

플라스크 웹 프로그램

Flash 메시지

유효성 검증(밸리데이션) 추가하기

- Flash 메시지는 동작 실행 후에 간단한 메시지를 표시하는 기능
- 완료 시나 오류 발생 시 등 일시적으로 메시지를 표시할 때 이용
- 문의 품 화면에 유효성 검증 (입력 체크 처리) 을 추가
- 입력 체크 시 오류가 있으면 Flash 메시지를 사용하여 한 번만 오류 정보를 표시
- Flash 메시지는 flash 함수를 사용하여 설정하고, 템플릿에서 get_flashed_messages 함수를 사용
- Flash 메시지를 이용하려면 세션이 필요하므로 config의 SECRET_KEY 를 설정

Flash 메시지

유효성 검증(밸리데이션) 추가하기

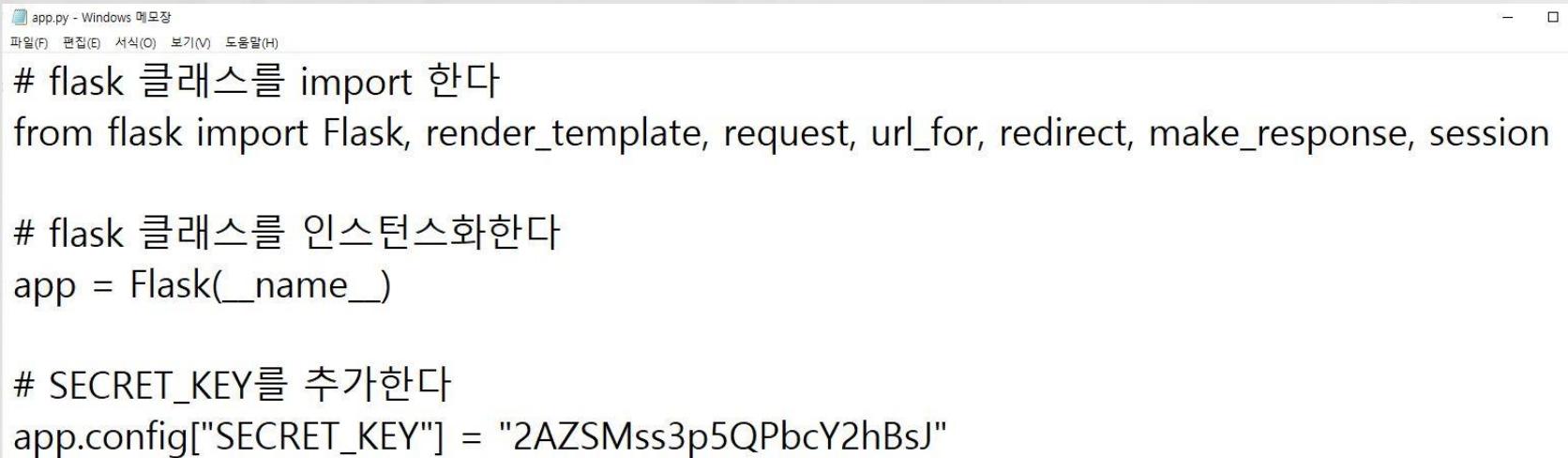
- config는 앱을 이용하는 데 필요한 설정
- config에 값을 설정하려면

```
app.config["config_key"] = config_value
```

Flash 메시지

SECRET_KEY 설정하기

- 세션을 사용하려면 세션 정보 보안을 위해 비밀 키 (SECRET_KEY)를 설정
- 비밀 키는 랜덤한 값으로 해야 함
- C:\flaskbook\apps\minimalapp\app.py 수정



app.py - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
# flask 클래스를 import 한다
from flask import Flask, render_template, request, url_for, redirect, make_response, session

# flask 클래스를 인스턴스화한다
app = Flask(__name__)

# SECRET_KEY를 추가한다
app.config["SECRET_KEY"] = "2AZSMss3p5QPbcY2hBsJ"
```

Flash 메시지

POST 값의 입력 체크를 추가

- 이메일 주소(email)가 올바른 형식인지 여부를 체크하기 위해서 email-validator 패키지를 설치
- **pip install email-validator**

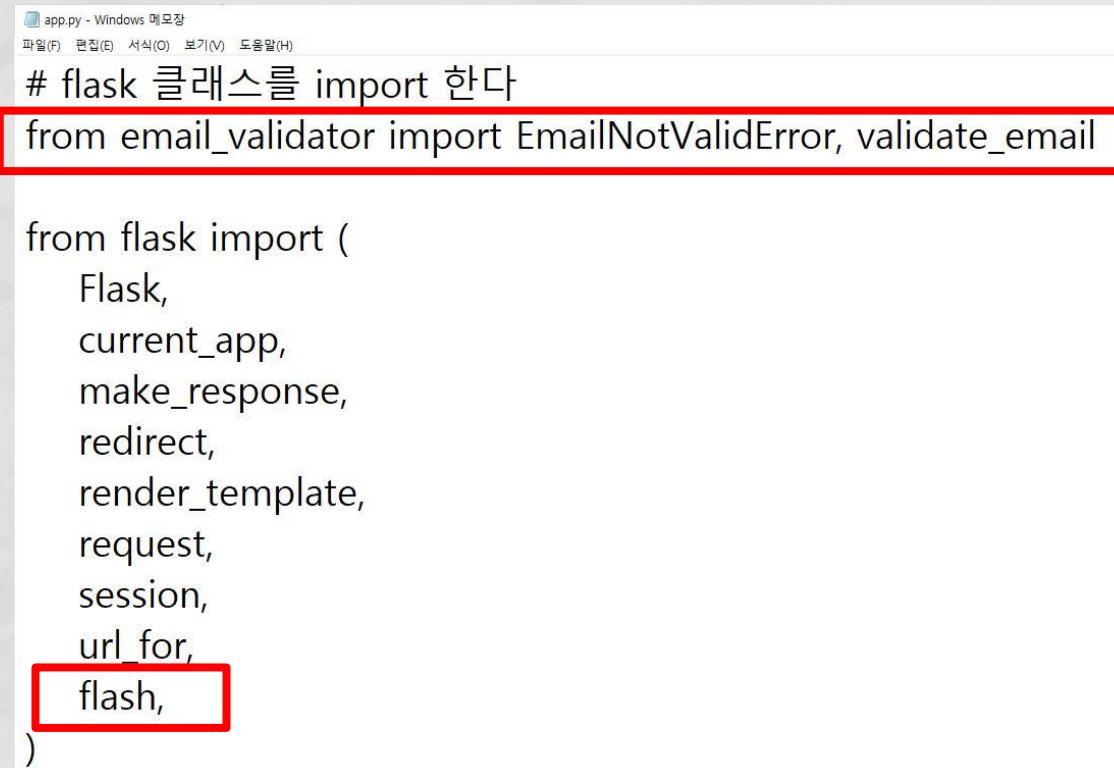


