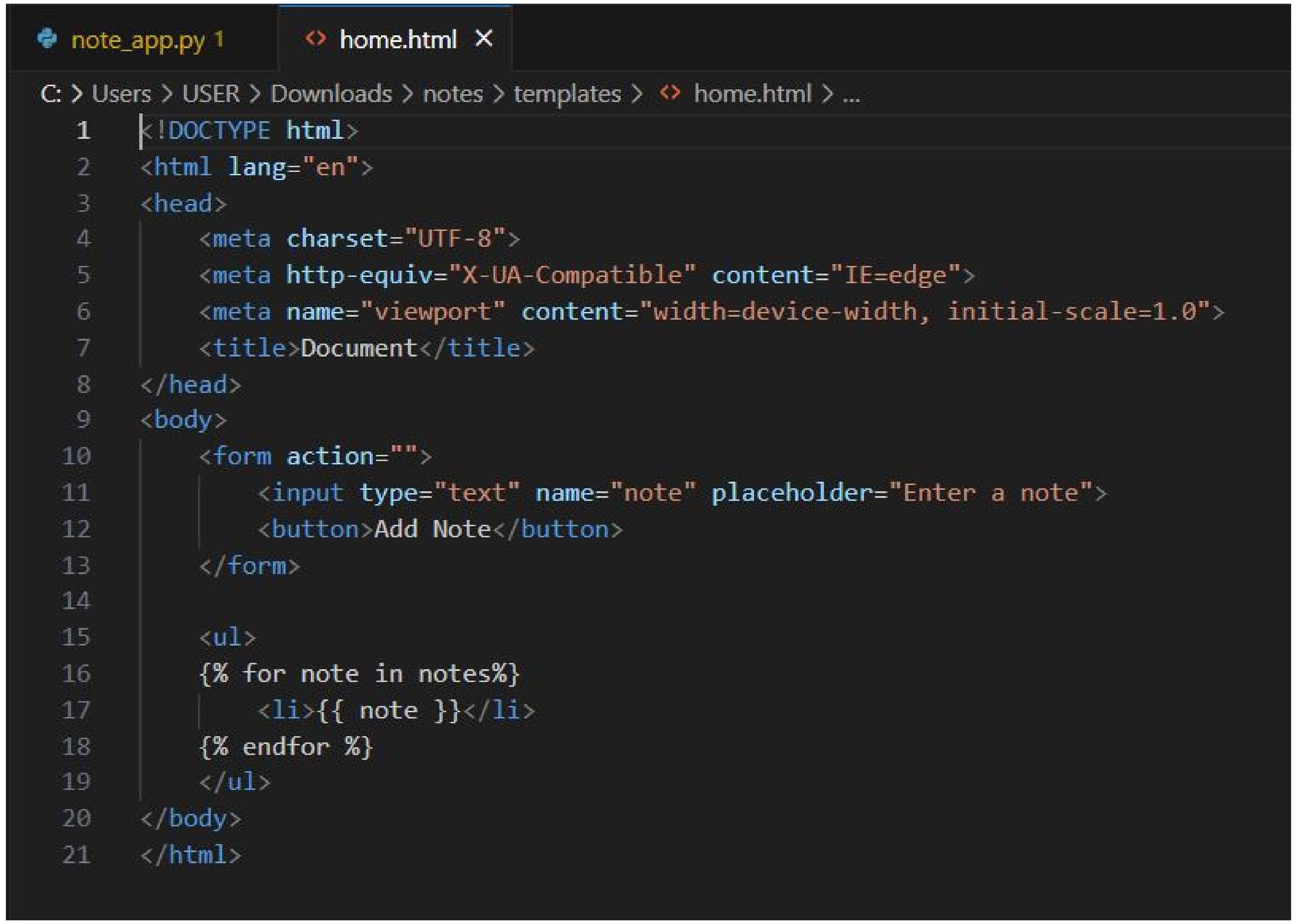
I have been provided with the code base for this project as follows:



```python
from flask import Flask, render_template, request

app = Flask(__name__)

notes = []
@app.route('/', methods=["POST"])
def index():
    note = request.args.get("note")
    notes.append(note)
    return render_template("home.html", notes=notes)


if __name__ == '__main__':
    app.run(debug=True)
```

Python - flask codebase

HTML file


note_app.py 1
home.html
C: > Users > USER > Downloads > notes > templates > home.html > ...

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <form action="">
        <input type="text" name="note" placeholder="Enter a note">
        <button>Add Note</button>
    </form>

