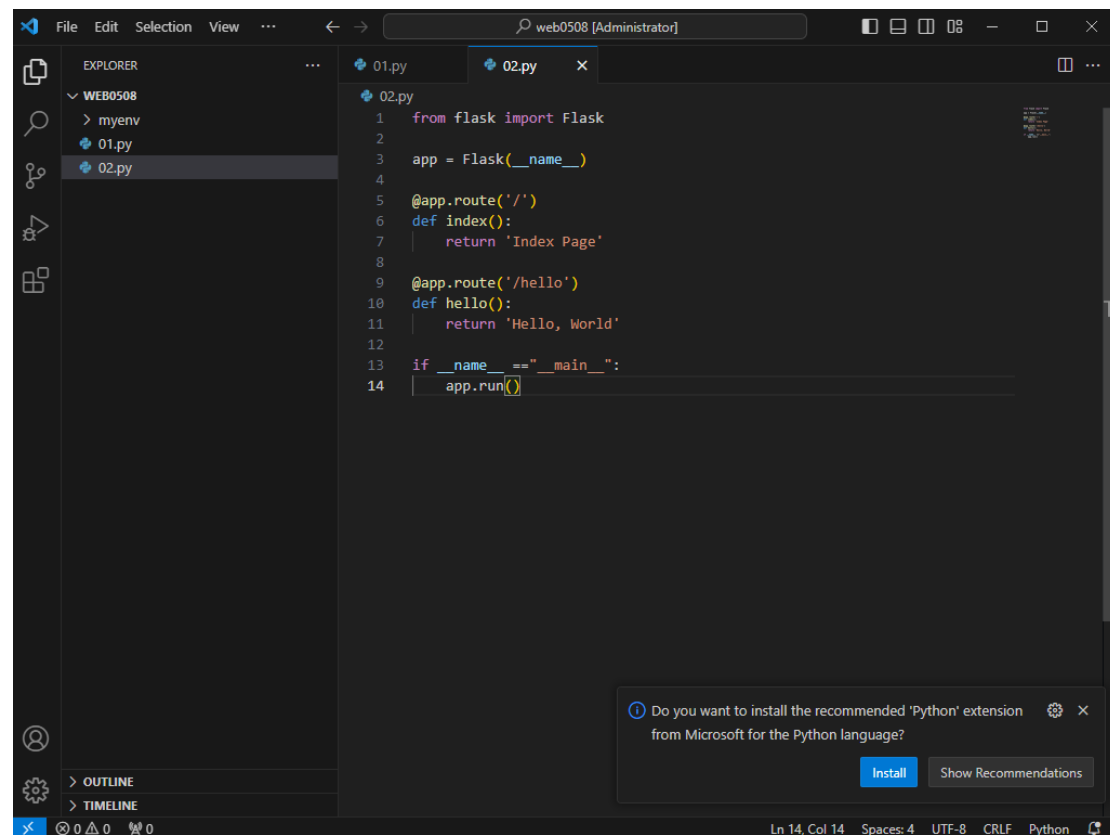
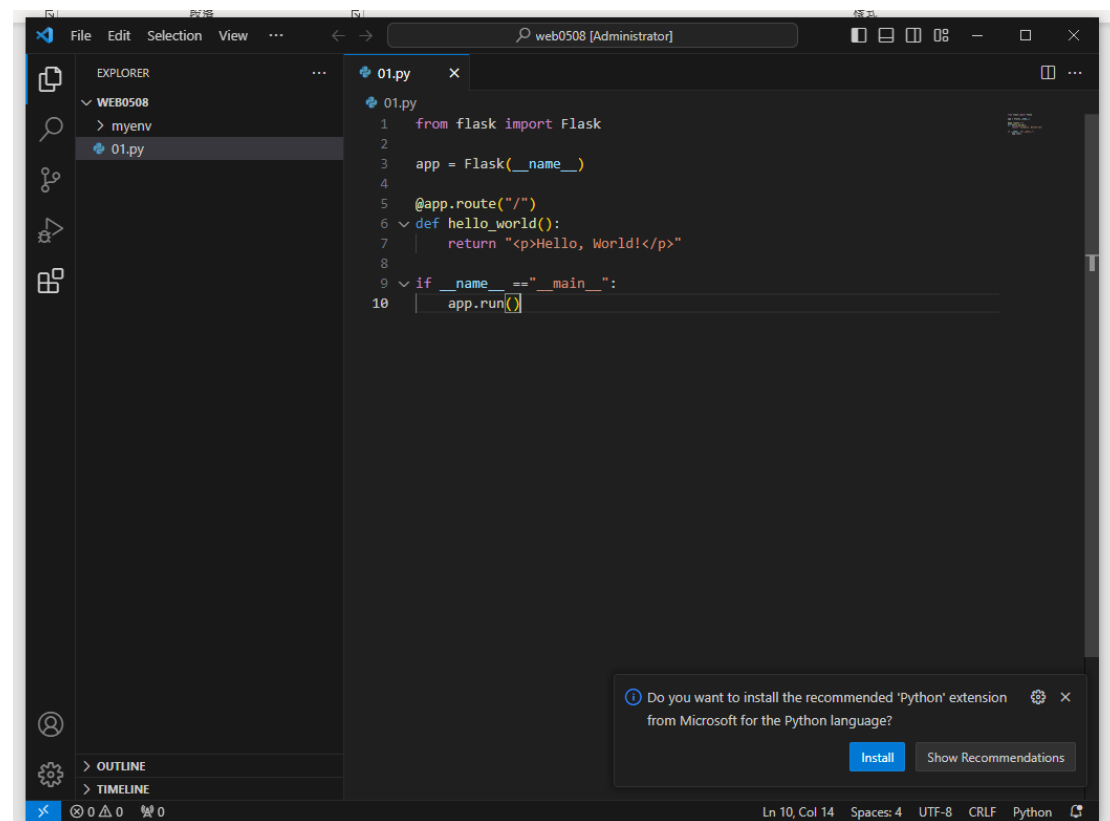
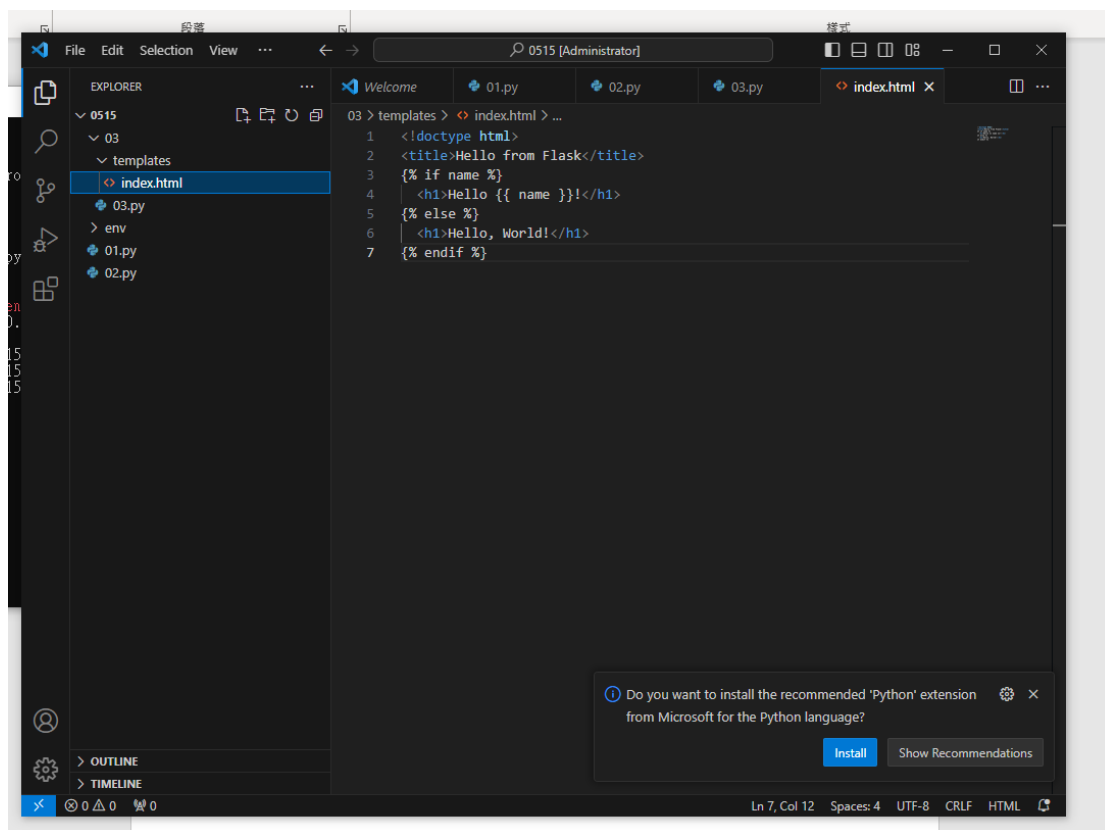
