St. Francis Institute of Technology Borivali (West), Mumbai-400103

(Autonomous Institute)
Department of Information Technology

Academic Year: 2024-25

Class: TE-ITA/B   Semester: VI

Subject: Web Lab

Experiment –8: To Design Feedback Form using Flask.

1. **Aim:** To learn Flask Framework.
2. **Objectives:** Aim of this experiment is that, the students will be able
    - To install Flask Framework
    - To understand Basics of Flask.
    - To understand Flask Application
3. **Outcomes:** After study of this experiment, the students will be able
    - To build applications.
    - To build URL
    - To understand HTTP methods.
4. **Prerequisite:** Basic understanding of HTML and Python etc
5. **Requirements:** Personal Computer, Windows operating system, VSCode, Python 2.6 or higher, browser, Internet Connection, google doc, latest version of Python.
6. **Pre-Experiment Exercise:**
   Brief Theory: Refer shared material
7. **Laboratory Exercise**
    A. **Procedure:**
        Install Python 3 on local machine
        Set up a programming environment via the command line
        Installactivate Python environment
        Install Flask using the pip package installer
        a. **Answer the following:**
            - List What is Web Framework? What is Flask?
            - Write and explain Methods of Mail class?
        b. **Attach screenshots:**
            - Flask SS

8. **Post-Experiments Exercise**
    A. **Extended Theory:**
        Nil
    B. **Questions:**
        - Why is Flask called a Microframework?
        - What is the difference between Django and Flask?
    C. **Conclusion:**
        - Write what was performed in the experiment.
        - Write the significance of the topic studied in the experiment.
    D. **References:**
        1. Flask Web Development, by Miguel Grinberg

Answer the following:

List What is Web Framework? What is Flask?

A **web framework** is a software tool that provides a structure and set of tools for building web applications. It simplifies the process of developing web services, APIs, and websites by handling common tasks such as:

- URL routing1
- Request and response handling
- Session management
- Templating (rendering HTML pages)
- Integration with databases
- Form handling and validation

Examples of popular web frameworks:

- Python: Flask, Django
- JavaScript: Express.js, Next.js
- PHP: Laravel
- Java: Spring

Flask is a lightweight and flexible web framework written in Python. It's known for being simple to use and is often used for small to medium-sized web applications and APIs.

Key features of Flask:

- Microframework: Minimal core, but highly extensible
- Built-in development server and debugger
- RESTful request dispatching
- Integrated support for unit testing
- Uses Jinja2 templating engine
- Supports secure cookies (client-side sessions)
- Easily extended with Flask extensions (e.g., Flask-Mail, Flask-Login)
- Flask is ideal for those who want more control and flexibility without the complexity of a larger framework like Django.

Write and explain Methods of Mail class?

Flask-Mail is an extension for Flask that makes it easy to send emails from your Flask applications. The main class provided is Mail, and here are some commonly used methods and attributes related to it:

a. Mail.init_app(app)

Binds the Flask app to the Mail instance.

Used when creating the Mail object separately from the Flask app.

```
mail = Mail()
mail.init_app(app)
```

b. Mail.send(message)

Sends an email message.

Takes a Message object as a parameter.