    <ul>
    {% for note in notes%}
        <li>{{ note }}</li>
    {% endfor %}
    </ul>
</body>
</html>
```

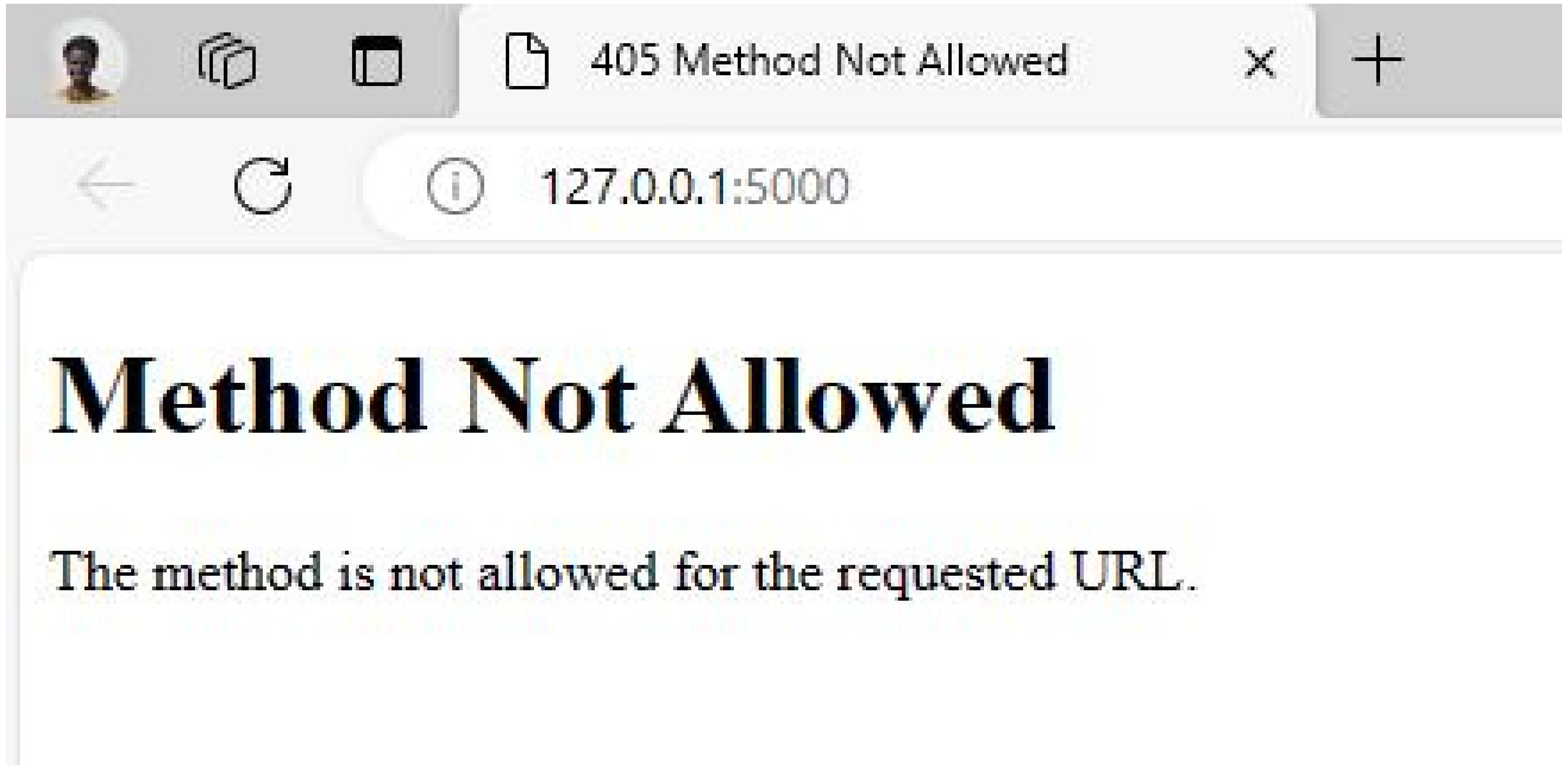
Objectives

The objectives for this project are twofold, namely:

- To refactor the existing codebase and ensure the proper functioning of the note-taking application,
- Document all identified bugs during the debugging process.

Run the code

When the code base is run, a 405 error occurs.



405 Method Not Allowed

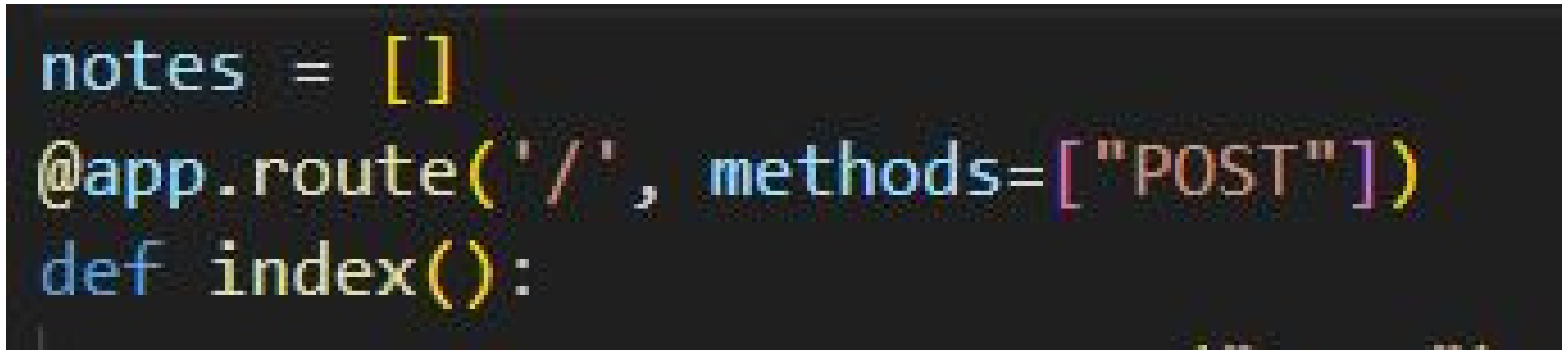
127.0.0.1:5000

```
Method Not Allowed

The method is not allowed for the requested URL.
```

Why is this?

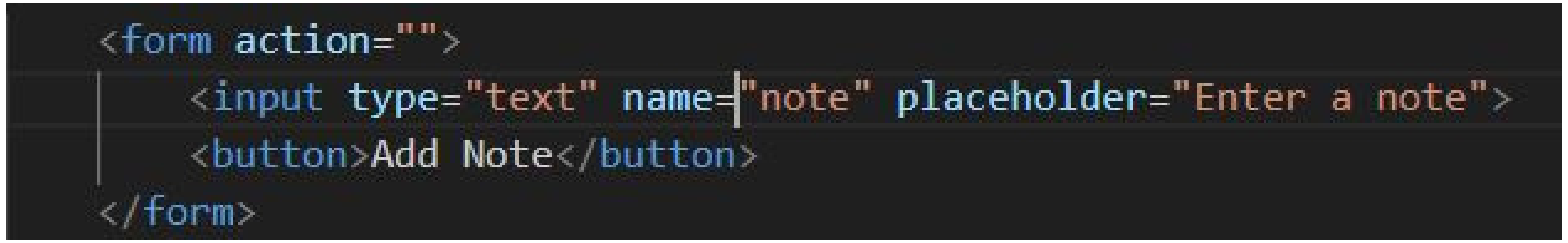
Issue 1 - Route method mismatch

- The route in the Python code is setup to only handle POST requests.
- Based on the app's functionality, we need to first be able to access the page (i.e., get data from it), but the 'GET' method has not been defined here. This leads to the method error when the app is run.