```
관리자: Anaconda Prompt
(base) C:\flaskbook\apps\minimalapp>pip install email-validator
Collecting email-validator
  Downloading email_validator-2.2.0-py3-none-any.whl.metadata (25 kB)
Collecting dnspython>=2.0.0 (from email-validator)
  Downloading dnspython-2.7.0-py3-none-any.whl.metadata (5.8 kB)
Requirement already satisfied: idna>=2.0.0 in c:\programdata\anaconda3\lib\site-packages (from email-validator) (3.7)
Downloaded email_validator-2.2.0-py3-none-any.whl (33 kB)
Downloaded dnspython-2.7.0-py3-none-any.whl (313 kB)
----- 313.6/313.6 kB 6.5 MB/s eta 0:00:00
Installing collected packages: dnspython, email-validator
Successfully installed dnspython-2.7.0 email-validator-2.2.0
(base) C:\flaskbook\apps\minimalapp>
```

Flash 메시지

POST 값의 입력 체크를 추가

- C:\flaskbook\apps\minimalapp\app.py의 import 추가 및 contact_complete 갱신



The screenshot shows a Windows Notepad window titled "app.py - Windows 메모장". The code is as follows:

```
# flask 클래스를 import 한다
from email_validator import EmailNotValidError, validate_email

from flask import (
    Flask,
    current_app,
    make_response,
    redirect,
    render_template,
    request,
    session,
    url_for,
    flash,
)
```

The line "from email_validator import EmailNotValidError, validate_email" is highlighted with a red rectangle. The word "flash" in the final line of the imports is also highlighted with a red rectangle.

Flash 메시지

POST 값의 입력 체크를 추가

- C:\flaskbook\apps\minimalapp\app.py의 import 추가 및 contact_complete 갱신

```
app.py - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

@app.route("/contact/complete", methods=["GET", "POST"])
def contact_complete():
    if request.method == "POST":
        # form 속성을 사용해서 폼의 값을 취득한다
        username = request.form["username"]
        email = request.form["email"]
        description = request.form["description"]

        # 입력 체크
        is_valid = True
        if not username:
            flash("사용자명은 필수입니다")
            is_valid = False

        if not email:
            flash("메일 주소는 필수입니다")
            is_valid = False
```

```
try:
    validate_email(email)
except EmailNotValidError:
    flash("메일 주소의 형식으로 입력해 주세요")
    is_valid = False

if not description:
    flash("문의 내용은 필수입니다")
    is_valid = False

if not is_valid:
    return redirect(url_for("contact"))

# 문의 완료 엔드포인트로 리다이렉트한다
flash("문의 내용은 메일로 송신했습니다. 문의해 주셔서 감사합니다.")

return redirect(url_for("contact_complete"))
return render_template("contact_complete.html")
```

Flash 메시지

POST 값의 입력 체크를 추가

- 이메일 주소 형식 체크용 validate_email과 EmailNotValidError를 import
- flash 를 추가로 import
- 사용자명, 이메일 주소, 문의 내용 입력란이 비어 있으면 flash에 오류 메시지 를 설정
- email이 이메일 주소의 형식인지 여부를 validate_email 함수로 확인
- 형식이 정확하지 않은 경우는 예외가 발생하므로 try-except로 사용
- POST 값에 문제가 없는 경우는 Flash 메시지에 “문의해 주셔서 감사합니다.” 를 설정
- 문의 완료 화면으로 리다이렉트

Flash 메시지

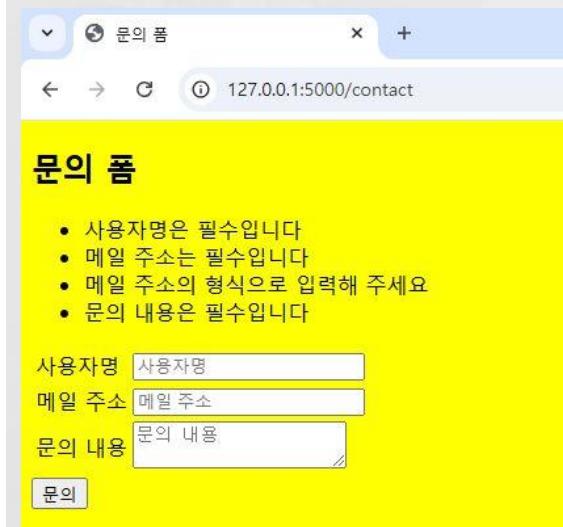
POST 값의 입력 체크를 추가

- C:\flaskbook\apps\minimalapp\templates\contact.html 의
`get_flashed_messages` 함수 추가

```
contact.html - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

<body>
  <h2>문의 폼</h2>

  {% with messages = get_flashed_messages() %}
  {% if messages %}
    <ul>
      {% for message in messages %}
        <li class="flash">{{ message }}</li>
      {% endfor %}
    </ul>
  {% endif %}
  {% endwith %}
```



Flash 메시지

POST 값의 입력 체크를 추가

- C:\flaskbook\apps\minimalapp\templates\contact_complete.html

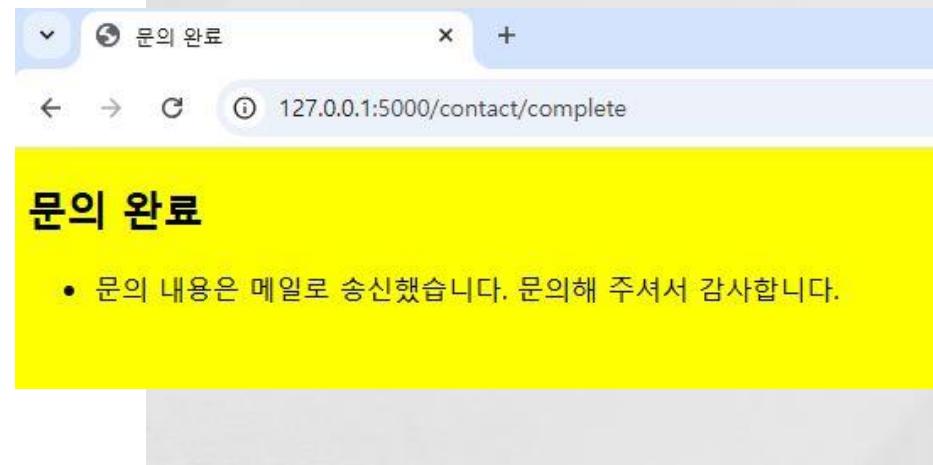
에도 flash 메시지 추가

