Welcome to

Python Flask: Web Development Basics to Machine Learning Deployment

By Md. Azizul Hakim

Lecturer in Daffodil International University

Debugging and Error Handling

>> Debugging and error handling are critical parts of developing Flask applications.
>> Proper handling ensures smooth development and a better user experience.

Debugging in Flask:

- Flask's debug mode (`debug=True`) helps identify issues during development by providing detailed error messages.
- Flask includes a built-in debugger with a Werkzeug Debugger.

Common Errors in Flask:

- **404 (Not Found)**: URL not found.
- **500 (Internal Server Error)**: Issues in the application logic or server-side errors.

Error Handling:

- Use custom error pages to improve user experience.
- Flask provides a way to catch specific HTTP errors using the `@app.errorhandler` decorator.

Best Practices:

- Use `try-except` blocks for sensitive operations.
- Log errors for debugging and monitoring.
- Avoid exposing sensitive information in error messages in production.

Code Example

```
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def home():
    return "Welcome to the Homepage!"

@app.errorhandler(404)
def page_not_found(error):
    return render_template('404.html'), 404

@app.errorhandler(500)
def internal_server_error(error):
    return render_template('500.html'), 500

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

Code Example

```
from flask import Flask, request
app = Flask(__name__)
@app.route('/divide', methods=['GET'])
def divide():
    try:
    num1 = float(request.args.get('num1'))
    num2 = float(request.args.get('num2'))
    result = num1 / num2
    return f"The result is {result}"
    except ZeroDivisionError:
    return "Error: Division by zero is not allowed!"
    except (TypeError, ValueError):
    return "Error: Invalid input. Please provide two numbers."
    except Exception as e:
    # Log error details
    print(f"Unexpected error: {e}")
    return "An unexpected error occurred."
if __name__ == '__main__':
    app.run(debug=True)
```

Code Example

```
import logging
from flask import Flask

app = Flask(__name__)

# Configure logging
logging.basicConfig(filename='app.log', level=logging.ERROR)

@app.route('/')
def home():
    return "Welcome to the Homepage!"

@app.route('/error')
def error_route():
    try:
    raise ValueError("An intentional error")
    except ValueError as e:
    app.logger.error(f"Error occurred: {e}")
    return "An error occurred. Check the logs for details."
if __name__ == '__main__':
    app.run(debug=True)
```

Thank You