BOLT

BOLT IOT ARTIFICIAL INTELLIGENCE TRAINING

VANSH
B.Tech from Netaji
Subhash University
of Technology

Problem Statement 1

Project Objective:

Develop a web-based AI tool that dynamically generates educational content for any given course title. This tool will use HTML, JavaScript, and OpenAI's API to create a user-friendly interface where educators and students can input a course or subject title and receive a detailed, AI-generated outline that includes:

1. **Objective of the Course**: A concise statement that describes the purpose and goals of the course.
2. **Sample Syllabus**: An AI-generated syllabus outline that covers the main topics and modules to be taught.
3. **Three Measurable Outcomes**: Specific, measurable learning outcomes categorized according to Bloom's Taxonomy levels: Knowledge, Comprehension, and Application.
4. **Assessment Methods**: Suggestions on how to evaluate the learning outcomes through various forms of assessment.
5. **Recommended Readings and Textbooks**: A list of AI-recommended resources, including books, articles, and other materials relevant to the course content.

Detailed Requirements:

User Interface:

- A clean and simple UI with a single input box for users to enter the course title.
- A submit button to process the input.
- A single output box (or area) where the generated content will be displayed.

Functionality:

- Use JavaScript to capture the input and handle the event triggered by the submit button.
- Integrate with OpenAI's API to generate content based on the input. Ensure the requests to OpenAI are structured to produce coherent, relevant, and useful output for each of the required categories.
- Display the AI-generated content in the output box in an organized manner.

Enhancements for Better User Experience:

- Implement error handling for cases where the AI might not generate relevant or any output.
- Provide a loading indicator while the AI generates the content, as API calls can take a few seconds.
- Allow users to easily copy the generated content to the clipboard.

Considerations for Bloom's Taxonomy:

- Ensure the AI-generated learning outcomes are actionable and measurable. For instance, for "Knowledge," the outcome might involve recalling facts and basic concepts; for "Comprehension," explaining ideas or concepts; and for "Application," using the information in new situations.
- Include prompts in your API request that encourage the generation of content aligning with these cognitive levels.

Privacy and Data Handling:

- Clearly communicate to users how their input data is being used and ensure that data privacy is maintained, particularly if you plan to log requests for improvement purposes.

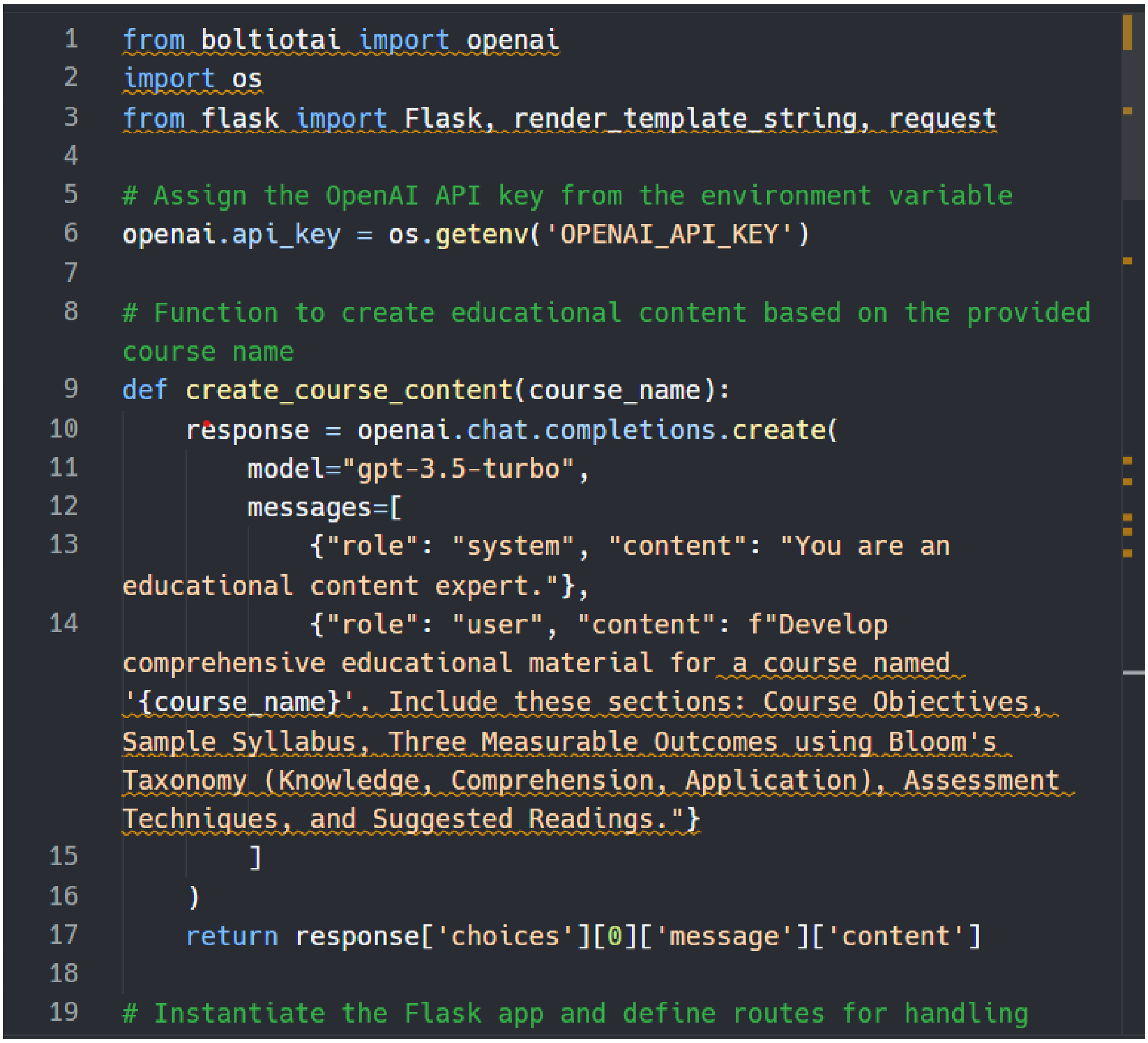
Documentation and Comments:

- Provide thorough documentation within the code to explain the logic and flow of the application.
- Include comments on how future improvements can be made or how students can experiment with different features.

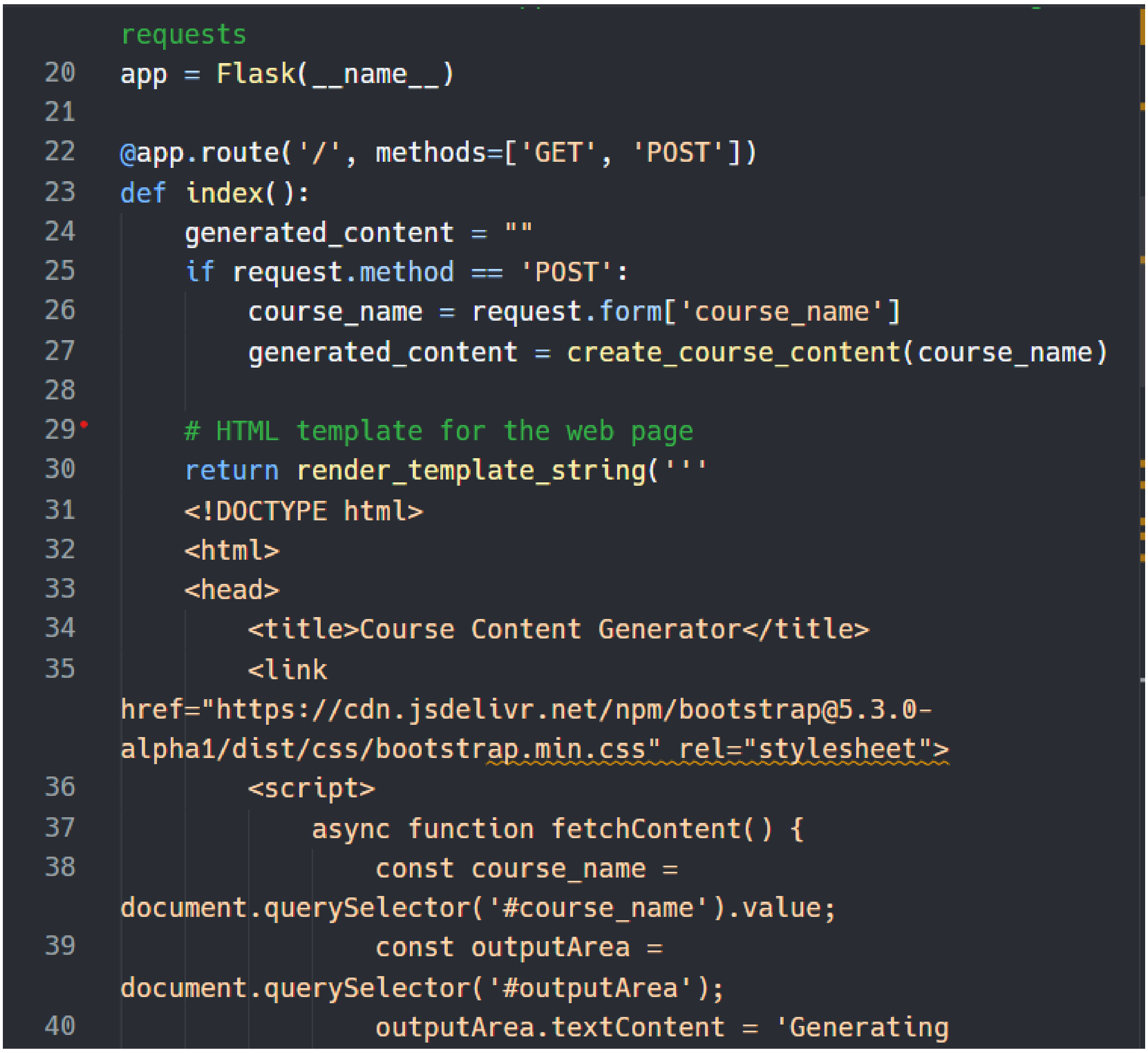
Testing and Validation:

- Encourage students to test the tool with various course titles to evaluate the effectiveness and relevance of the AI-generated content.
- Discuss the importance of critically evaluating the output for accuracy and appropriateness before use.

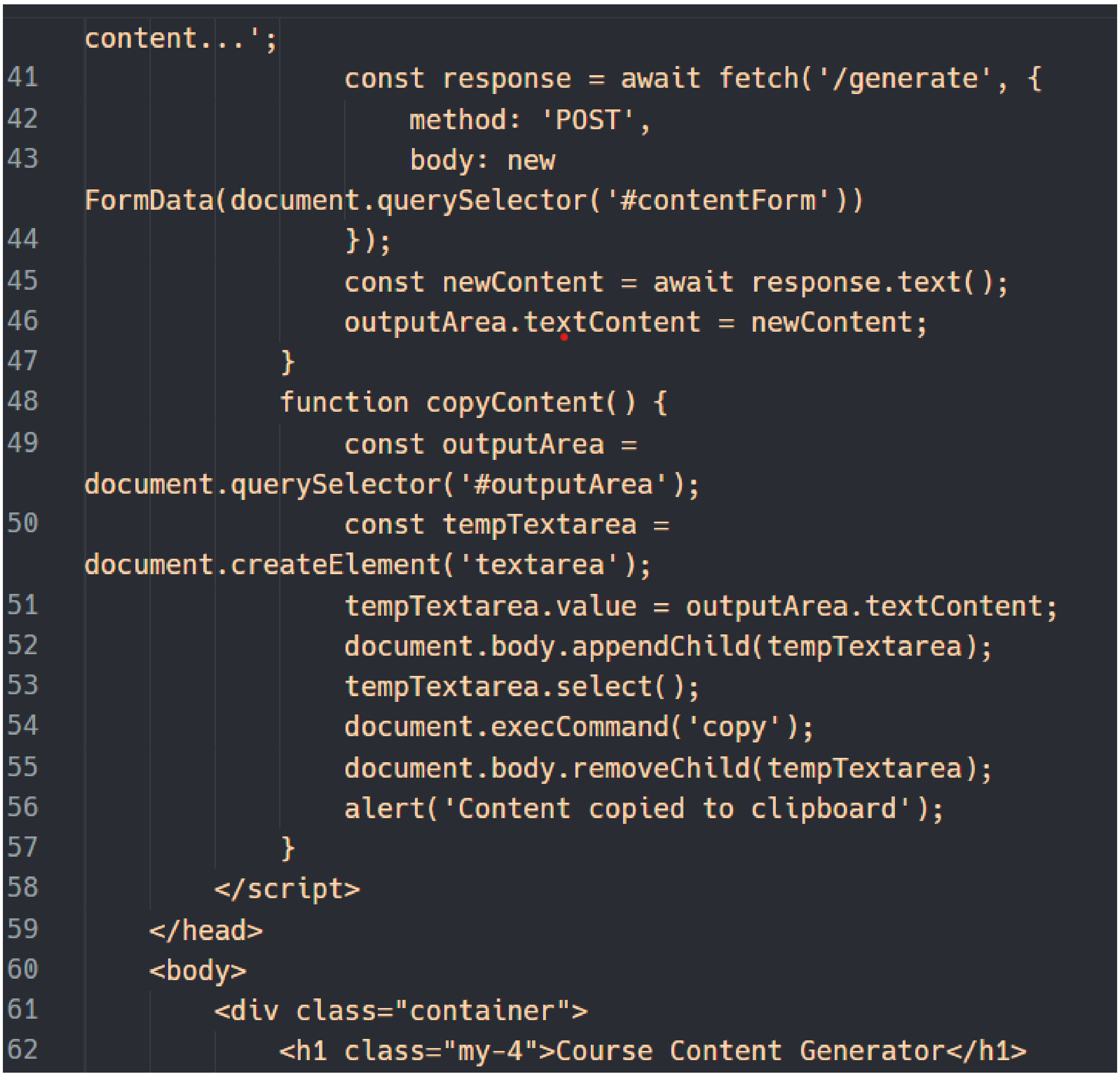
Code written on replit ide



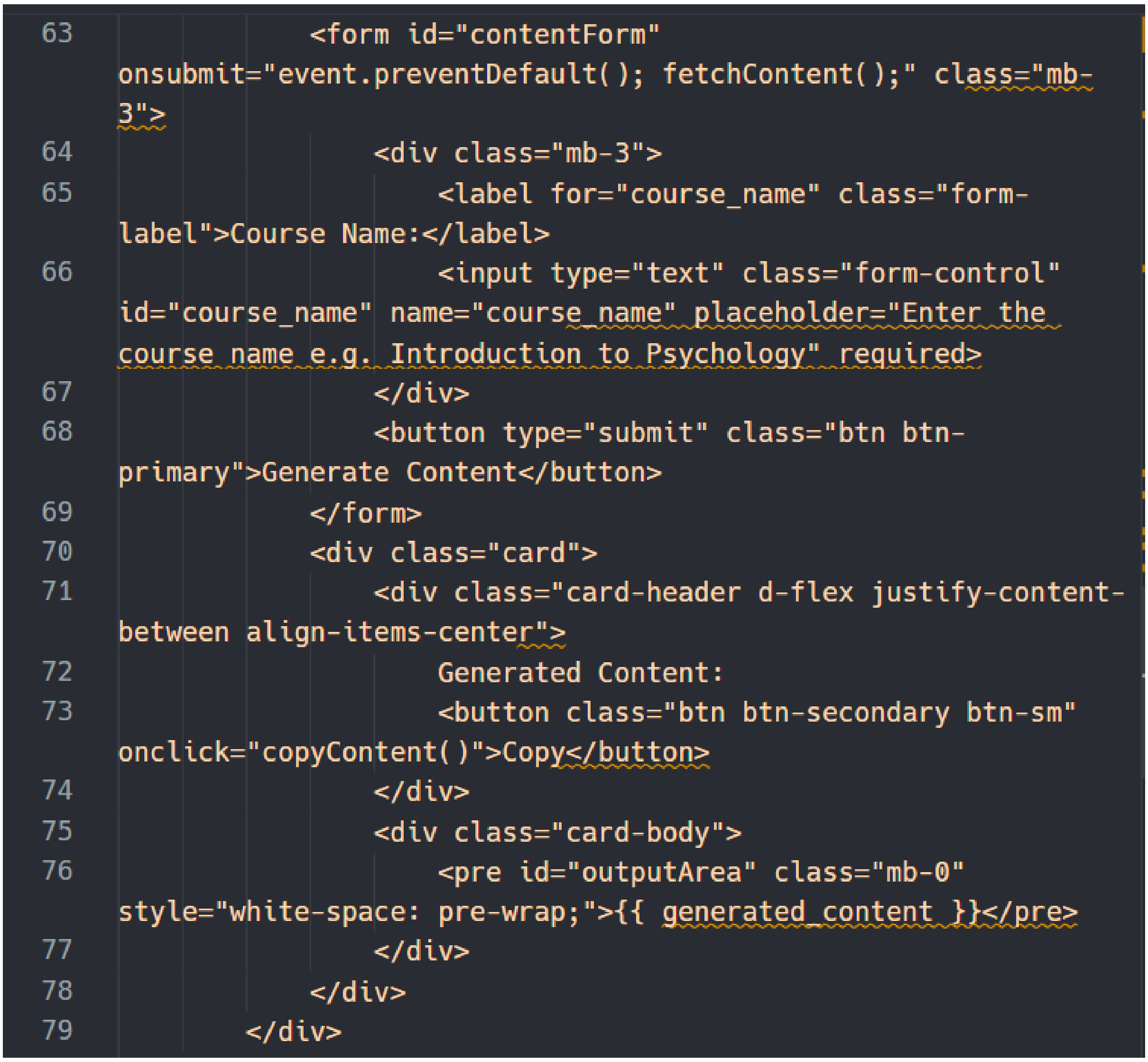
```python
from boltiotai import openai
import os
from flask import Flask, render_template_string, request

# Assign the OpenAI API key from the environment variable
openai.api_key = os.getenv('OPENAI_API_KEY')

# Function to create educational content based on the provided course name
def create_course_content(course_name):
    response = openai.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=[
            {"role": "system", "content": "You are an educational content expert."},
            {"role": "user", "content": f"Develop comprehensive educational material for a course named '{course_name}'. Include these sections: Course Objectives, Sample Syllabus, Three Measurable Outcomes using Bloom's Taxonomy (Knowledge, Comprehension, Application), Assessment Techniques, and Suggested Readings."}
        ]
    )
    return response['choices'][0]['message']['content']

# Instantiate the Flask app and define routes for handling
```