```python
notes = []
@app.route('/', methods=["POST"])
def index():
```

The HTML code contains no corresponding method, which sets it to the default method of 'GET', resulting in the data not being sent to the server correctly.

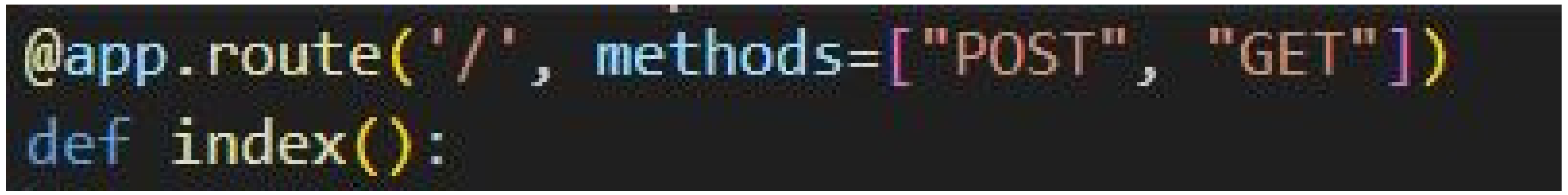


```html
<form action="">
    <input type="text" name="note" placeholder="Enter a note">
    <button>Add Note</button>
</form>
```

Bug fix

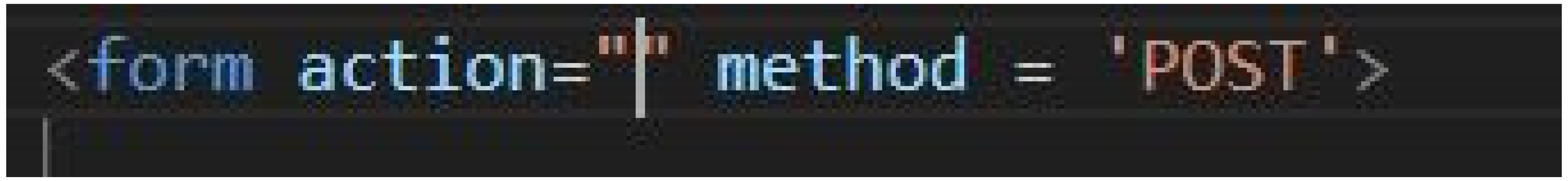
The 'POST' method is included in the HTML code, and the 'GET' method is included as well in the Python code. This resolves the 405 error.

Python



```python
@app.route('/', methods=["POST", "GET"])
def index():
```

HTML

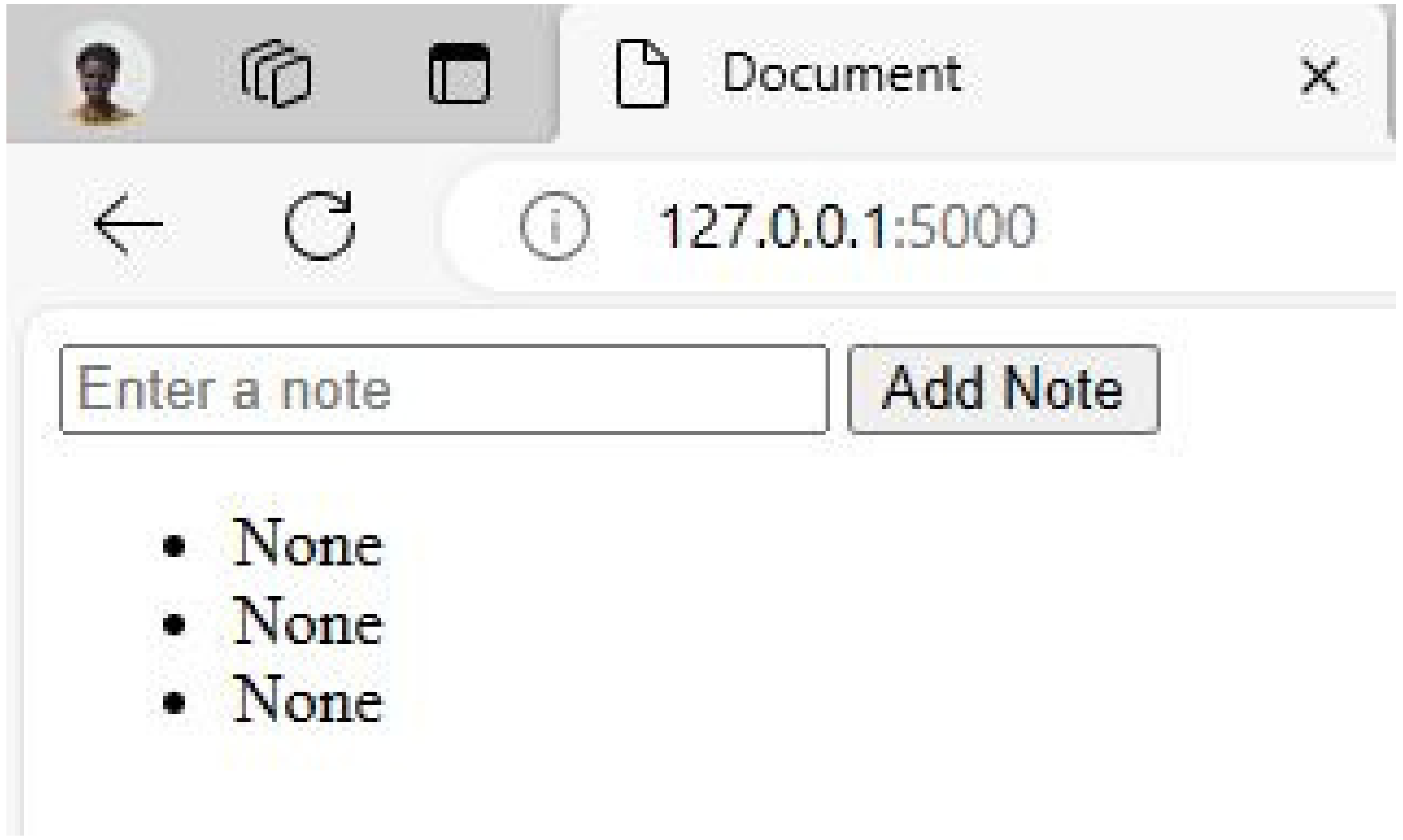