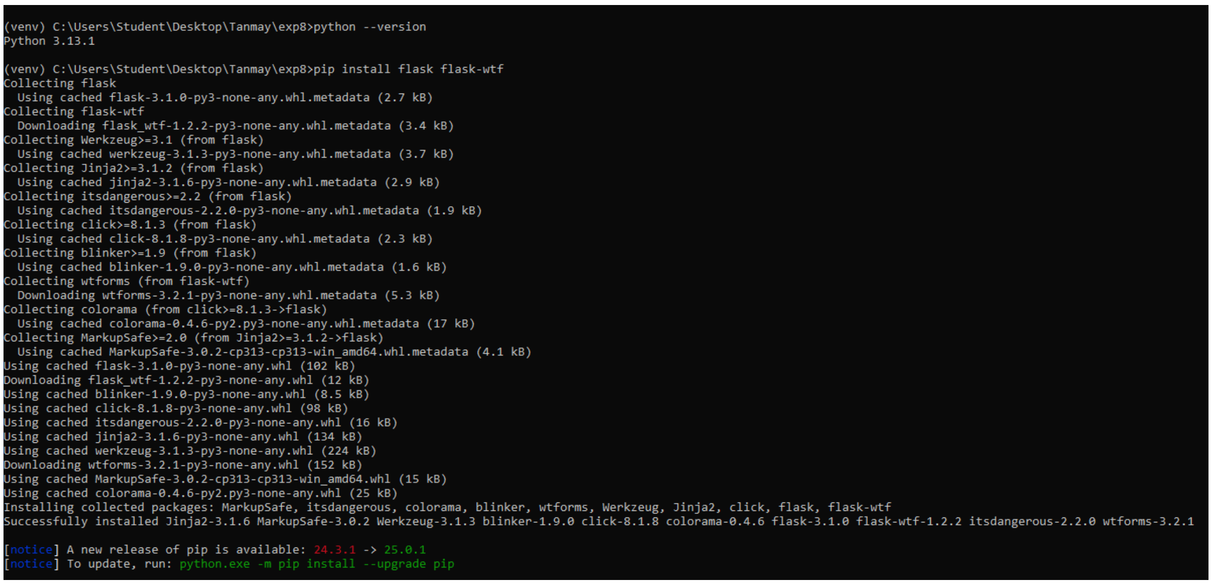
```
from flask_mail import Message
msg = Message('Hello', sender='your@example.com', recipients=['someone@example.com'])
msg.body = 'This is the email body.'
mail.send(msg)
```

Message class (used with Mail)
To send an email, you typically use the Message class.
Common attributes of Message:

- subject: Subject line of the email
- recipients: List of recipient email addresses
- body: Plain text content of the email
- html: (Optional) HTML content of the email
- sender: Email address of the sender
- cc, bcc: CC and BCC recipients

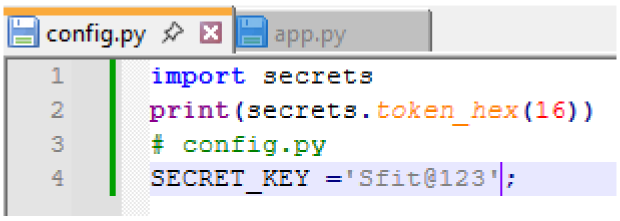
```
msg = Message(
    subject="Welcome!",
    sender="admin@example.com",
    recipients=["user@example.com"],
    body="Thanks for signing up!"
)
mail.send(msg)
```

LAB EXERCISE:



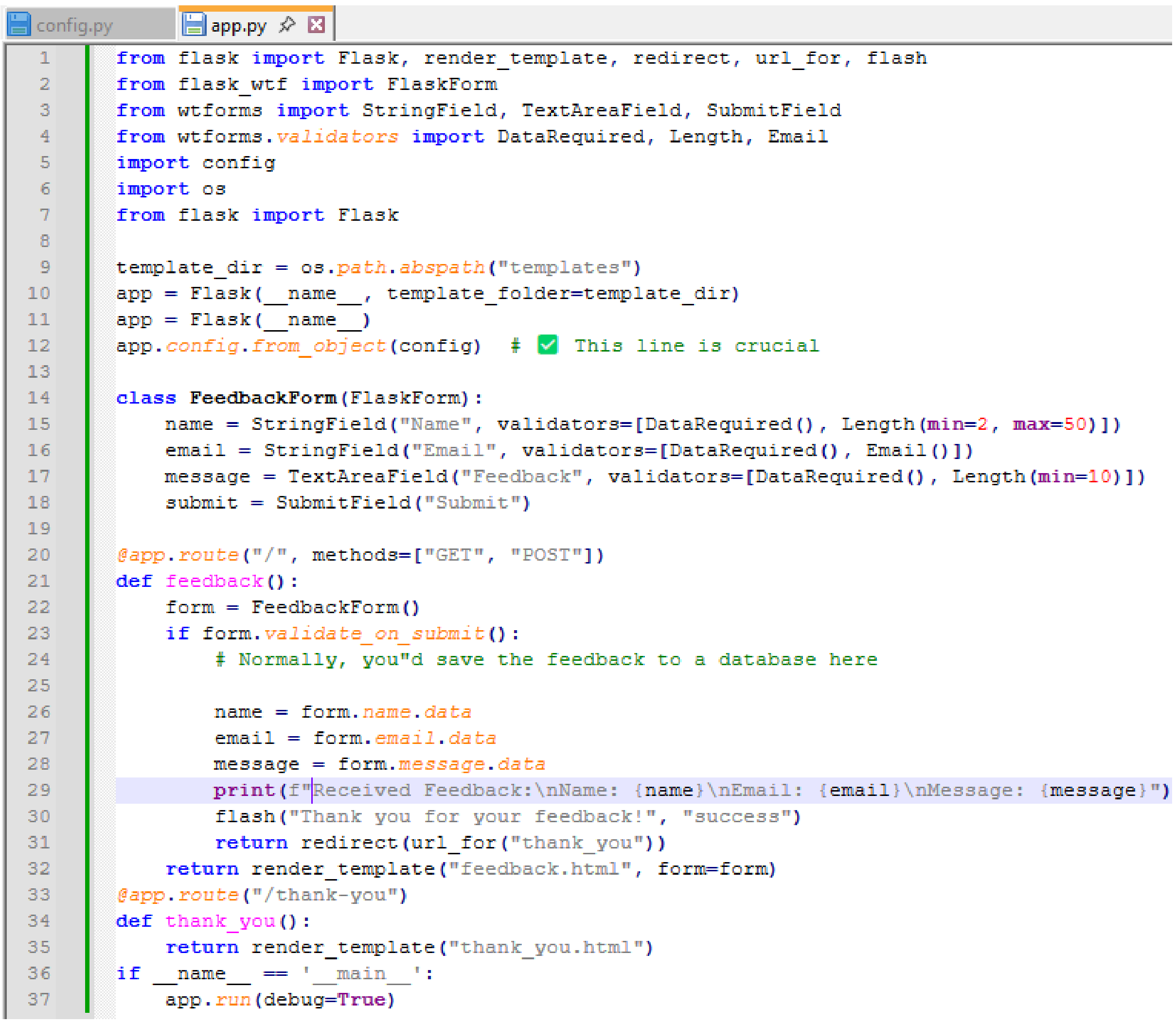
```
(venv) C:\Users\Student\Desktop\Tanmay\exp8>python --version
Python 3.13.1

(venv) C:\Users\Student\Desktop\Tanmay\exp8>pip install flask flask-wtf
Collecting flask
  Using cached flask-3.1.0-py3-none-any.whl.metadata (2.7 kB)
Collecting flask-wtf
  Downloading flask_wtf-1.2.2-py3-none-any.whl.metadata (3.4 kB)
Collecting Werkzeug>=3.1 (from flask)
  Using cached werkzeug-3.1.3-py3-none-any.whl.metadata (3.7 kB)
Collecting Jinja2>=3.1.2 (from flask)
  Using cached jinja2-3.1.6-py3-none-any.whl.metadata (2.9 kB)
Collecting itsdangerous>=2.2 (from flask)
  Using cached itsdangerous-2.2.0-py3-none-any.whl.metadata (1.9 kB)
Collecting click>=8.1.3 (from flask)
  Using cached click-8.1.8-py3-none-any.whl.metadata (2.3 kB)
Collecting blinker>=1.9 (from flask)
  Using cached blinker-1.9.0-py3-none-any.whl.metadata (1.6 kB)
Collecting wtforms (from flask-wtf)
  Downloading wtforms-3.2.1-py3-none-any.whl.metadata (5.3 kB)
Collecting colorama (from click>=8.1.3->flask)
  Using cached colorama-0.4.6-py2.py3-none-any.whl.metadata (17 kB)
Collecting MarkupSafe>=2.0 (from Jinja2>=3.1.2->flask)
  Using cached MarkupSafe-3.0.2-cp313-cp313-win_amd64.whl.metadata (4.1 kB)
Using cached flask-3.1.0-py3-none-any.whl (102 kB)
Downloading flask_wtf-1.2.2-py3-none-any.whl (12 kB)
Using cached blinker-1.9.0-py3-none-any.whl (8.5 kB)
Using cached click-8.1.8-py3-none-any.whl (98 kB)
Using cached itsdangerous-2.2.0-py3-none-any.whl (16 kB)
Using cached jinja2-3.1.6-py3-none-any.whl (134 kB)
Using cached werkzeug-3.1.3-py3-none-any.whl (224 kB)
Downloading wtforms-3.2.1-py3-none-any.whl (152 kB)
Using cached MarkupSafe-3.0.2-cp313-cp313-win_amd64.whl (15 kB)
Using cached colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Installing collected packages: MarkupSafe, itsdangerous, colorama, blinker, wtforms, Werkzeug, Jinja2, click, flask, flask-wtf
Successfully installed Jinja2-3.1.6 MarkupSafe-3.0.2 Werkzeug-3.1.3 blinker-1.9.0 click-8.1.8 colorama-0.4.6 flask-3.1.0 flask-wtf-1.2.2 itsdangerous-2.2.0 wtforms-3.2.1

[notice] A new release of pip is available: 24.3.1 -> 25.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

INSTALLING FLASK