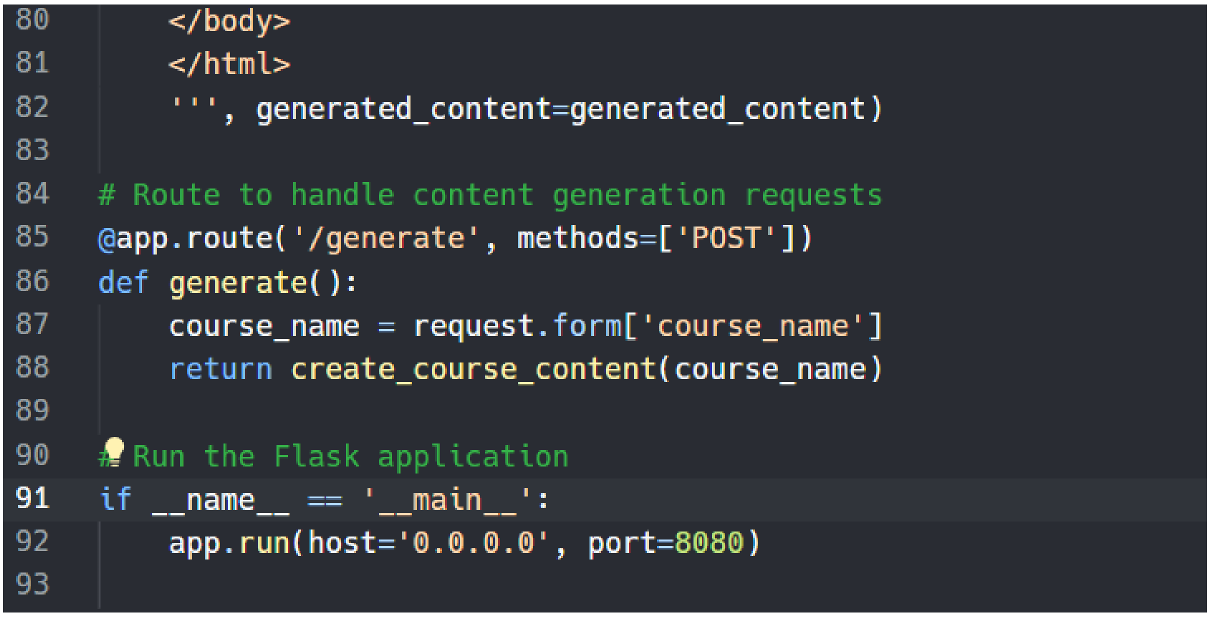
```
requests
app = Flask(__name__)

@app.route('/', methods=['GET', 'POST'])
def index():
    generated_content = ""
    if request.method == 'POST':
        course_name = request.form['course_name']
        generated_content = create_course_content(course_name)

    # HTML template for the web page
    return render_template_string('''
    <!DOCTYPE html>
    <html>
    <head>
        <title>Course Content Generator</title>
        <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
        <script>
            async function fetchContent() {
                const course_name =
document.querySelector('#course_name').value;
                const outputArea =
document.querySelector('#outputArea');
                outputArea.textContent = 'Generating
```



