1.2. Student Handout

Flask Introduction and Application Setup: Student Handout

Overview

This handout provides a concise guide to getting started with Flask, a popular Python web development framework. By the end of this guide, you will understand the basics of Flask, how to set up your environment, and create a simple Flask application.

What is Flask?

Flask is a **micro web framework** for Python, known for its simplicity and flexibility. It allows developers to build web applications with minimal setup and code.

Key Features of Flask:

1. **Simplicity**: Easy to learn and use.
2. **Flexibility**: Add extensions as needed.
3. **Minimal Boilerplate**: Write clean and concise code.
4. **Community Support**: Access to a large community and resources.

Setting Up Flask Environment

Step 1: Install Python

Ensure Python is installed on your system:

```
python --version
```

Download Python from python.org if not installed.

Step 2: Create a Virtual Environment

1. Open terminal/command prompt.
2. Navigate to your project folder.
3. Create a virtual environment:

```
python –m venv myenv
```

4. Activate the virtual environment:

- Windows:

```
myenv\Scripts\activate
```

- macOS/Linux:

```
source myenv/bin/activate
```

Step 3: Install Flask

Install Flask within the virtual environment:

```
pip install Flask
```

Creating Your First Flask Application

Example 1: "Hello World" Application

1. Create a file named `app.py`.

2. Add the following code:

```
from flask import Flask


app = Flask(__name__)


@app.route('/')
def hello_world():
return 'Hello, World!'


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

Example 2: Adding Another Route

Add a new route to your application:

```
@app.route('/about')
def about():
return 'This is the About page.'
```

Example 3: Custom Greeting

Create a route with a custom greeting:

```
@app.route('/greet/<name>')
def greet(name):
```

```
return f'Hello, {name}!'
```

Project Structure

Organize your Flask project as follows:

```
my_flask_app/
│
├── app.py # Main application file
├── static/ # Static files (CSS, JavaScript, images)
├── templates/ # HTML templates
└── venv/ # Virtual environment
```

Running the Flask Development Server

Run your Flask application:

```
python app.py
```

Access your app at `http://127.0.0.1:5000/`.

Understanding Routing and URL Mapping

Flask uses routing to map URLs to functions. Define routes using the `@app.route()` decorator.

Example 1: Home Route

```
@app.route('/')

def home():

return 'Welcome to the Home Page!'
```

Example 2: Contact Route

```
@app.route('/contact')

def contact():

return 'Contact us at contact@example.com.'
```

Example 3: Dynamic Route

```
@app.route('/user/<username>')

def show_user_profile(username):

return f'User: {username}'
```

Using Flask's Built-in Development Tools

- **Debugger**: Provides detailed error messages.
- **Reloader**: Automatically reloads the app on code changes.

Enable these tools with `debug=True`:

```
app.run(debug=True)
```

Conclusion

This guide covered the basics of Flask, including:

- Setting up a virtual environment and installing Flask.
- Creating a simple Flask application.
- Understanding routing and URL mapping.
- Using Flask's development tools.

With this foundation, you are ready to explore more advanced Flask topics.

Flask Application Flow Diagram

```
User Request (Browser) ---> Flask App (app.py) ---> Route (URL) --->
Function (hello_world) ---> Response (Hello, World!)
```

Feel free to reach out with any questions as you continue your Flask journey!