```
contact_complete.html - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

<body>
    <h2>문의 완료</h2>

    {% with messages = get_flashed_messages() %}
    {% if messages %}
        <ul>
            {% for message in messages %}
                <li>{{ message }}</li>
            {% endfor %}
        </ul>
    {% endif %}
    {% endwith %}

</body>
```

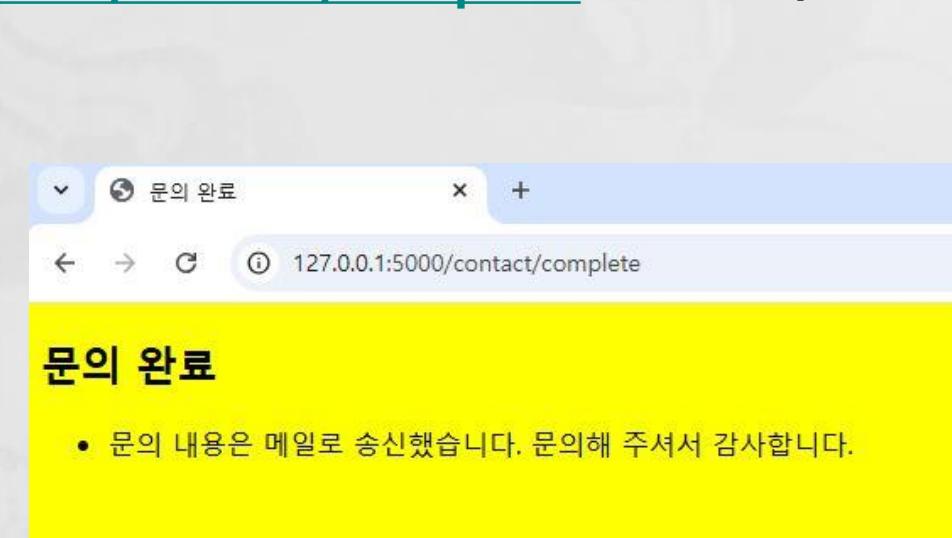


Flash 메시지

Flask 실행 및 확인

- (base) C:\flaskbook\apps\minimalapp>flask run
- 브라우저에서 <http://127.0.0.1:5000/contact> URL에 접근
- 브라우저에서 <http://127.0.0.1:5000/contact/complete> URL에 접근

A screenshot of a web browser window. The title bar says "문의 품". The address bar shows the URL "127.0.0.1:5000/contact". The main content area has a yellow background and displays the following text:
문의 품
• 사용자명은 필수입니다
• 메일 주소는 필수입니다
• 메일 주소의 형식으로 입력해 주세요
• 문의 내용은 필수입니다
Below this, there are three input fields:
- 사용자명: (labeled "사용자명")
- 메일 주소: (labeled "메일 주소")
- 문의 내용: (labeled "문의 내용")
At the bottom is a single button:
문의



이메일 보내기

Flask-mail 설치

- 이메일을 보내기 위해서 플라스크의 flask-mail 확장을 설치
- pip install flask-mail

```
관리자: Anaconda Prompt

(base) C:\flaskbook\apps\minimalapp>pip install flask-mail
Collecting flask-mail
  Downloading flask_mail-0.10.0-py3-none-any.whl.metadata (2.1 kB)
Requirement already satisfied: flask in c:\programdata\anaconda3\lib\site-packages (from flask-mail)
Requirement already satisfied: blinker in c:\programdata\anaconda3\lib\site-packages (from flask-mail)
Requirement already satisfied: Werkzeug>=3.0.0 in c:\programdata\anaconda3\lib\site-packages (from flask-mail)
Requirement already satisfied: Jinja2>=3.1.2 in c:\programdata\anaconda3\lib\site-packages (from flask-mail)
Requirement already satisfied: itsdangerous>=2.1.2 in c:\programdata\anaconda3\lib\site-packages (from flask-mail)
Requirement already satisfied: click>=8.1.3 in c:\programdata\anaconda3\lib\site-packages (from flask-mail)
Requirement already satisfied: colorama in c:\programdata\anaconda3\lib\site-packages (from flask-mail)
Requirement already satisfied: MarkupSafe>=2.0 in c:\programdata\anaconda3\lib\site-packages (from flask-mail)
  Downloading flask_mail-0.10.0-py3-none-any.whl (8.5 kB)
Installing collected packages: flask-mail
Successfully installed flask-mail-0.10.0