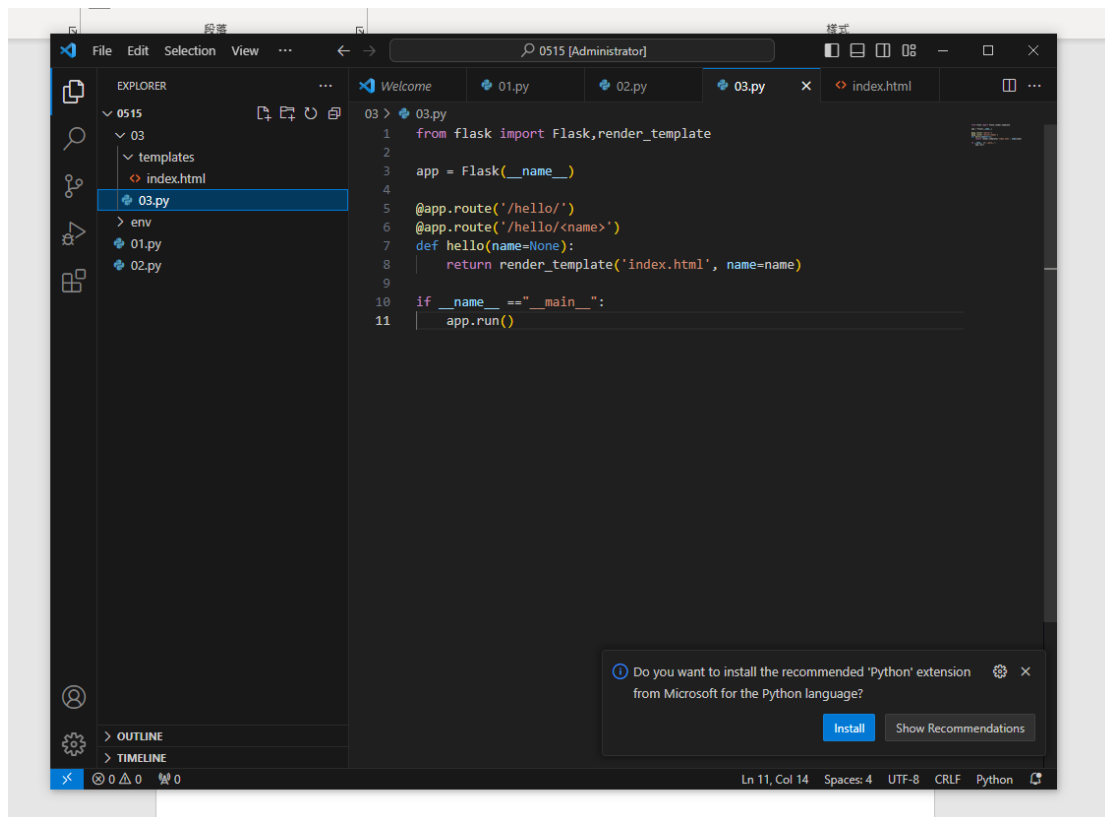
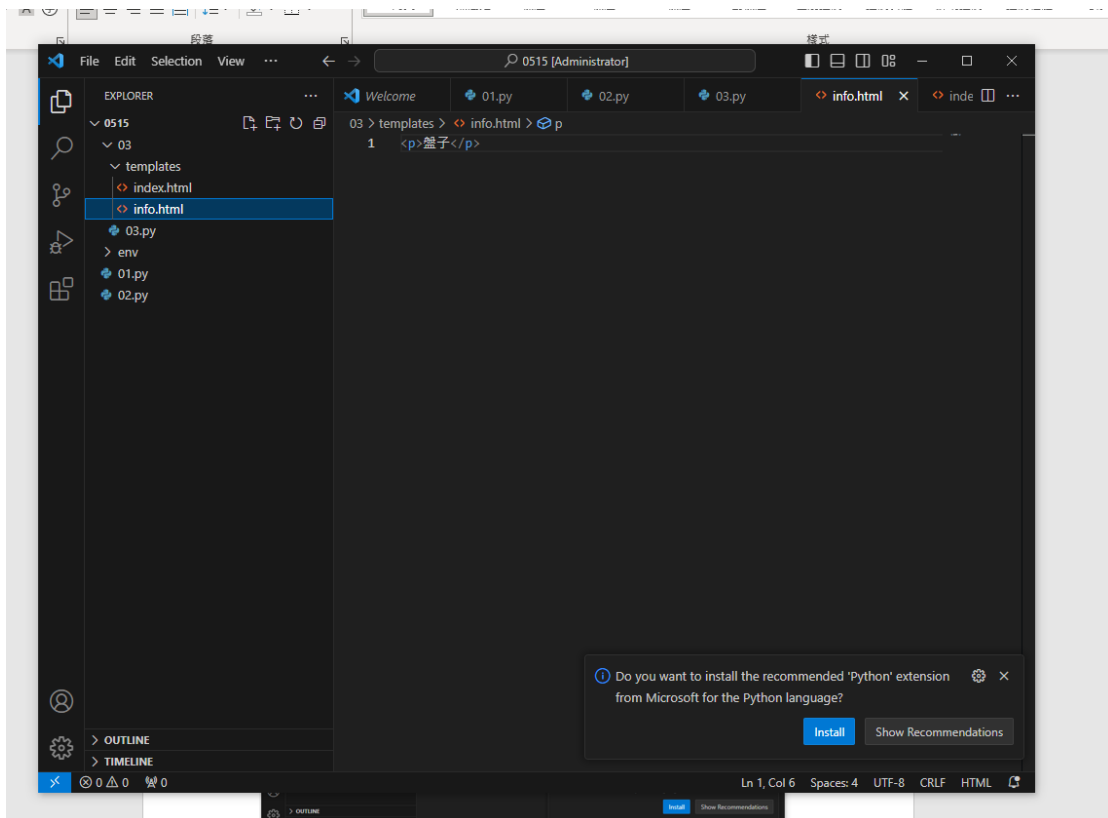
