Q1. Explain GET and POST methods

HTTP Methods allow communication between client (browser/Postman) and server (Flask app).

1. GET Method

- Used to **request data** from the server.
- Appends parameters to the URL (`?key=value`).
- **Not secure** for sensitive data.
- Can be bookmarked.
- Example URL: `http://127.0.0.1:5000/?name=Alice`

Flask Example:

```
from flask import Flask, request

app = Flask(__name__)

@app.route('/get_example')
def get_example():
    name = request.args.get('name')  # Retrieve query parameter
    return f"Hello {name}!"

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

---

2. POST Method

- Used to **send data to the server** (like forms).
- Data is sent in the **request body**, not URL.
- **More secure** for sensitive data.

- Cannot be bookmarked.

Flask Example:

```
from flask import Flask, request

app = Flask(__name__)

@app.route('/post_example', methods=['POST'])
def post_example():
    name = request.form['name']
    return f"Hello {name}!"

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

Q2. Why is `request` used in Flask?

- `request` is used to **access incoming request data** in Flask.
- Provides access to:
    - Query parameters (`request.args`)
    - Form data (`request.form`)
    - JSON data (`request.json`)
    - Headers (`request.headers`)

Example:

```
from flask import Flask, request

app = Flask(__name__)

@app.route('/data', methods=['GET', 'POST'])
```

```
def data():
    if request.method == 'POST':
        return f"POST data: {request.form}"
    else:
        return f"GET data: {request.args}"

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

Q3. Why is `redirect()` used in Flask?

- `redirect()` is used to **redirect the user to a different URL**.
- Useful for **after form submission** or when moving to another page.

Example:

```
from flask import Flask, redirect, url_for

app = Flask(__name__)

@app.route('/')
def home():
    return "Home Page"

@app.route('/go-to-home')
def go_to_home():
    return redirect(url_for('home'))

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

- Accessing `/go-to-home` will redirect the user to `/`.

Q4. What are templates in Flask? Why use `render_template()`?

Templates:

- HTML files that can include **dynamic content**.
- Stored in the `templates/` folder of the Flask project.

`render_template()`:

- Renders an HTML file and passes variables from Python to HTML.

Example:

```
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def home():
    user = "Alice"
    return render_template('home.html', username=user)

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

home.html (in `templates/` folder):

```
<!DOCTYPE html>
<html>
<head>
    <title>Home Page</title>
</head>
<body>
    <h1>Welcome, {{ username }}!</h1>
</body>
</html>
```

- Output: `Welcome, Alice!`

Q5. Create a simple API and test with Postman

Flask API Example (returns JSON data):

```
from flask import Flask, jsonify, request

app = Flask(__name__)

# Sample API route
@app.route('/api/greet', methods=['GET', 'POST'])
def greet():
    if request.method == 'POST':
        data = request.json
        name = data.get('name', 'Guest')
        return jsonify({"message": f"Hello {name}!"})
    else:
        return jsonify({"message": "Send a POST request with your
name."})

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

Steps to test in Postman:

1. Start Flask app (`python filename.py`).
2. Open Postman → Create new POST request → URL:
   `http://127.0.0.1:5000/api/greet`
3. Set **Body** → **raw** → **JSON**:

```
{
```

```
    "name": "Alice"
}
```

4. Send request → Response:

```
{
    "message": "Hello Alice!"
}
```

5. Take a **screenshot** of Postman response for Jupyter Notebook.

I can also create a **ready-to-run Jupyter Notebook** with all these examples, including:

- GET & POST examp