(base) C:\flaskbook\apps\minimalapp>
```

이메일 보내기

config 설정

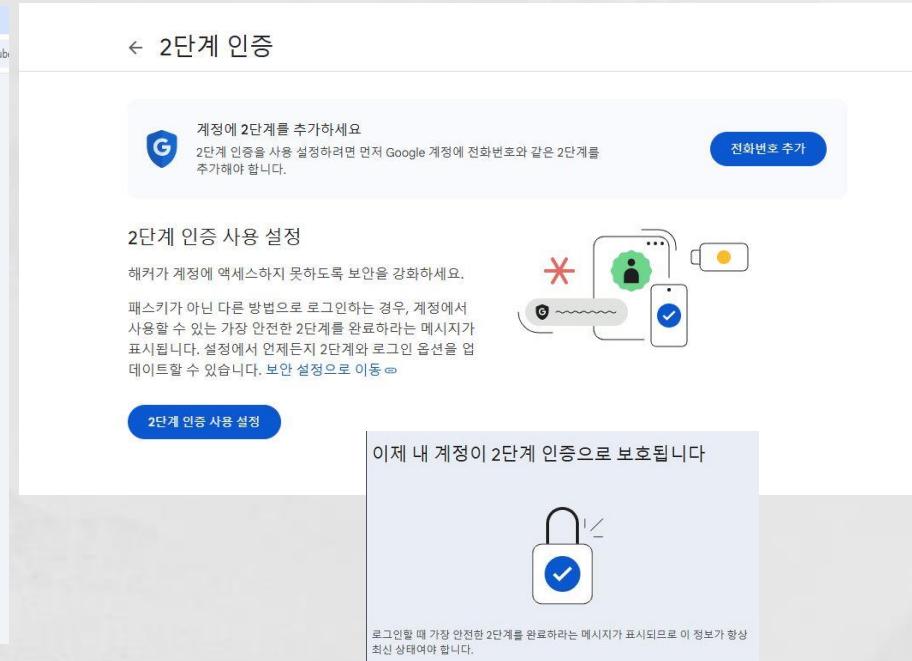
- config의 설정

설정	기본값	설명
MAIL_SERVER	localhost	이메일 서버의 호스트명
MAIL_PORT	25	이메일 서버의 포트
MAIL_USE_TLS	False	TLS를 유효로 하는가
MAIL_USE_SSL	False	SSL을 유효로 하는가
MAIL_DEBUG	app.debug	디버그 모드
MAIL_USERNAME	None	송신자 이메일 주소
MAIL_PASSWORD	None	송신자 이메일 주소의 비밀번호
MAIL_DEFAULT_SENDER	None	이메일의 송신자명과 이메일 주소

이메일 보내기

애플리케이션에서 Gmail로 이메일 보내기 준비

- Gmail을 사용하여 앱에서 이메일을 보내려면 먼저 다음의 Gmail 2단계 인증 프로세스 페이지에서 2단계 인증을 설정
- [A screenshot of a web browser showing the Google Two-Step Verification enrollment page. The URL in the address bar is \[accounts.google.com/v3/signin/challenge/pwd?TL=AO-GBTdaOetUmh49urMVK0QTbhR0PQdpqyY3VC553PNlFFN2zHd-I-7TbLP68o&checkConnection=youtube%3A284&checkedDomains=youtube.com\]\(https://accounts.google.com/v3/signin/challenge/pwd?TL=AO-GBTdaOetUmh49urMVK0QTbhR0PQdpqyY3VC553PNlFFN2zHd-I-7TbLP68o&checkConnection=youtube%3A284&checkedDomains=youtube.com\). The page displays a large 'G' logo and the user's name 'junhwa lee'. It asks the user to enter their password \('비밀번호 입력'\) and provides a checkbox for '비밀번호 표시'. Below these fields are buttons for '비밀번호 찾기' and '다음'. At the bottom, there are links for '한국어', '도움말', '개인정보처리방침', and '약관'.](https://myaccount.google.com/signinoptions/two-step-verification/enroll>Welcome</div><div data-bbox=)



이메일 보내기

애플리케이션에서 Gmail로 이메일 보내기 준비

- 앱용 비밀번호 만들기
- <https://security.google.com/settings/security/app passwords>

The screenshot shows the 'App Passwords' section of the Google Security settings. On the left, there's a sidebar with user info (junhwa lee) and a list of apps (flaskbook). The main area has a heading 'App Passwords' with a note about its use for less secure apps. It shows a table with one row for 'flaskbook'. A red box highlights the 'App Name' column entry 'flaskbook'. Below the table, a button 'Create' is highlighted with a red box. To the right, a panel titled 'Generated App Password' shows the generated password 'ohxp oiyt jmeo uein', which is also highlighted with a red box. A note below explains the password's purpose and usage.

App	Type	Label
flaskbook	Web browser	flaskbook

생성된 앱 비밀번호

기기용 앱 비밀번호

ohxp oiyt jmeo uein

사용 방법

설정하려는 애플리케이션 또는 기기의 Google 계정 설정으로 이동합니다. 비밀번호를 위에 표시된 16자리 비밀번호로 교체합니다.
일반적인 비밀번호와 마찬가지로 앱 비밀번호는 Google 계정에 대한 완전한 액세스 권한을 부여합니다. 비밀번호를 기억하지 않아도 되므로 적어 놓거나 다른 사용자와 공유하지 마세요.

확인

이메일 보내기

이메일 송신 기능 구현

- 환경 변수를 취득하기 위해서 os를 import
- Mail 클래스를 import
- Mail 클래스의 config를 환경 변수로부터 설정
- flask-mail 확장을 앱에 등록

이메일 보내기

이메일 송신 기능 구현

- C:\flaskbook\apps\minimalapp\app.py에 flask-mail 처리 추가



The screenshot shows a Windows Notepad window titled "app.py - Windows 메모장". The menu bar includes "파일(F)", "편집(E)", "서식(O)", "보기(V)", and "도움말(H)". The code in the editor is as follows:

```
# flask 클래스를 import 한다
import os

from flask_mail import Mail, Message
```

이메일 보내기

이메일 송신 기능 구현

- C:\flaskbook\apps\minimalapp\app.py에 flask-mail 처리 추가



*app.py - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
# Mail 클래스의 컨피그를 추가한다
app.config["MAIL_SERVER"] = os.environ.get("MAIL_SERVER")
app.config["MAIL_PORT"] = os.environ.get("MAIL_PORT")
app.config["MAIL_USE_TLS"] = os.environ.get("MAIL_USE_TLS")
app.config["MAIL_USERNAME"] = os.environ.get("MAIL_USERNAME")
app.config["MAIL_PASSWORD"] = os.environ.get("MAIL_PASSWORD")
app.config["MAIL_DEFAULT_SENDER"] = os.environ.get("MAIL_DEFAULT_SENDER")

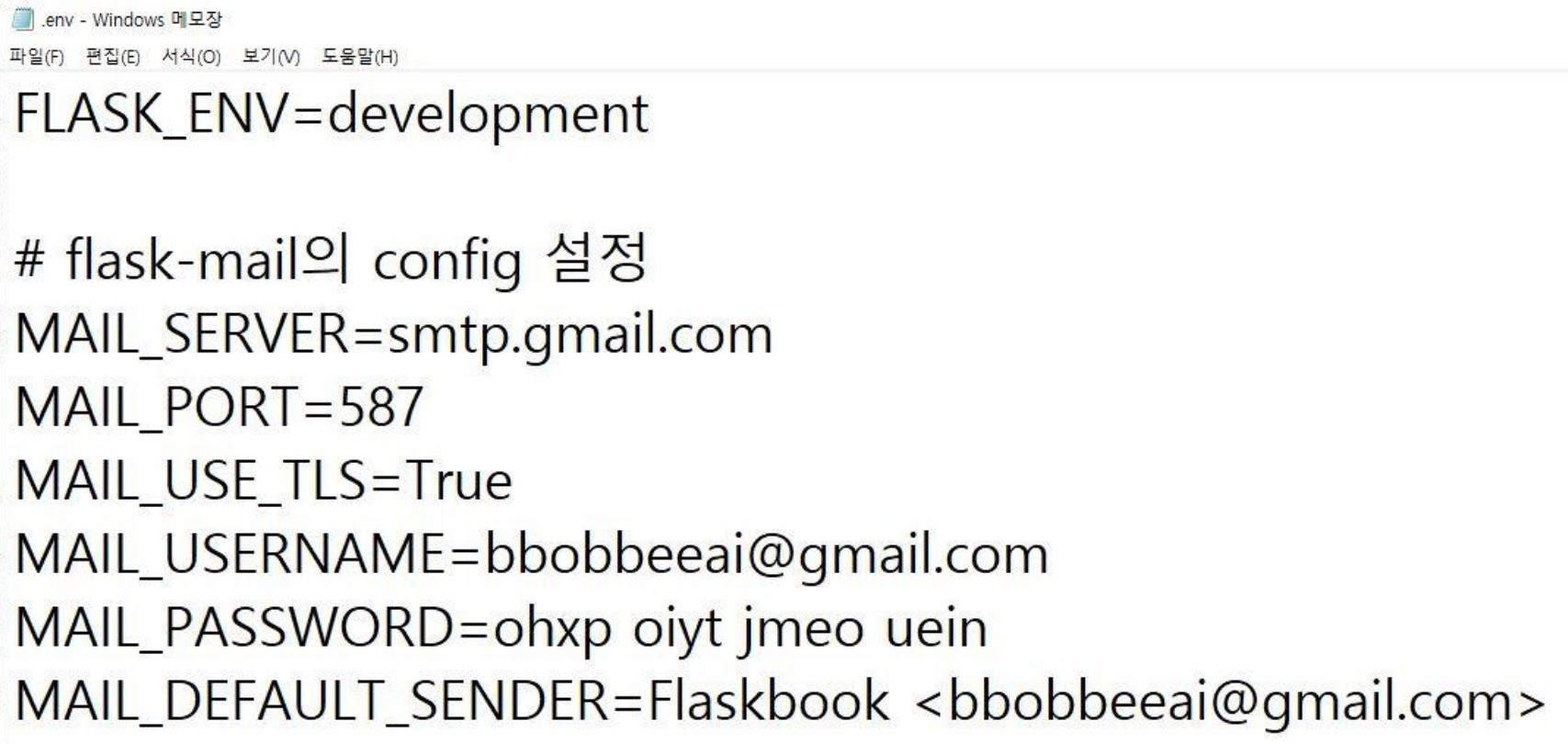
# flask-mail 확장을 등록한다
mail = Mail(app)

# URL과 실행할 함수를 매핑한다
@app.route("/")
```