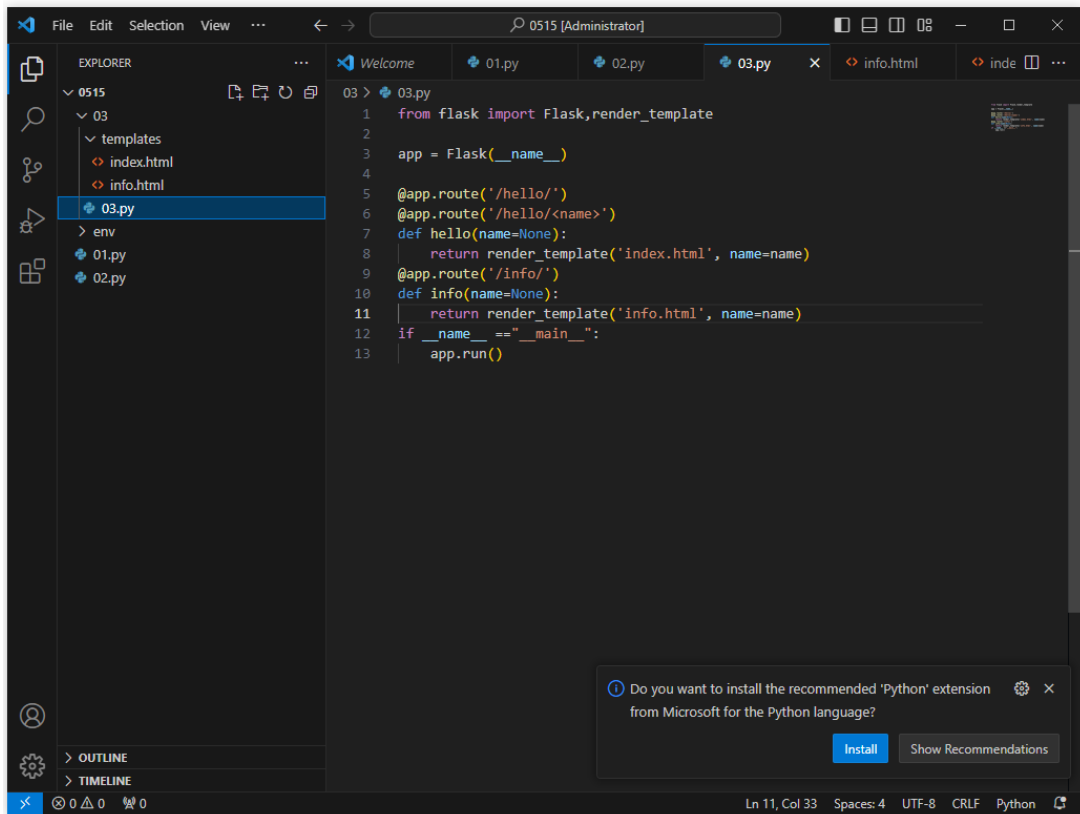


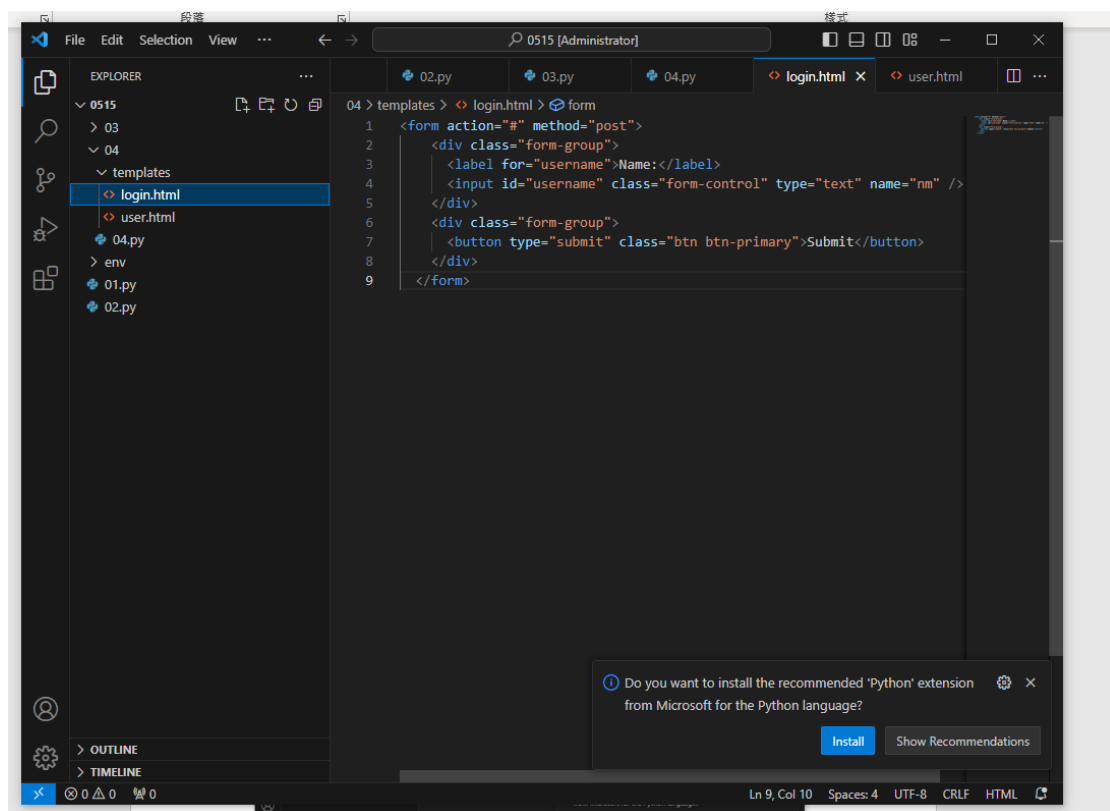
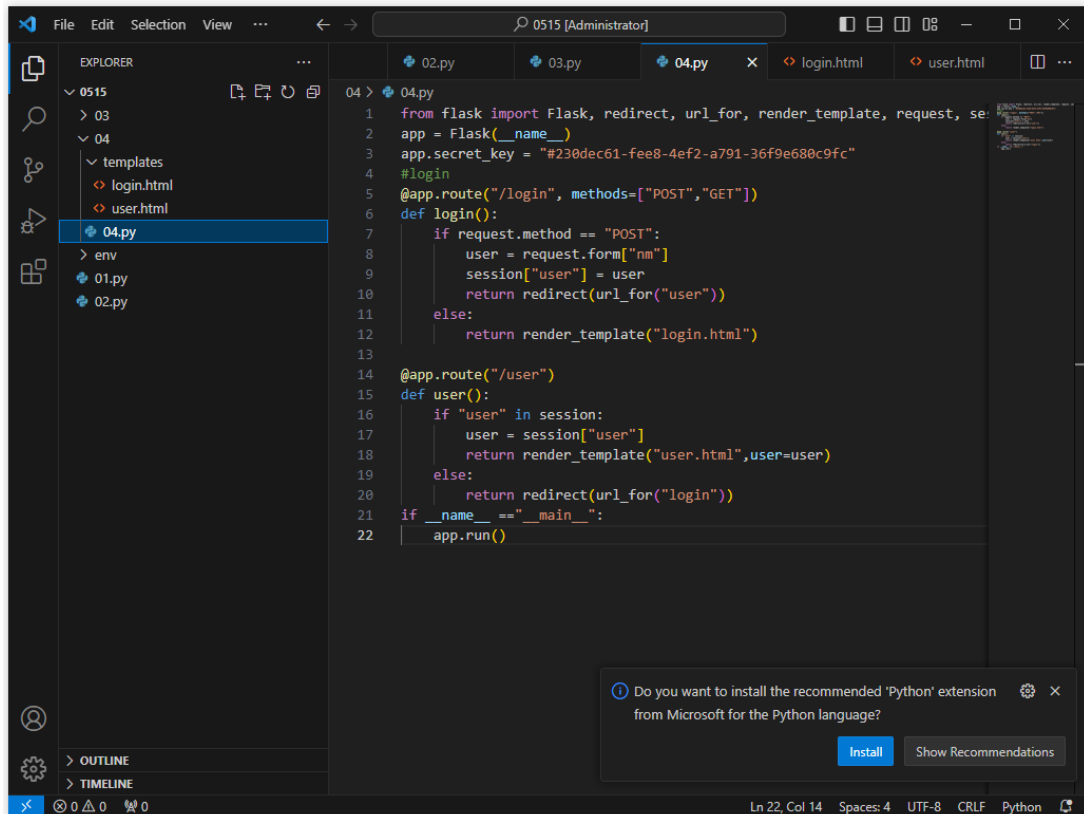
0508