```
import secrets
print(secrets.token_hex(16))
# config.py
SECRET_KEY ='Sfit@123';
```

Config.py



```python
from flask import Flask, render_template, redirect, url_for, flash
from flask_wtf import FlaskForm
from wtforms import StringField, TextAreaField, SubmitField
from wtforms.validators import DataRequired, Length, Email
import config
import os
from flask import Flask

template_dir = os.path.abspath("templates")
app = Flask(__name__, template_folder=template_dir)
app = Flask(__name__)
app.config.from_object(config)  # ✅ This line is crucial

class FeedbackForm(FlaskForm):
    name = StringField("Name", validators=[DataRequired(), Length(min=2, max=50)])
    email = StringField("Email", validators=[DataRequired(), Email()])
    message = TextAreaField("Feedback", validators=[DataRequired(), Length(min=10)])
    submit = SubmitField("Submit")

@app.route("/", methods=["GET", "POST"])
def feedback():
    form = FeedbackForm()
    if form.validate_on_submit():
        # Normally, you"d save the feedback to a database here

        name = form.name.data
        email = form.email.data
        message = form.message.data
        print(f"Received Feedback:\nName: {name}\nEmail: {email}\nMessage: {message}")
        flash("Thank you for your feedback!", "success")
        return redirect(url_for("thank_you"))
    return render_template("feedback.html", form=form)
@app.route("/thank-you")
def thank_you():
    return render_template("thank_you.html")
if __name__ == '__main__':
    app.run(debug=True)
```

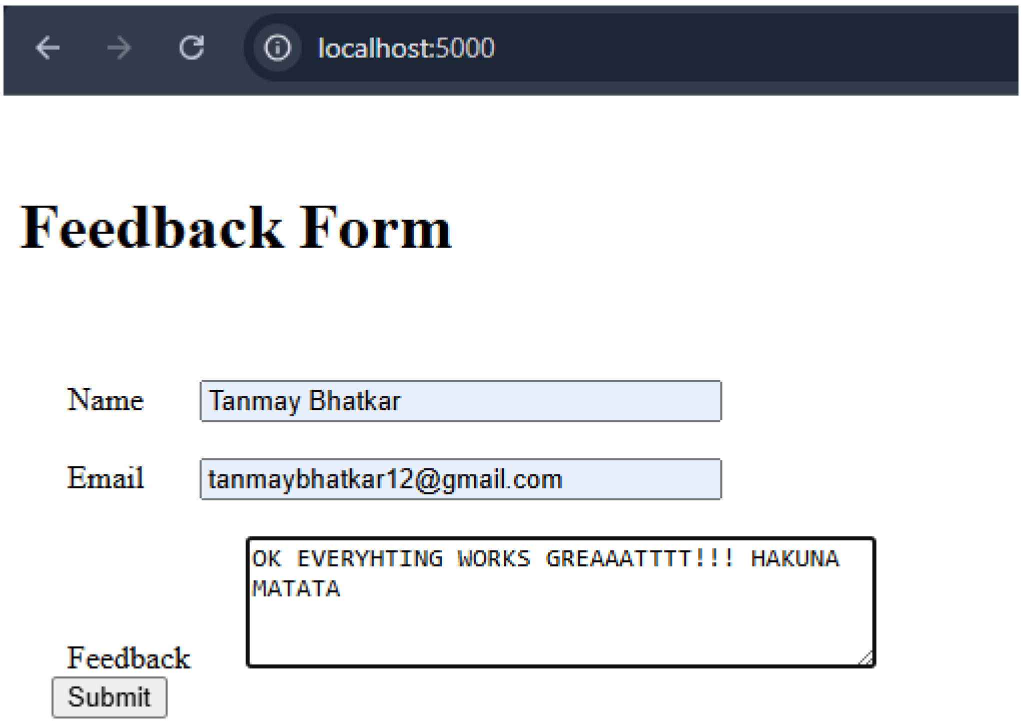
app.py

templates/feedback.html