```html
<form action="" method = 'POST'>
```

Issue 2 - Data retrieval issues

Now that we have the Python code and the HTML files communicating properly, we have another issue: data retrieval.

Here, after entering a note, the user cannot see his or her notes as the application is expected to work, instead we see ‘None’.

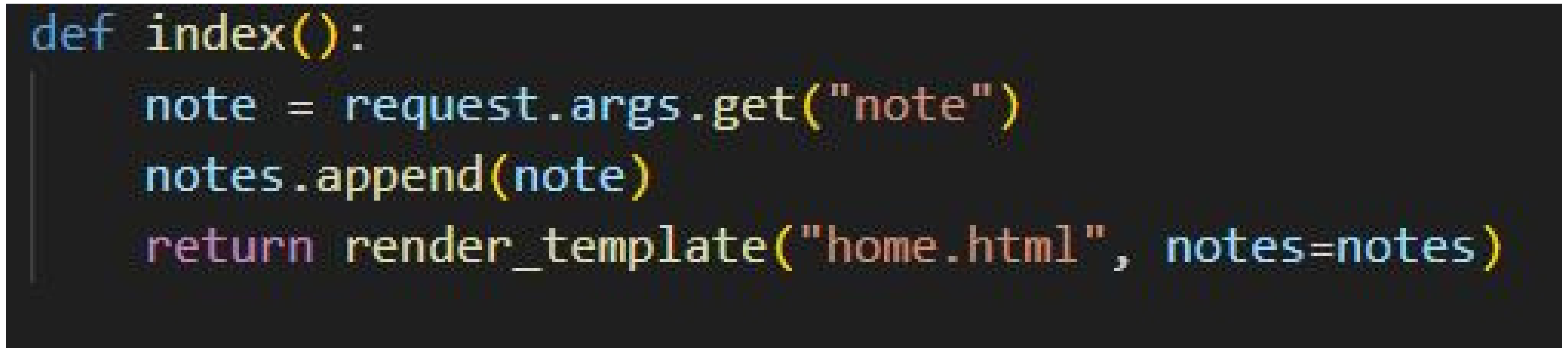


- None
- None
- None

Bug fix

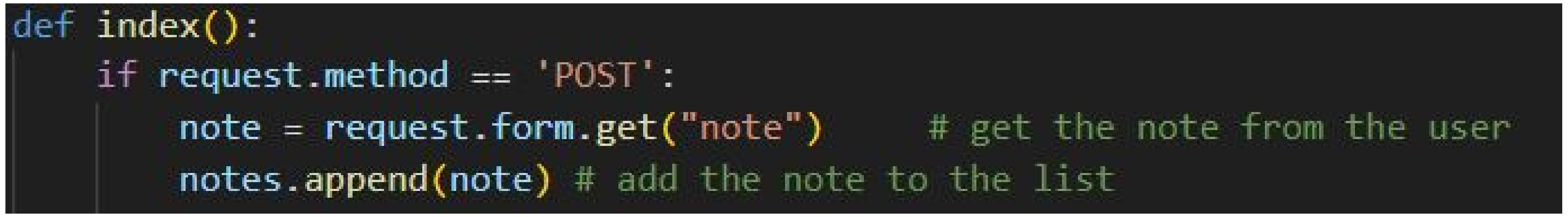
The issue here is that the Python code cannot correctly retrieve the notes already submitted via the form. This is because the note variable is stored using request.args.get("note"), which matches a GET request. We ought to use request.form.get() for POST requests, as form data is usually accessed this way to match the POST request in the HTML code.

Previous code



```
def index():
    note = request.args.get("note")
    notes.append(note)
    return render_template("home.html", notes=notes)
```

Corrected code

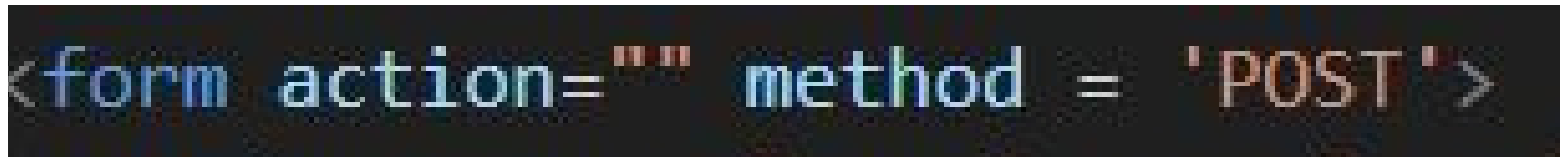