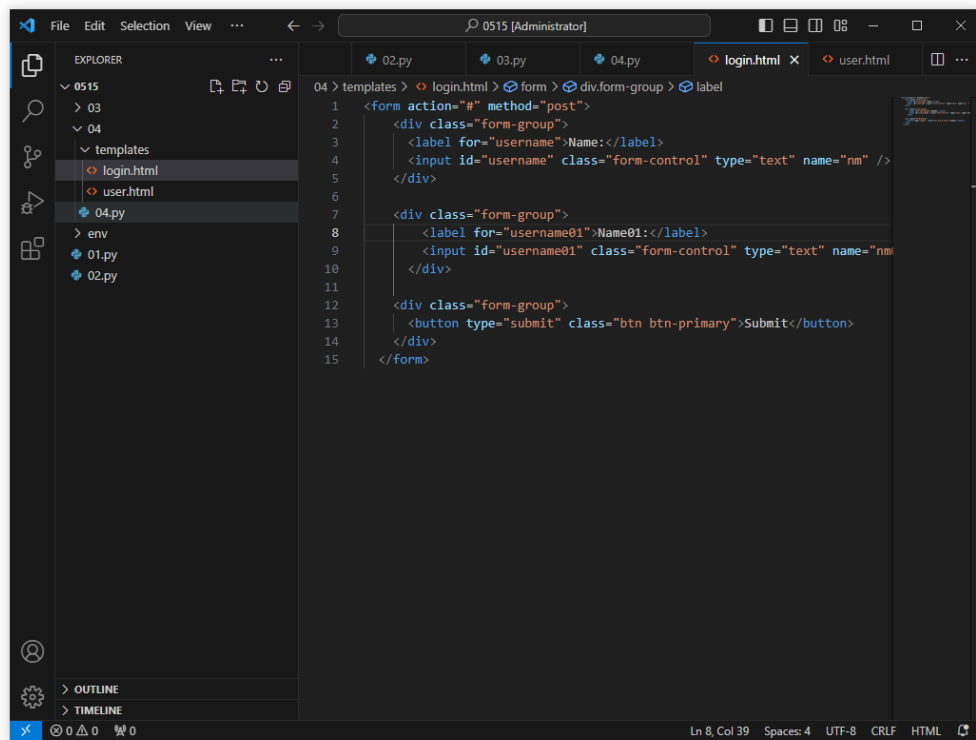
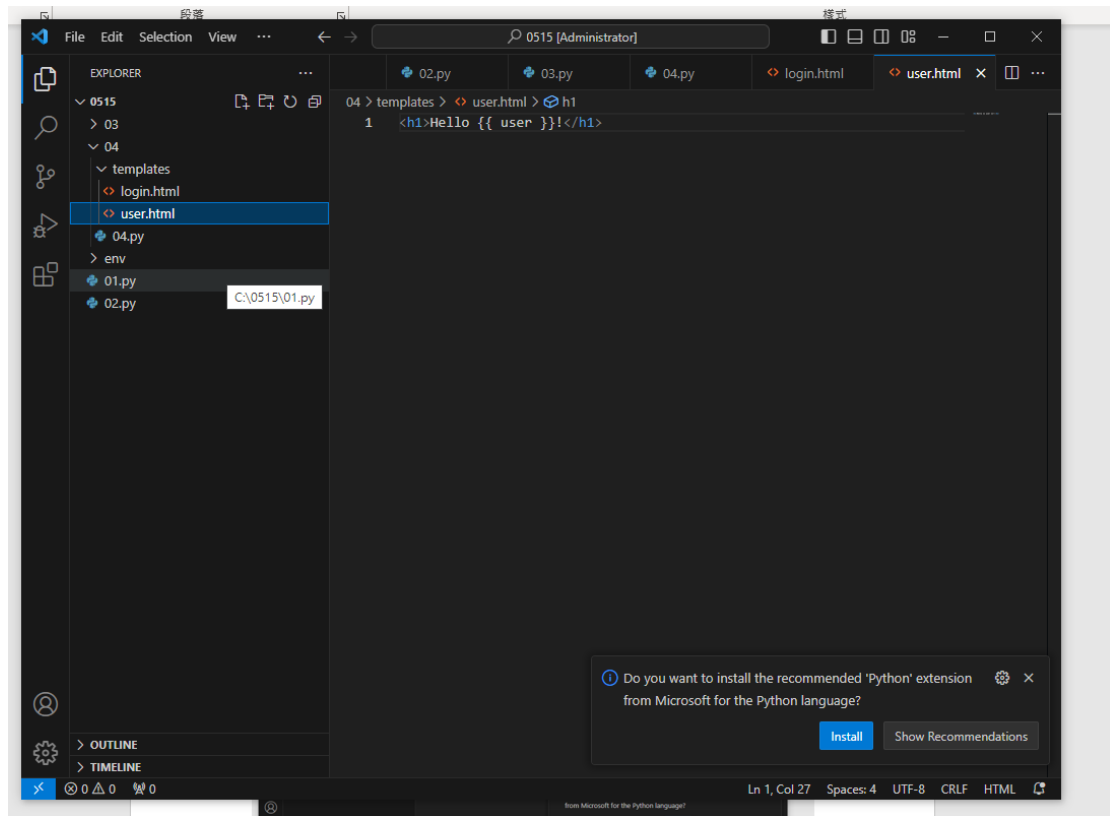


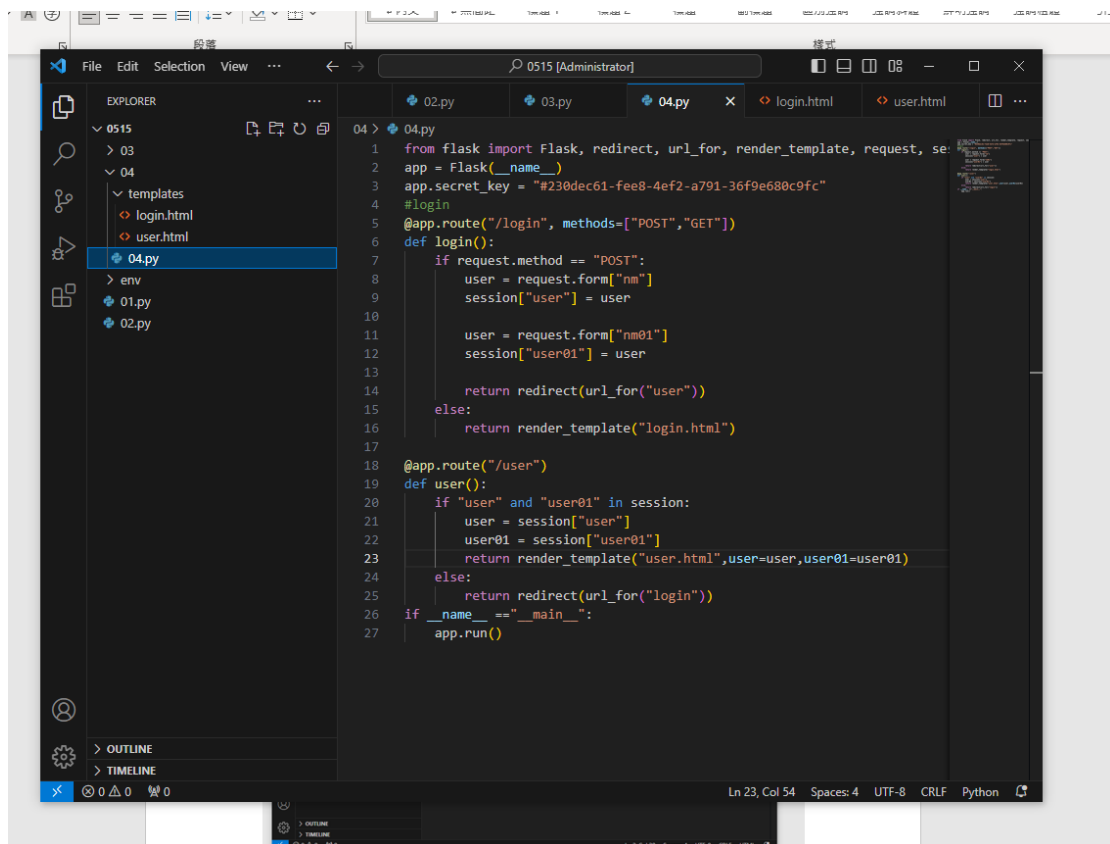