```html
<!doctype html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Feedback Form</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      margin: 2rem;
    }
    .form-field {
      margin-bottom: 1rem;
    }
    .form-field label {
      display: block;
      font-weight: bold;
    }
    .form-field input,
    .form-field textarea {
      width: 100%;
      padding: 0.5rem;
    }
    .error {

      color: red;
      font-size: 0.9em;
    }
  </style>
</head>
<body>
  <h1>Feedback Form</h1>
  <form method="POST">
    {{ form.hidden_tag() }}
    <div class="form-field">
      {{ form.name.label }}
      {{ form.name(size=32) }}
      {% for error in form.name.errors %}
        <div class="error">[{{ error }}]</div>
      {% endfor %}
    </div>
    <div class="form-field">
      {{ form.email.label }}
      {{ form.email(size=32) }}
      {% for error in form.email.errors %}
        <div class="error">[{{ error }}]</div>
      {% endfor %}
    </div>
    <div class="form-field">
      {{ form.message.label }}
      {{ form.message(rows=4, cols=40) }}
      {% for error in form.message.errors %}
        <div class="error">[{{ error }}]</div>
      {% endfor %}
    </div>
    {{ form.submit() }}
  </form>
</body>
</html>
```

templates/Thank_you.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Feedback Form</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      margin: 2rem;
    }
    .form-field {
      margin-bottom: 1rem;
    }
    .form-field label {
      display: block;
      font-weight: bold;
    }
    .form-field input,
    .form-field textarea {
      width: 100%;
      padding: 0.5rem;
    }
    .error {

      color: red;
      font-size: 0.9em;
    }
  </style>
</head>
<body>
  <h1>Feedback Form</h1>
  <form method="POST">
    {{ form.hidden_tag() }}
    <div class="form-field">
      {{ form.name.label }}
      {{ form.name(size=32) }}
      {% for error in form.name.errors %}
        <div class="error">[{{ error }}]</div>
      {% endfor %}
    </div>
    <div class="form-field">
      {{ form.email.label }}
      {{ form.email(size=32) }}
      {% for error in form.email.errors %}
        <div class="error">[{{ error }}]</div>
      {% endfor %}
    </div>
    <div class="form-field">
      {{ form.message.label }}
      {{ form.message(rows=4, cols=40) }}
      {% for error in form.message.errors %}
        <div class="error">[{{ error }}]</div>
      {% endfor %}
    </div>
    {{ form.submit() }}
  </form>
</body></html>
```