```
def index():
    if request.method == 'POST':
        note = request.form.get("note")     # get the note from the user
        notes.append(note) # add the note to the list
```

Issue 3 and bug fix - No action is specified

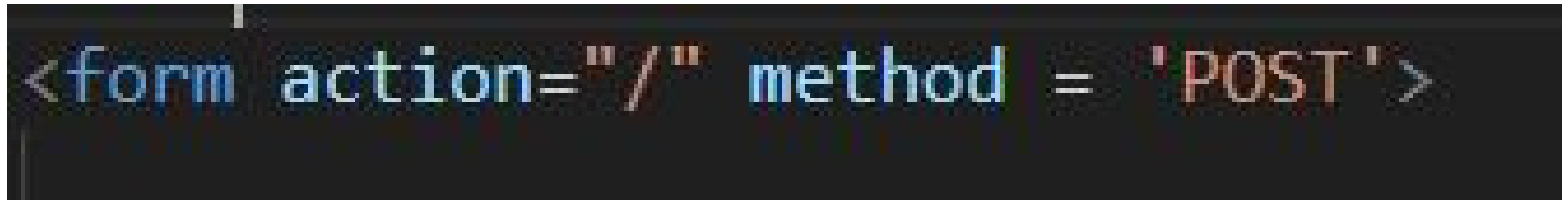
In this application, the form is intended to submit the data to the root URL (‘/’) when the user adds a new note. Since it wasn’t stated in the action attribute, the form data was not sent to the