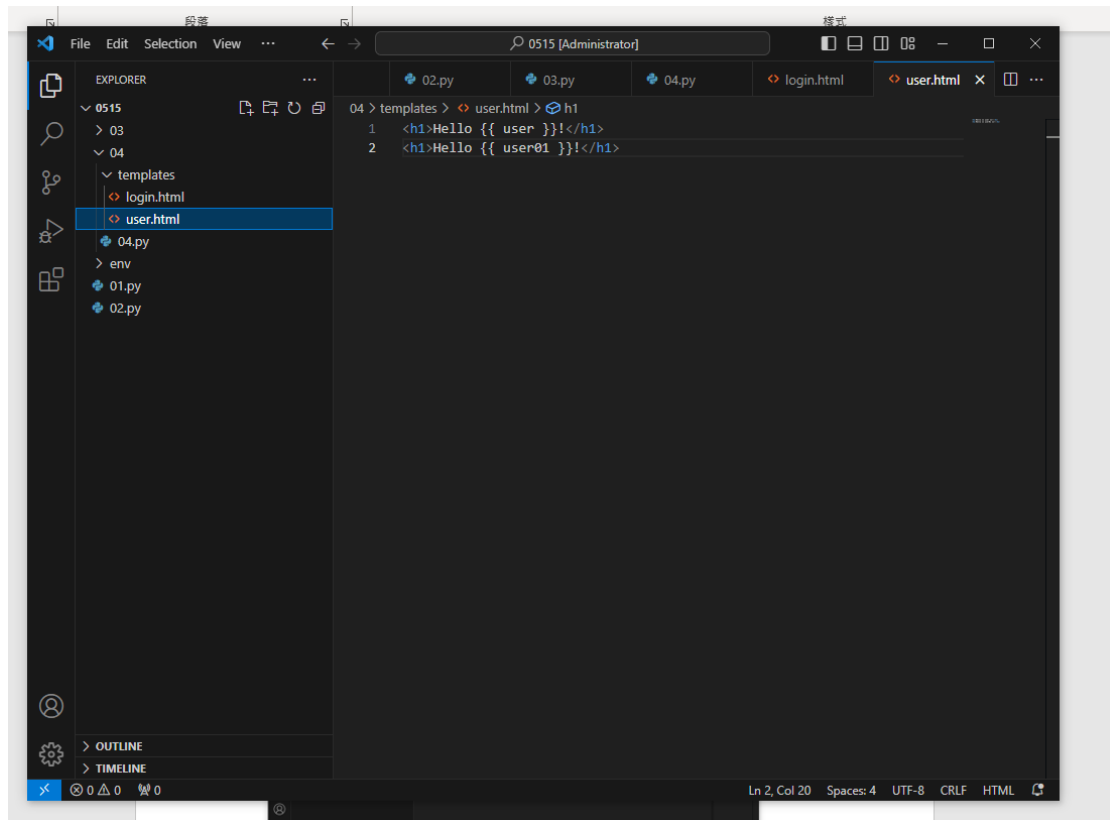
0515

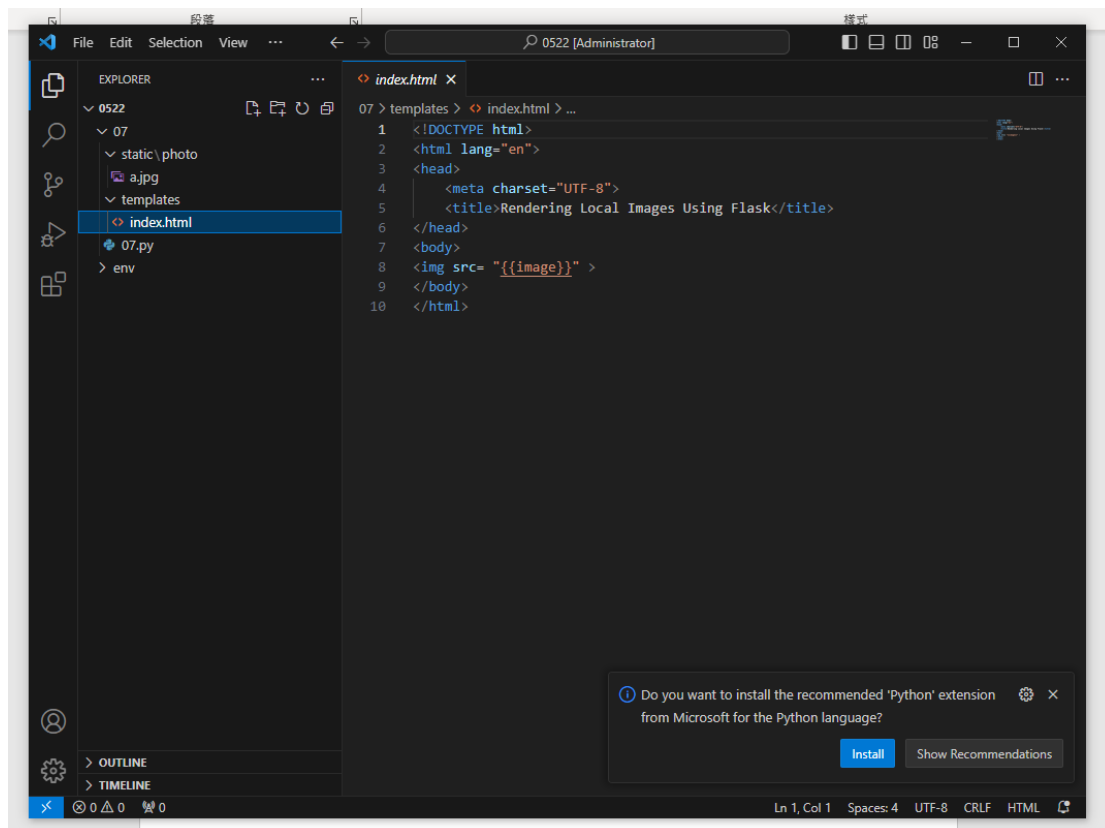