이메일 보내기

이메일 송신 기능 구현

- C:\flaskbook\apps\minimalapp\env



.env - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
FLASK_ENV=development
```

flask-mail의 config 설정

```
MAIL_SERVER=smtp.gmail.com
MAIL_PORT=587
MAIL_USE_TLS=True
MAIL_USERNAME=bbobbeeai@gmail.com
MAIL_PASSWORD=ohxp oyit jmeo uein
MAIL_DEFAULT_SENDER=Flaskbook <bbobbeeai@gmail.com>
```

이메일 보내기

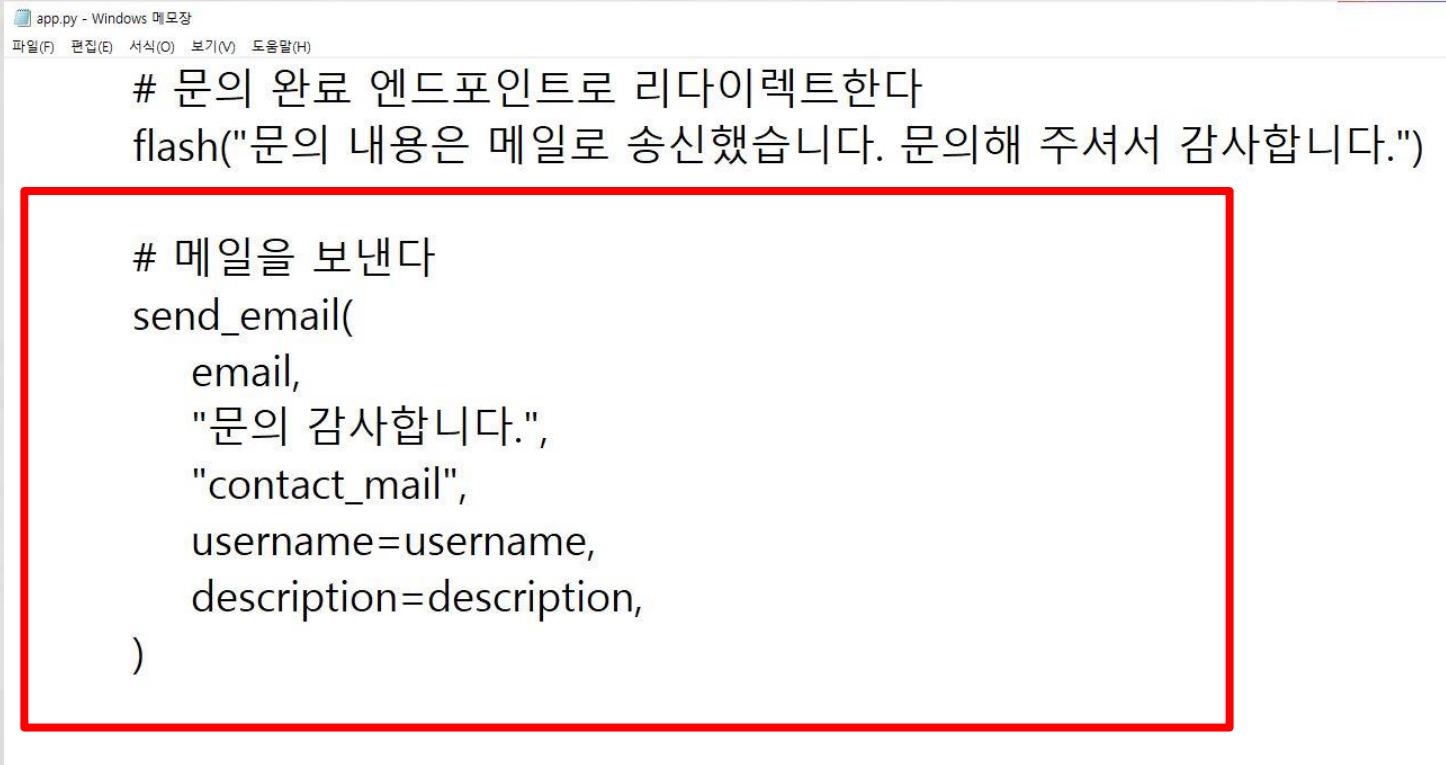
이메일 송신 기능 구현

- MAIL_SERVER는 Gmail의 송신 이메일 서버 **smtp.gmail.com**을 설정
- MAIL_PORT는 TLS(STARTTLS : 이메일 암호화)를 사용하기 위해 포트 번호 **587**을 설정
- MAIL_USE_TLS는 TLS를 유효로 하기 위해 **True**를 설정
- MAIL_USERNAME은 이용하는 **Gmail**의 이메일 주소를 설정
- MAIL_PASSWORD는 새로 생성한 앱용 **비밀번호**를 설정
- MAIL_DEFAULT_SENDER는 이메일의 송신자명 **Flaskbook <Gmail의 이메일 주소>**를 설정

이메일 보내기

이메일 보내기

- C:\flaskbook\apps\minimalapp\app.py에 문의 완료 엔드포인트 contact_complete에 이메일을 보내는 처리를 추가