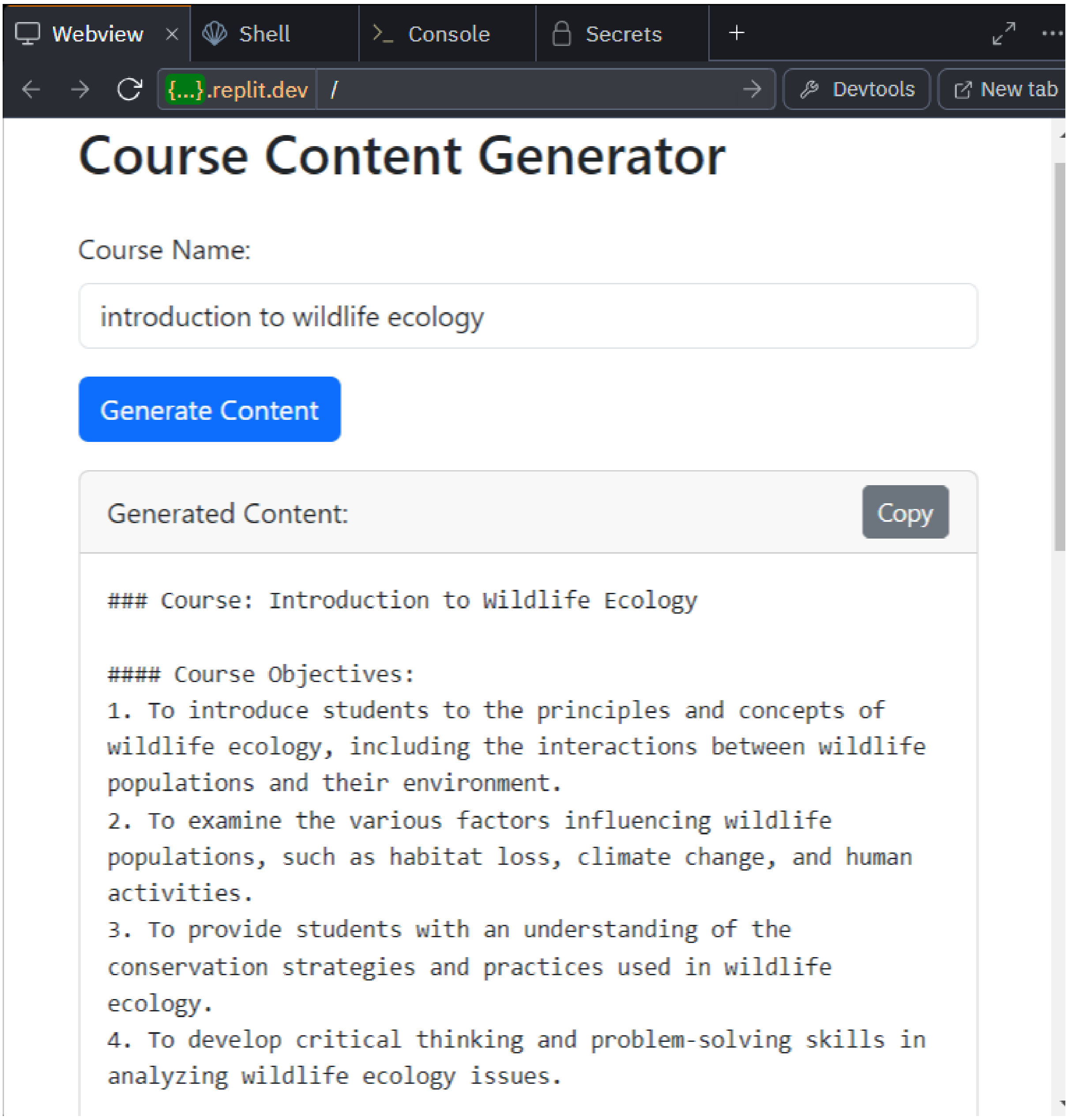
```
content...';
                const response = await fetch('/generate', {
                    method: 'POST',
                    body: new
FormData(document.querySelector('#contentForm'))
                });
                const newContent = await response.text();
                outputArea.textContent = newContent;
            }
            function copyContent() {
                const outputArea =
document.querySelector('#outputArea');
                const tempTextarea =
document.createElement('textarea');
                tempTextarea.value = outputArea.textContent;
                document.body.appendChild(tempTextarea);
                tempTextarea.select();
                document.execCommand('copy');
                document.body.removeChild(tempTextarea);
                alert('Content copied to clipboard');
            }
        </script>
    </head>
    <body>
        <div class="container">
            <h1 class="my-4">Course Content Generator</h1>
```



```
            <form id="contentForm" onsubmit="event.preventDefault(); fetchContent();" class="mb-3">
                <div class="mb-3">
                    <label for="course_name" class="form-label">Course Name:</label>
                    <input type="text" class="form-control" id="course_name" name="course_name" placeholder="Enter the course name e.g. Introduction to Psychology" required>
                </div>
                <button type="submit" class="btn btn-primary">Generate Content</button>
            </form>
            <div class="card">
                <div class="card-header d-flex justify-content-between align-items-center">
                    Generated Content:
                    <button class="btn btn-secondary btn-sm" onclick="copyContent()">Copy</button>
                </div>
                <div class="card-body">
                    <pre id="outputArea" class="mb-0" style="white-space: pre-wrap;">{{ generated_content }}</pre>
                </div>
            </div>
        </div>
```



```
    </body>
    </html>
    ''', generated_content=generated_content)

# Route to handle content generation requests
@app.route('/generate', methods=['POST'])
def generate():
    course_name = request.form['course_name']
    return create_course_content(course_name)

# Run the Flask application
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8080)