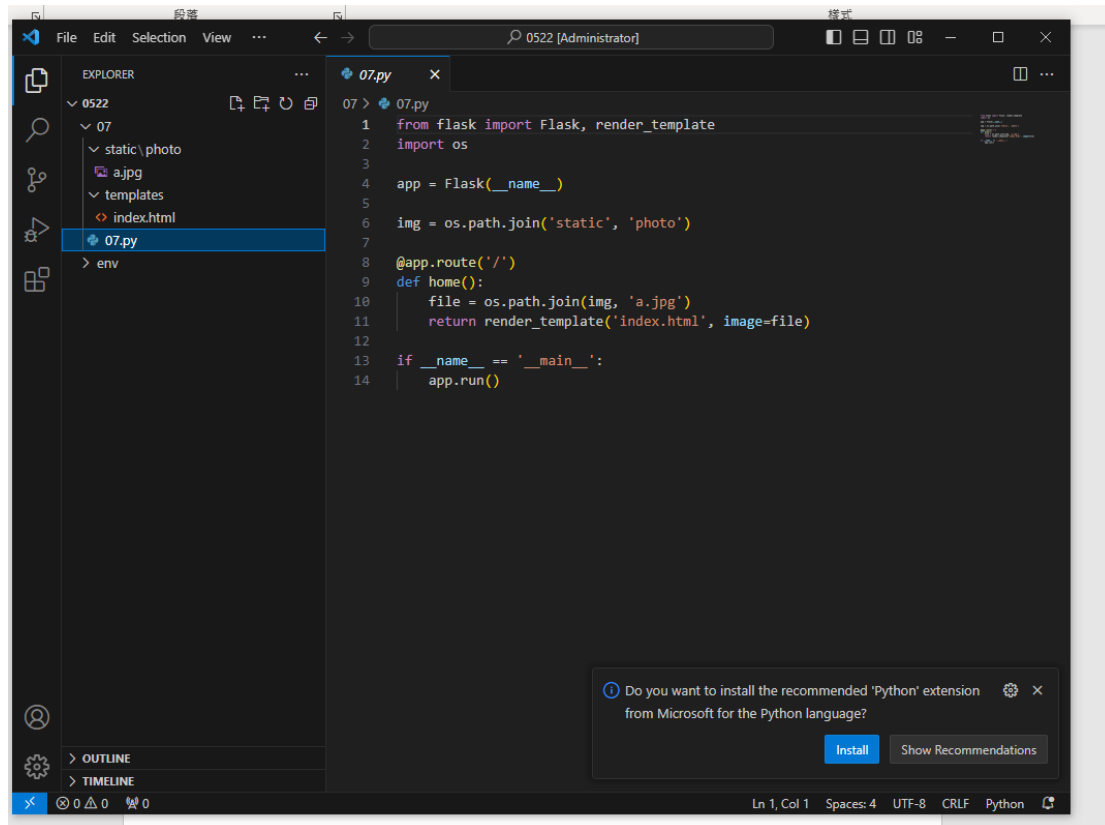


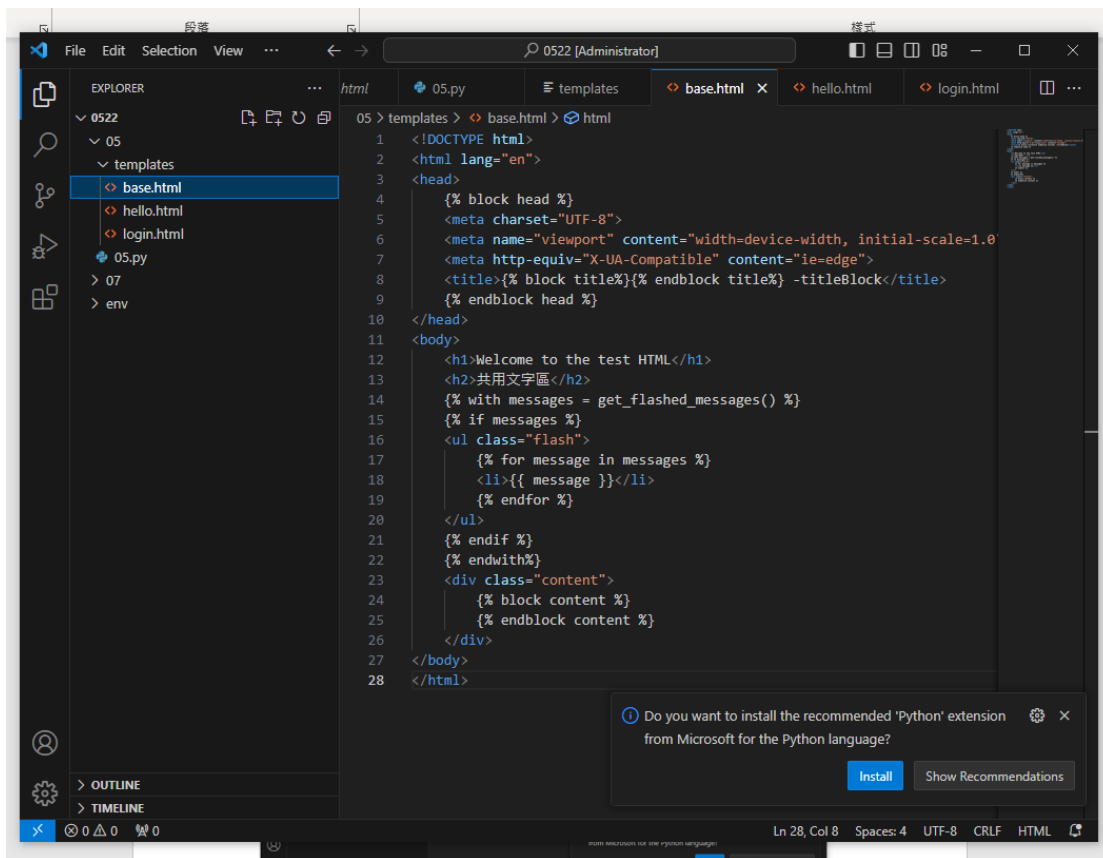
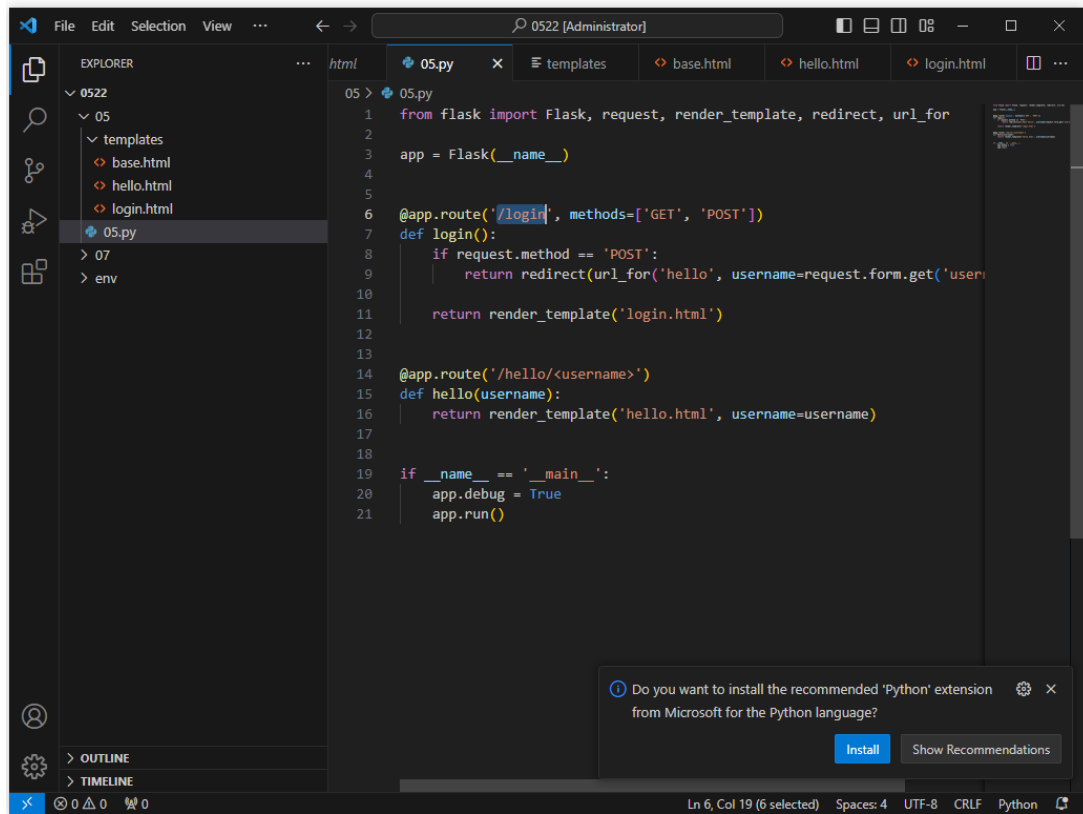