```
app.py - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

# 문의 완료 엔드포인트로 리다이렉트한다
flash("문의 내용은 메일로 송신했습니다. 문의해 주셔서 감사합니다.")

# 메일을 보낸다
send_email(
    email,
    "문의 감사합니다.",
    "contact_mail",
    username=username,
    description=description,
)
```

이메일 보내기

이메일 보내기

- C:\flaskbook\apps\minimalapp\app.py에 문의 완료 엔드포인트

contact_complete에 이메일을 보내는 처리를 추가

```
"contact_mail",
username=username,
description=description,
)

return redirect(url_for("contact_complete"))
return render_template("contact_complete.html")
```

```
def send_email(to, subject, template, **kwargs):
    """메일을 송신하는 함수"""
    msg = Message(subject, recipients=[to])
    msg.body = render_template(template + ".txt", **kwargs)
    msg.html = render_template(template + ".html", **kwargs)
    mail.send(msg)
```

이메일 보내기

이메일 템플릿 만들기

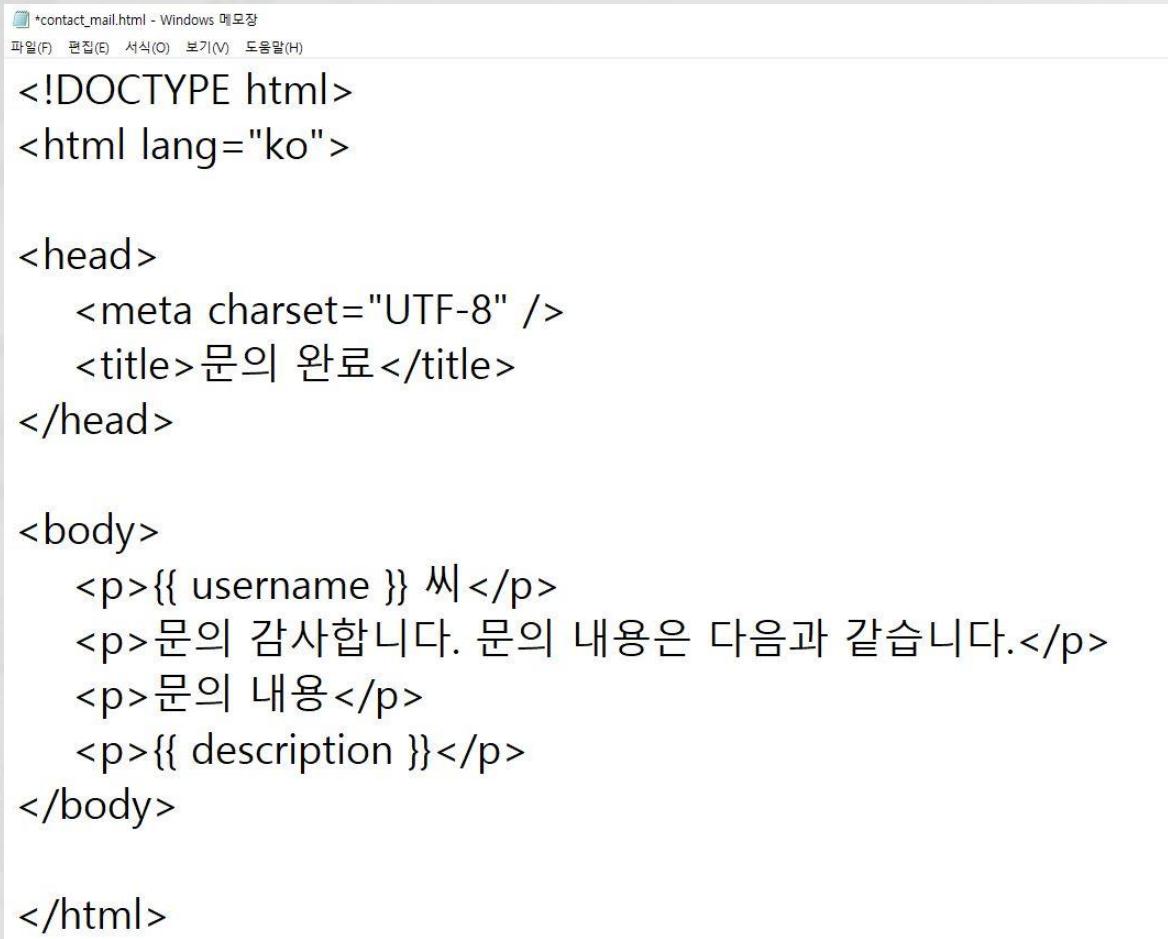
- C:\flaskbook\apps\minimalapp\templates\contact_mail.txt
- 텍스트 이메일의 템플릿 contact_mail.txt를 작성
- 변수로 치환하는 부분은 {{ }}로 지정



이메일 보내기

이메일 템플릿 만들기

- C:\flaskbook\apps\minimalapp\templates\contact_mail.html



The screenshot shows a Windows Notepad window with the file name "contact_mail.html". The menu bar includes "파일(F)", "편집(E)", "서식(O)", "보기(V)", and "도움말(H)". The content of the file is an HTML template:

```
<!DOCTYPE html>
<html lang="ko">

<head>
    <meta charset="UTF-8" />
    <title>문의 완료</title>
</head>

<body>
    <p>{{ username }} 씨</p>
    <p>문의 감사합니다. 문의 내용은 다음과 같습니다.</p>
    <p>문의 내용</p>
    <p>{{ description }}</p>
</body>