correct route (which is the homepage, so they could see previous notes they had made) for processing, and since it wasn't processed, we ended up with ‘None’ instead of the notes we entered.

Previous code



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

Corrected code



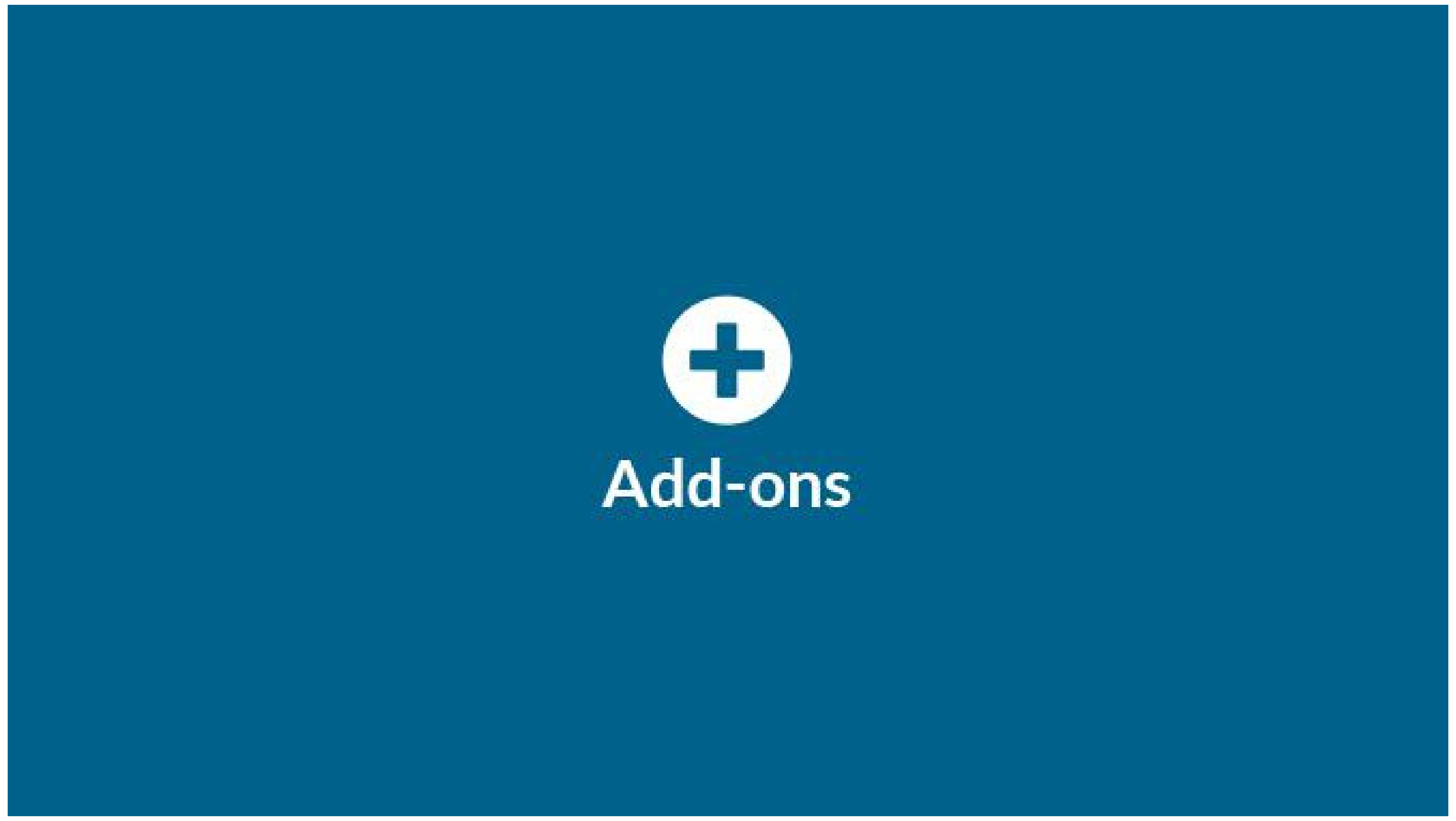
```
<form action="/" method = 'POST'>
```

Add-ons

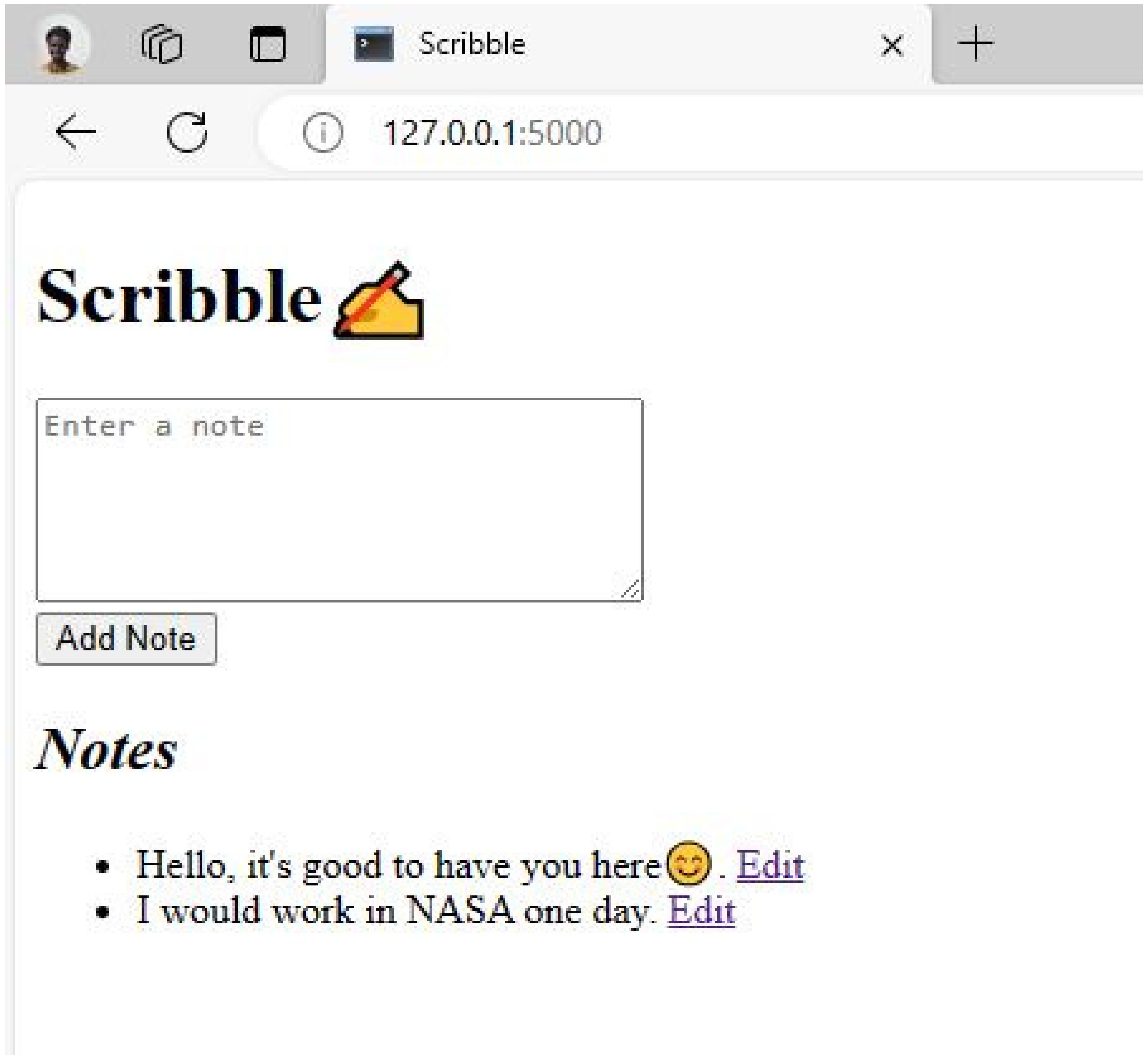
I added some features that extend the functionality of the Scribble app, namely:

- Users can edit already-created notes.

- The ‘Enter note’ field is wider to enable users to have an enhanced view of their notes as they type.



Final output


Scribble

127.0.0.1:5000

Scribble ✍️

Enter a note

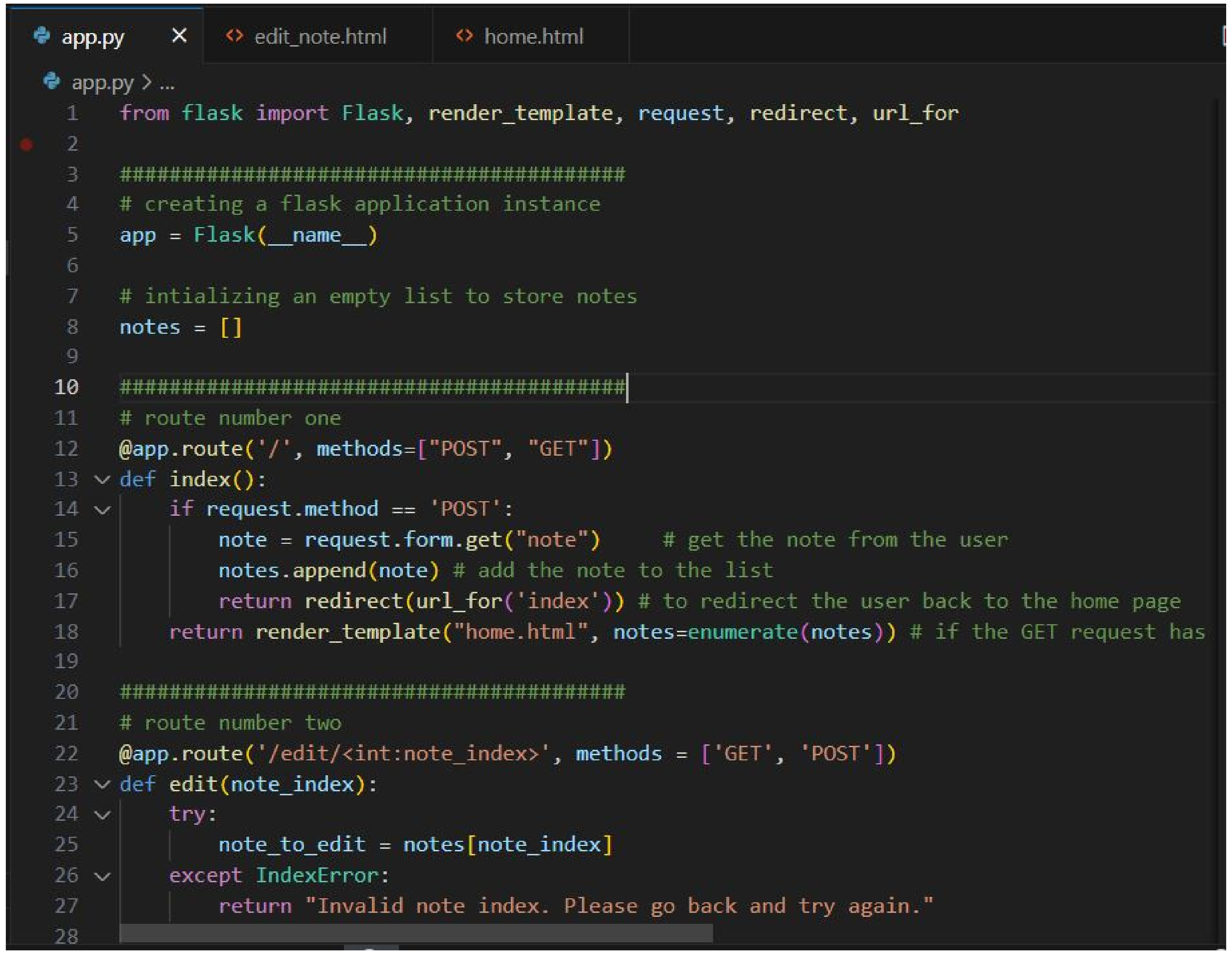
Add Note

Notes

- Hello, it's good to have you here😊. Edit
- I would work in NASA one day. Edit

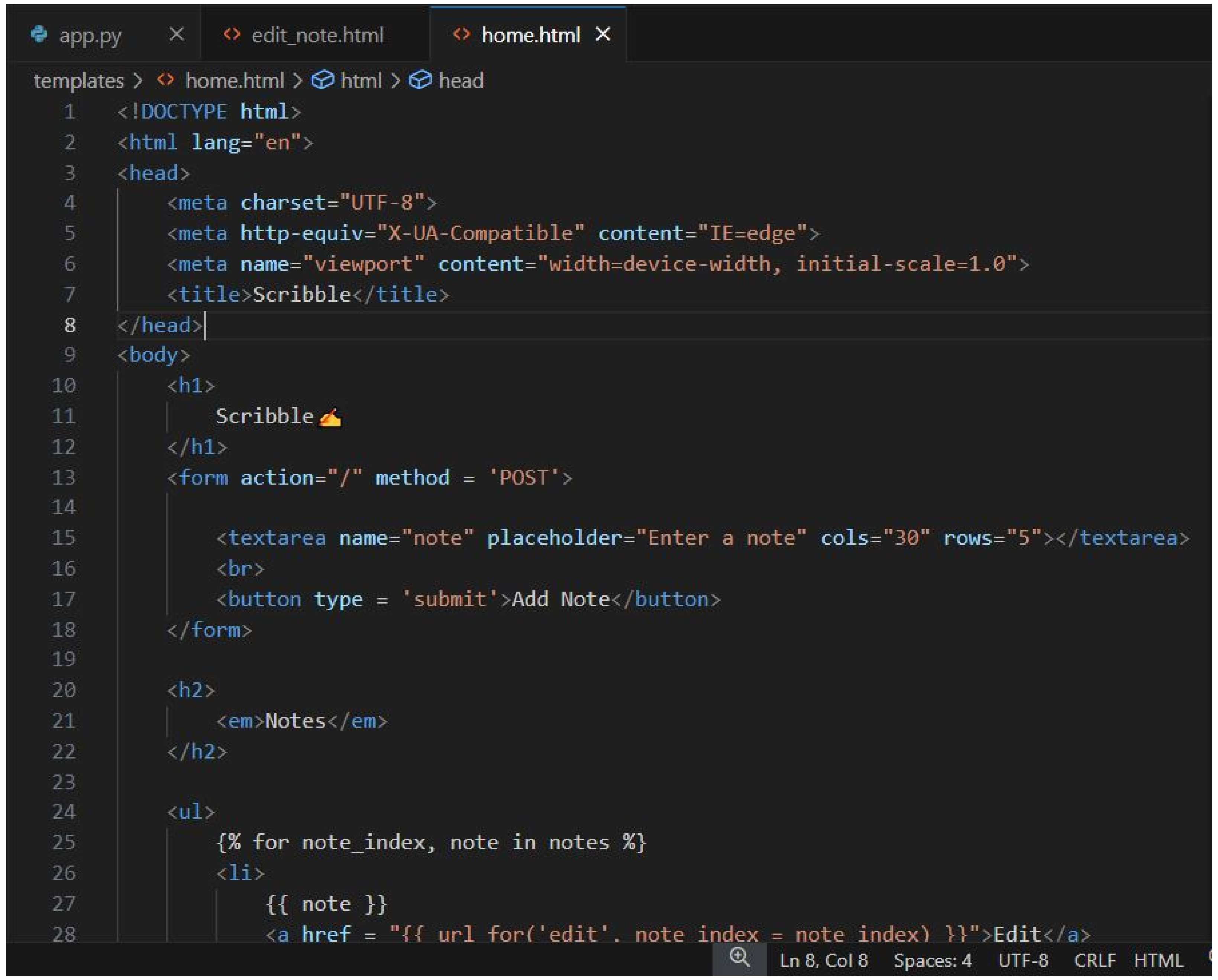
New code base 1

Python - flask codebase