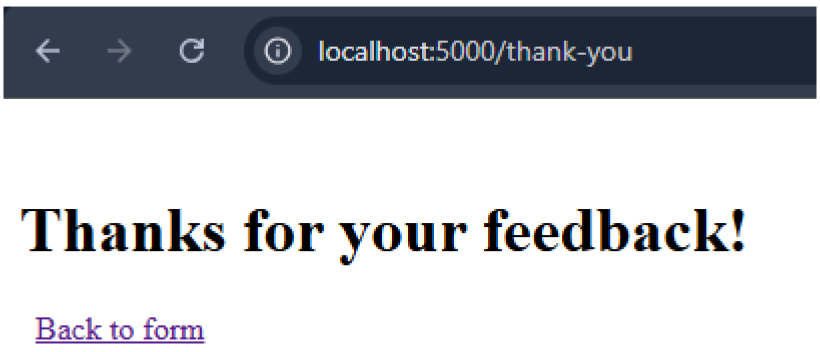


localhost:5000

Feedback Form

Name	Tanmay Bhatkar
Email	tanmaybhatkar12@gmail.com
Feedback	OK EVERYHTING WORKS GREAAATTTT!!! HAKUNA MATATA

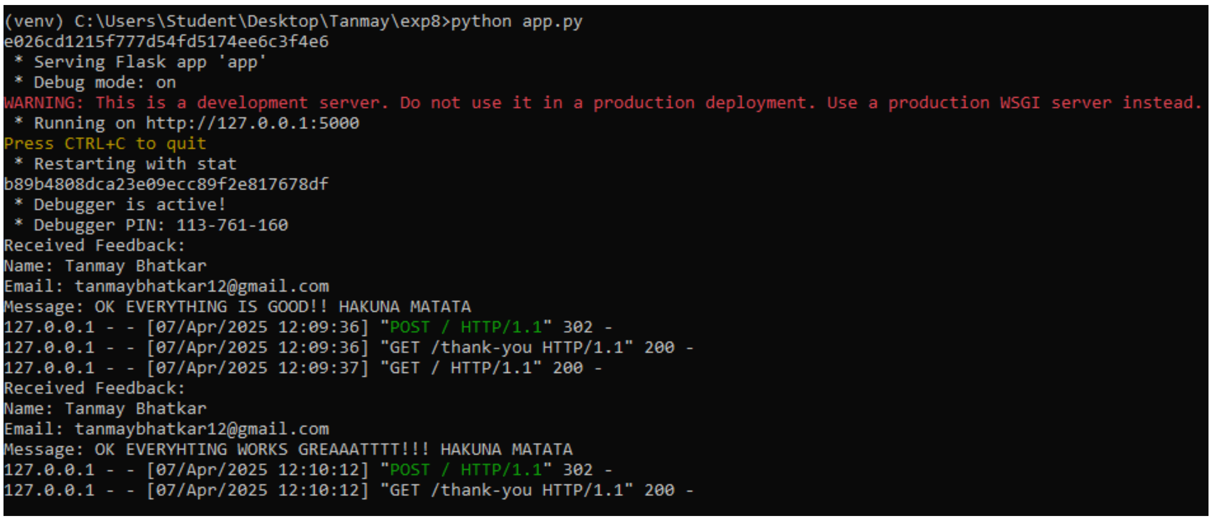
Submit



localhost:5000/thank-you

Thanks for your feedback!

Back to form



```
(venv) C:\Users\Student\Desktop\Tanmay\exp8>python app.py
e026cd1215f777d54fd5174ee6c3f4e6
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
b89b4808dca23e09ecc89f2e817678df
 * Debugger is active!
 * Debugger PIN: 113-761-160
Received Feedback:
Name: Tanmay Bhatkar
Email: tanmaybhatkar12@gmail.com
Message: OK EVERYTHING IS GOOD!! HAKUNA MATATA
127.0.0.1 - - [07/Apr/2025 12:09:36] "POST / HTTP/1.1" 302 -
127.0.0.1 - - [07/Apr/2025 12:09:36] "GET /thank-you HTTP/1.1" 200 -
127.0.0.1 - - [07/Apr/2025 12:09:37] "GET / HTTP/1.1" 200 -
Received Feedback:
Name: Tanmay Bhatkar
Email: tanmaybhatkar12@gmail.com
Message: OK EVERYHTING WORKS GREAAATTTT!!! HAKUNA MATATA
127.0.0.1 - - [07/Apr/2025 12:10:12] "POST / HTTP/1.1" 302 -
127.0.0.1 - - [07/Apr/2025 12:10:12] "GET /thank-you HTTP/1.1" 200 -
```