</html>
```

Flash 메시지

Flask 실행 및 확인

- (base) C:\flaskbook\apps\minimalapp>flask run

- 브라우저에서 <http://127.0.0.1:5000/contact> URL에 접근

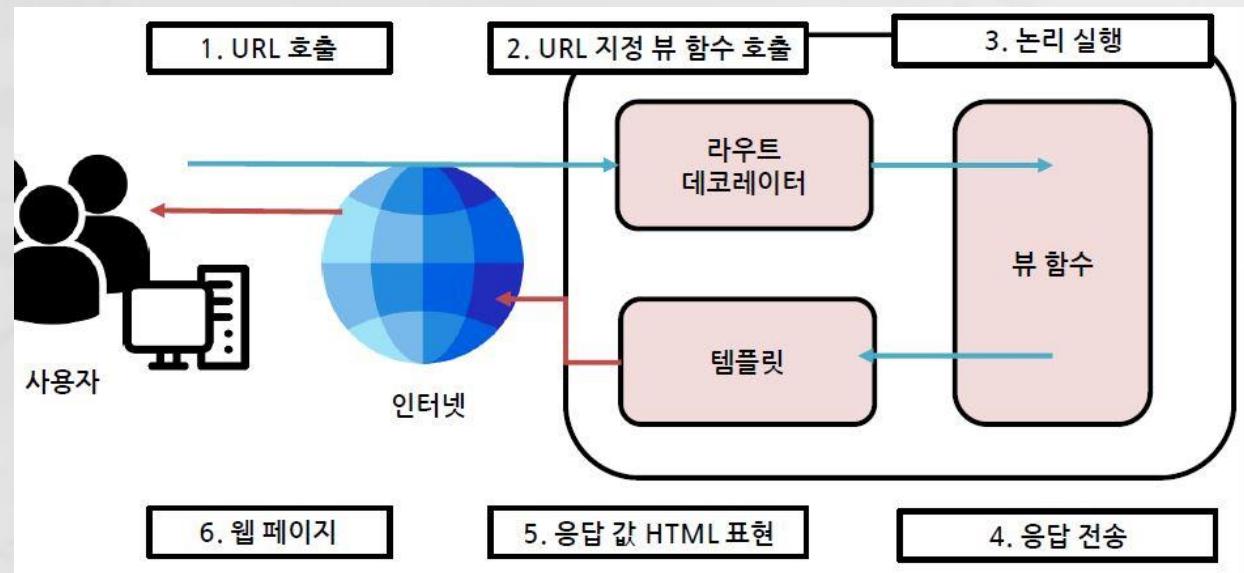
The screenshot illustrates the development and testing of a Flask application. At the top, a terminal window titled '관리자: Anaconda Prompt - flask run' shows the command `(base) C:\flaskbook\apps\minimalapp>flask run` being executed. Below it, a web browser window displays a contact form with a yellow background. The form fields include '사용자명' (Name) set to '이준화', '메일 주소' (Email) set to 'bbobbeeai@gmail.com', and a '문의 내용' (Message Content) field containing the text '안녕하세요 플라스크에 대해 질문 있습니다.' A red box highlights the '문의' (Inquiry) button at the bottom left of the form. To the right, a separate window shows an email inbox with a message from 'Flaskbook <bbobbeeai@gmail.com>' to '이준화 씨' (Lee Jun-hwa). The email body says '문의 감사합니다.' (Thank you for your inquiry.) and includes the same message content. Below the email are standard messaging controls for reply, forward, and delete.

플라스크 앱 구조

플라스크 앱 구조

- 플라스크 앱은 다음 과정을 통해 호출

- 특정 URL 호출
- 뷰 함수 호출
- 논리 실행
- 논리 결과 응답 전송
- 응답 값 HTML 표현
- 클라이언트 전달



플라스크 앱 구조

플라스크 정적 라우팅

- 플라스크는 복잡한 URI 를 쉽게 함수로 연결하는 방법 제공
- route() 함수 사용 앱 객체 제어

The screenshot shows a web browser interface with two requests to a Flask application running at 127.0.0.1:5000.

Request 1: URL 127.0.0.1:5000

```
from flask import Flask
app = Flask(__name__)

@app.route("/hello")
def hello():
    return "<h1>Hello World!</h1>"

if __name__ == "__main__":
    app.run()
```

Request 2: URL 127.0.0.1:5000/hello

Not Found
The requested URL was not found on the server. If you entered the URL manually, please check your spelling and try again. If you have an application deployed here, make sure you have configured the URL correctly.

Hello World!

플라스크 앱 구조

플라스크 동적 라우팅

- 동적 라우팅 상황에 따라 변화하는 URI 적용
- URL 을 변수 형태로 사용

▶ 입력 URL 인자에 따라 출력이 바뀐다. (/)

```
from flask import Flask, url_for
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello Page!"

@app.route('/profile/<username>/')
def get_profile(username):
    return 'profile : ' + username

if __name__ == "__main__":
    with app.test_request_context():
        print(url_for('get_profile', username='flask'))
    app.run()
```

← → ⌂ ⓘ 127.0.0.1:5000

Hello Page!

← → ⌂ ⓘ 127.0.0.1:5000/profile/user01

profile : user01

← → ⌂ ⓘ 127.0.0.1:5000/profile/user01/

Not Found

The requested URL was not found on the server. If

플라스크 앱 구조

플라스크 HTML 렌더링

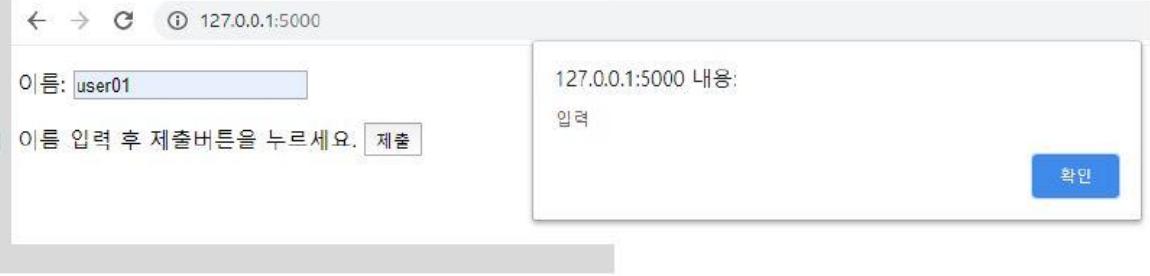
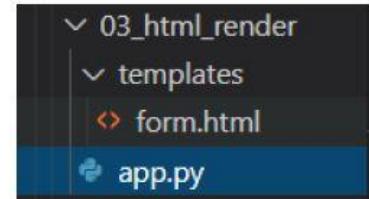
- 플라스크는 기본 HTML 코드를 반환하면 페이지로 렌더링함

```
<form>
  <p>이름: <input type="text" id="input"></p>
  <p>이름 입력 후 제출버튼을 누르세요.
    <input type="button" value="제출" onclick="alert('입력')"/>
  </p>
</form>
```

```
from flask import Flask, render_template
app = Flask(__name__)

@app.route("/")
def hello():
    return render_template('form.html')