```python
from flask import Flask, render_template, request, redirect, url_for

############################################
# creating a flask application instance
app = Flask(__name__)

# intializing an empty list to store notes
notes = []

############################################
# route number one
@app.route('/', methods=["POST", "GET"])
def index():
    if request.method == 'POST':
        note = request.form.get("note")      # get the note from the user
        notes.append(note) # add the note to the list
        return redirect(url_for('index')) # to redirect the user back to the home page
    return render_template("home.html", notes=enumerate(notes)) # if the GET request has

############################################
# route number two
@app.route('/edit/<int:note_index>', methods = ['GET', 'POST'])
def edit(note_index):
    try:
        note_to_edit = notes[note_index]
    except IndexError:
        return "Invalid note index. Please go back and try again."
```

Home page - html

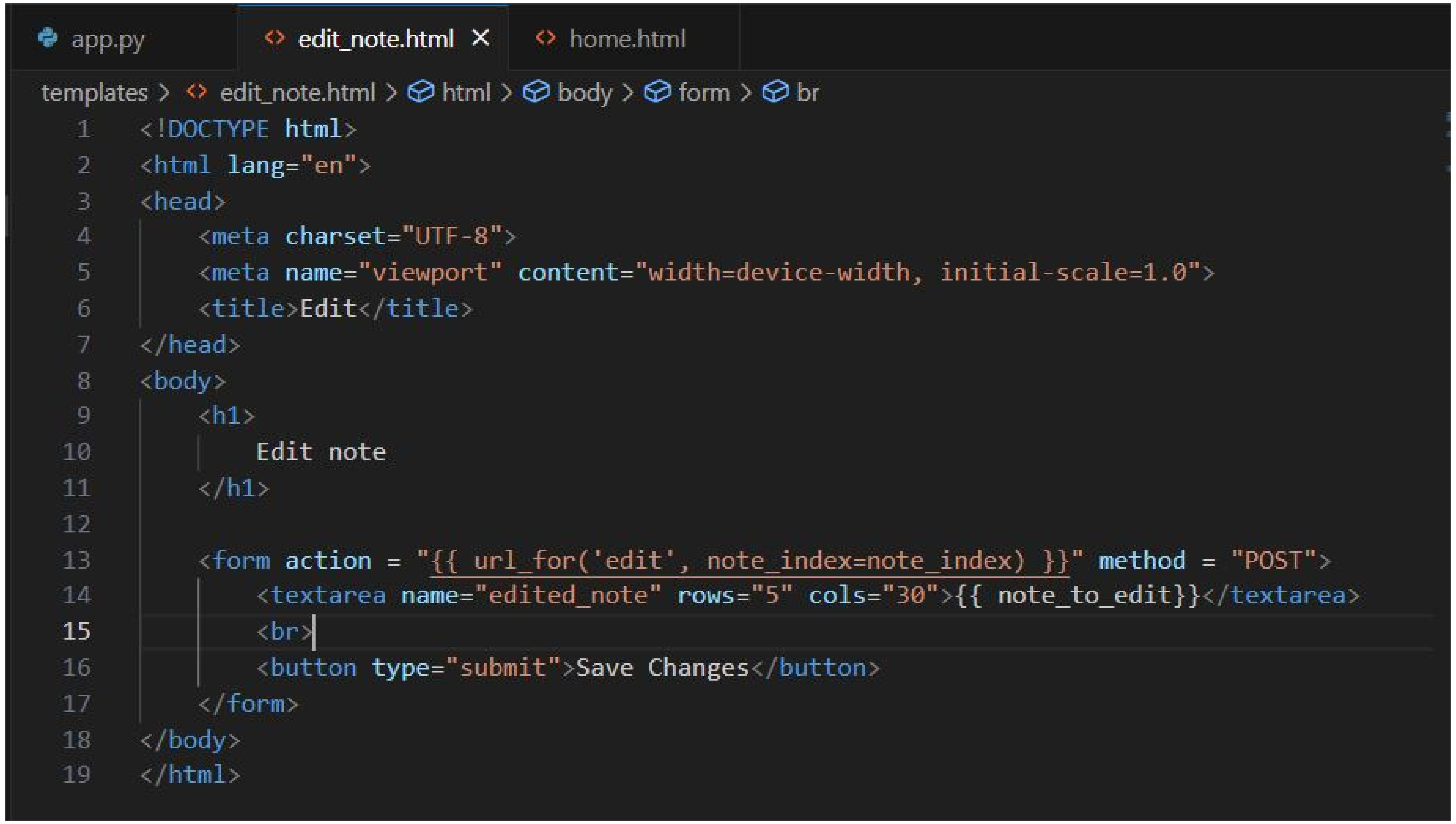

app.py
edit_note.html
home.html
templates > home.html > html > head
Ln 8, Col 8   Spaces: 4   UTF-8   CRLF   HTML

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Scribble</title>
</head>
<body>
    <h1>
        Scribble✍
    </h1>
    <form action="/" method = 'POST'>

        <textarea name="note" placeholder="Enter a note" cols="30" rows="5"></textarea>
        <br>
        <button type = 'submit'>Add Note</button>
    </form>

    <h2>
        <em>Notes</em>
    </h2>

    <ul>
        {% for note_index, note in notes %}
        <li>
            {{ note }}
            <a href = "{{ url_for('edit', note_index = note_index) }}">Edit</a>
```

INNOMATICS RESEARCH LABS

New code base 2

Edit html file



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Edit</title>
</head>
<body>
    <h1>
        Edit note
    </h1>

    <form action = "{{ url_for('edit', note_index=note_index) }}" method = "POST">
        <textarea name="edited_note" rows="5" cols="30">{{ note_to_edit}}</textarea>
        <br>
        <button type="submit">Save Changes</button>
    </form>
</body>
</html>
```

Conclusion

By refactoring the flask route, I was able to resolve the mismatch in the route methods used, include the GET method to allow us access the page, specify the needed action in the html file, and handle the data retrieval issues. The app now functions as intended. The app's features were also extended for an improved user experience.

Link to the project repository.

THANK YOU

INNOMATICS RESEARCH LABS