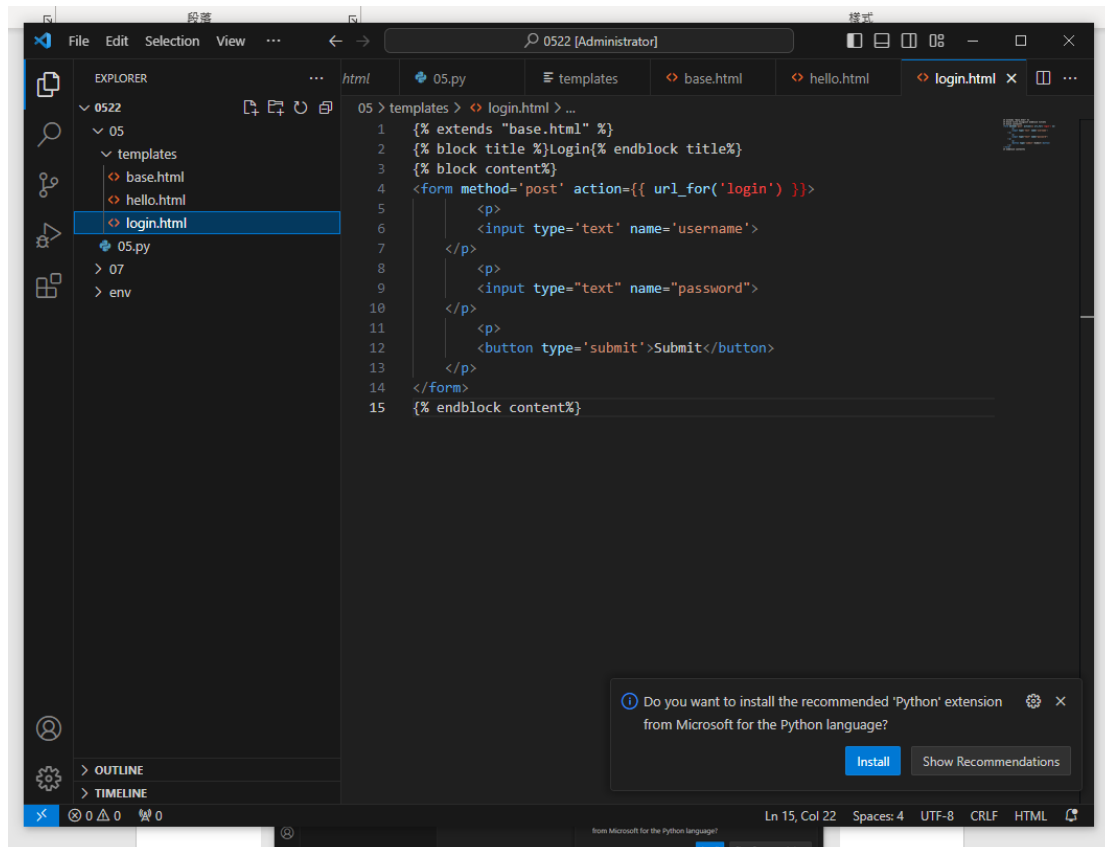
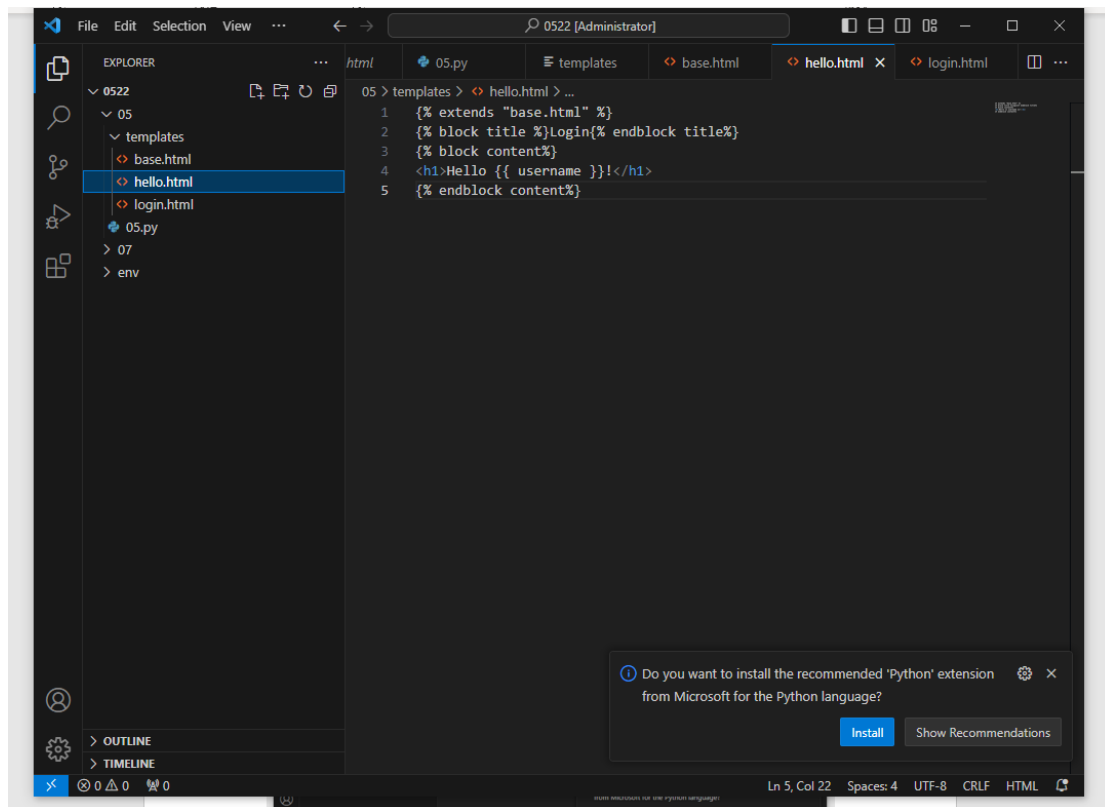