if __name__ == "__main__":
    app.run()
```



파일 서버(File Server) 구현

File Server 사이트 설계

- 전체

이름	만든 날짜	마지막으로 수정한 날짜	파일 크기
779b9fc61d93bb68031227c420a5155.jpg	[삭제]	2025-02-04	1.02 MB
flask.png	[삭제]	2025-02-04	46.84 KB
idea.txt	[삭제]	2025-02-04	0.59 KB
Zoomlt.zip	[삭제]	2025-02-04	448.07 KB

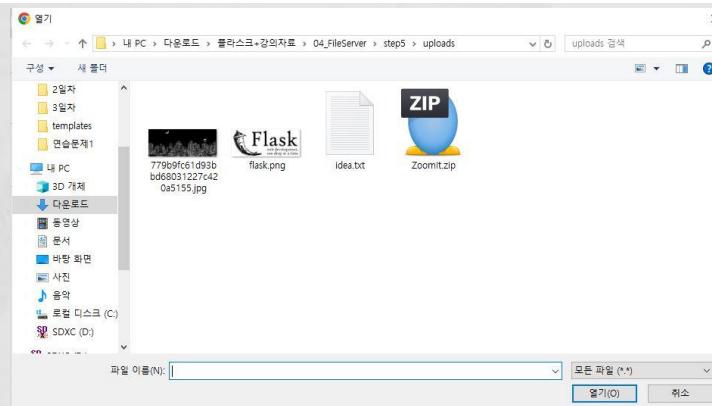
파일 서버(File Server) 구현

File Server 사이트 설계

◦ 파일 업로드

The screenshot shows a web browser window titled 'Server' at the URL '127.0.0.1:5000'. The page is titled 'FileServer' and displays a file upload interface. A red box highlights the '파일 선택' (File Selection) button and the message '선택된 파일 없음' (No files selected). To the right of the table is an 'Upload' icon (up arrow) and a 'Download' icon (down arrow).

이름	만든 날짜	마지막으로 수정한 날짜	파일 크기
779b9fc61d93bbd68031227c420a5155.jpg	[삭제]	2025-02-04	2025-02-04 1.02 MB
flask.png	[삭제]	2025-02-04	2025-02-04 46.84 KB
idea.txt	[삭제]	2025-02-04	2025-02-04 0.59 KB
ZoomIt.zip	[삭제]	2025-02-04	2025-02-04 448.07 KB



파일 서버(File Server) 구현

File Server 사이트 설계

◦ 파일 업로드

The screenshot shows a web browser window titled "Server" with the URL "127.0.0.1:5000". The page is titled "FileServer" and displays a file upload interface. The upload path is listed as "폴더: C:\flaskbook\FileServer\step5\uploads". A red box highlights the "제출" (Submit) button. Below it is a table listing four files:

이름	만든 날짜	마지막으로 수정한 날짜	파일 크기
779b9fc61d93bbd68031227c420a5155.jpg	[삭제]	2025-02-04	2025-02-04 1.02 MB
flask.png	[삭제]	2025-02-04	2025-02-04 46.84 KB
idea.txt	[삭제]	2025-02-04	2025-02-04 0.59 KB
ZoomIt.zip	[삭제]	2025-02-04	2025-02-04 448.07 KB

The screenshot shows a confirmation message for a file upload. The message reads "파일 업로드 성공!" (File upload successful!) and features a large checkmark icon. Above the message, the upload path is shown as "폴더: C:\Users\dsz08\Desktop\coding\python\flask\Flask-FileService-app\step5\uploads". There are also up and down arrow icons.

파일 서버(File Server) 구현

File Server 사이트 설계

○ 파일 업로드

The screenshot shows a file management interface titled "FileServer". At the top, there is a navigation bar with a logo and the title. Below it, a message indicates the upload path: "폴더: C:\flaskbook\FileServer\step5\uploads". On the right side, there are two icons: an upward arrow for uploading and a downward arrow for downloading.

Below the folder path, there are three buttons: "파일 선택" (Select File), which is highlighted in red; "선택된 파일 없음" (No selected files); and "제출" (Submit).

The main area displays a table of uploaded files:

이름	만든 날짜	마지막으로 수정한 날짜	파일 크기
779b9fc67d95bb88031227c420a5155.jpg	[삭제]	2025-02-04	1.02 MB
app.py	[삭제]	2025-02-04	2.20 KB
flask.jpg	[삭제]	2025-02-04	16.04 KB
idea.txt	[삭제]	2025-02-04	0.59 KB
ZoomIt.zip	[삭제]	2025-02-04	448.07 KB

파일 서버(File Server) 구현

File Server 사이트 설계

○ 파일 업로드 검증

FileServer

폴더: C:\flaskbook\FileServer\step5\uploads

이름	파일 선택	선택된 파일 없음	제출	만든 날짜	마지막으로 수정한 날짜	파일 크기
 779b9fc61d93bbd68031227c420a5155.jpg	[삭제]	파일을 선택하세요.		2025-02-04	2020-08-25	1.02 MB
 app.py	[삭제]			2025-02-04	2025-02-04	0.31 KB
 flask.png	[삭제]			2025-02-04	2020-08-25	46.84 KB
 idea.txt	[삭제]			2025-02-04	2020-08-25	0.59 KB
 ZoomIt.zip	[삭제]			2025-02-04	2020-08-25	448.07 KB

파일 서버(File Server) 구현

File Server 사이트 설계

◦ 파일 정보 확인 및 나열

The screenshot shows a file management interface titled "FileServer". At the top, there's a navigation bar with a logo and the title. Below it, a message indicates the folder path: "폴더: C:\flaskbook\FileServer\step5\uploads". On the right side, there are two icons for file upload and download. In the center, there are two buttons: "파일 선택" (File Selection) and "제출" (Submit). To the left, a table lists five files with their details:

	이름	
	779b9fc61d93bbd68031227c420a5155.jpg	[삭제]
	app.py	[삭제]
	flask.png	[삭제]
	idea.txt	[삭제]
	ZoomIt.zip	[삭제]

To the right of this table is another table showing file metadata:

만든 날짜	마지막으로 수정한 날짜	파일 크기
2025-02-04	2025-02-04	1.02 MB
2025-02-04	2025-02-04	2.20 KB
2025-02-04	2025-02-04	46.84 KB
2025-02-04	2025-02-04	0.59 KB
2025-02-04	2025-02-04	448.07 KB

The last five rows of this table are highlighted with a red border.

파일 서버(File Server) 구현

File Server 사이트 설계

○ 파일 삭제

FileServer

폴더: C:\flaskbook\FileServer\step5\uploads

파일 선택 선택된 파일 없음 제출

이름	만든 날짜	마지막으로 수정한 날짜	파일 크기
[삭제] 779b9fc61d93bbd68031227c420a5155.jpg	2025-02-04	2025-02-04	1.02 MB
[삭제] app.py	2025-02-04	2025-02-04	2.20 KB
[삭제] flask.png	2025-02-04	2025-02-04	46.84 KB
[삭제] idea.txt	2025-02-04	2025-02-04	0.59 KB
[삭제] ZoomIt.zip	2025-02-04	2025-02-04	448.07 KB

폴더: C:\flaskbook\FileServer\step5\uploads

파일 선택 선택된 파일 없음 제출

이름	만든 날짜	마지막으로 수정한 날짜	파일 크기
779b9fc61d93bbd68031227c420a5155.jpg	2025-02-04	2025-02-04	1.02 MB
flask.png	2025-02-04	2025-02-04	46.84 KB
idea.txt	2025-02-04	2025-02-04	0.59 KB
ZoomIt.zip	2025-02-04	2025-02-04	448.07 KB

정리

정리

- Flash 메시지
- 이메일 보내기
- File Server 만들기