```



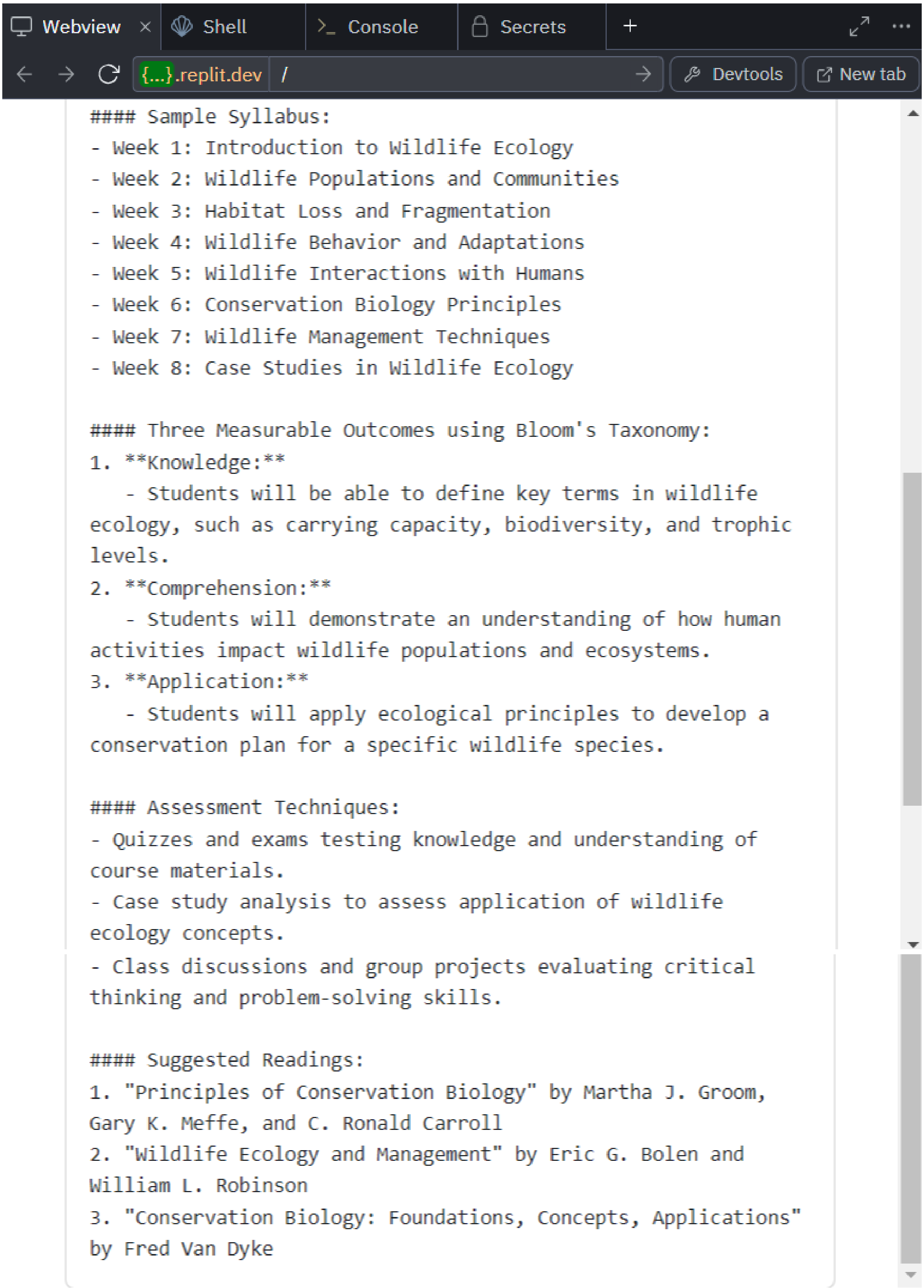
Course Content Generator

Course Name:

introduction to wildlife ecology

Generated Content:

```
### Course: Introduction to Wildlife Ecology

#### Course Objectives:
1. To introduce students to the principles and concepts of
wildlife ecology, including the interactions between wildlife
populations and their environment.
2. To examine the various factors influencing wildlife
populations, such as habitat loss, climate change, and human
activities.
3. To provide students with an understanding of the
conservation strategies and practices used in wildlife
ecology.
4. To develop critical thinking and problem-solving skills in
analyzing wildlife ecology issues.
```



```
#### Sample Syllabus:
- Week 1: Introduction to Wildlife Ecology
- Week 2: Wildlife Populations and Communities
- Week 3: Habitat Loss and Fragmentation
- Week 4: Wildlife Behavior and Adaptations
- Week 5: Wildlife Interactions with Humans
- Week 6: Conservation Biology Principles
- Week 7: Wildlife Management Techniques
- Week 8: Case Studies in Wildlife Ecology

#### Three Measurable Outcomes using Bloom's Taxonomy:
1. **Knowledge:**
   - Students will be able to define key terms in wildlife
ecology, such as carrying capacity, biodiversity, and trophic
levels.
2. **Comprehension:**
   - Students will demonstrate an understanding of how human
activities impact wildlife populations and ecosystems.
3. **Application:**
   - Students will apply ecological principles to develop a
conservation plan for a specific wildlife species.

#### Assessment Techniques:
- Quizzes and exams testing knowledge and understanding of
course materials.
- Case study analysis to assess application of wildlife
ecology concepts.
- Class discussions and group projects evaluating critical
thinking and problem-solving skills.

#### Suggested Readings:
1. "Principles of Conservation Biology" by Martha J. Groom,
Gary K. Meffe, and C. Ronald Carroll
2. "Wildlife Ecology and Management" by Eric G. Bolen and
William L. Robinson
3. "Conservation Biology: Foundations, Concepts, Applications"
by Fred Van Dyke
```