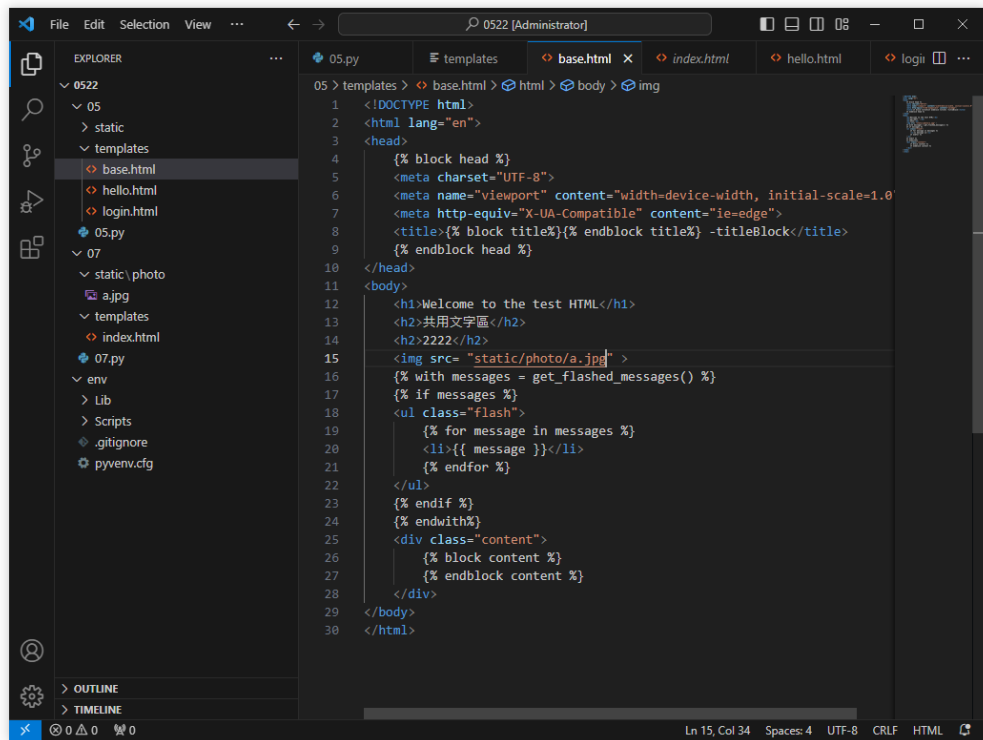






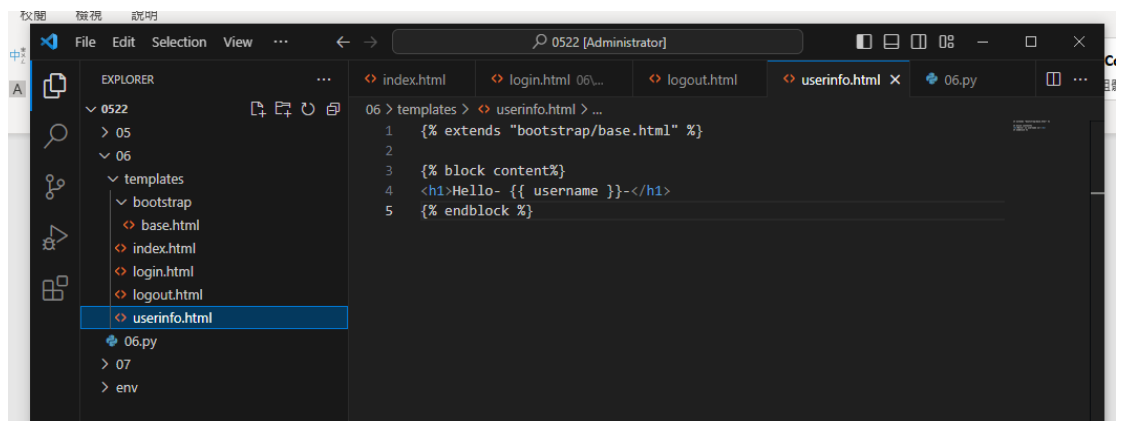
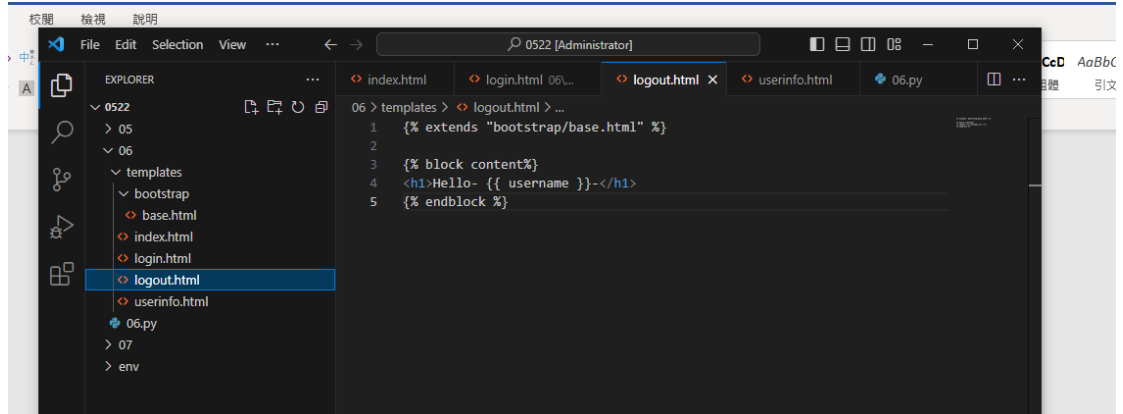
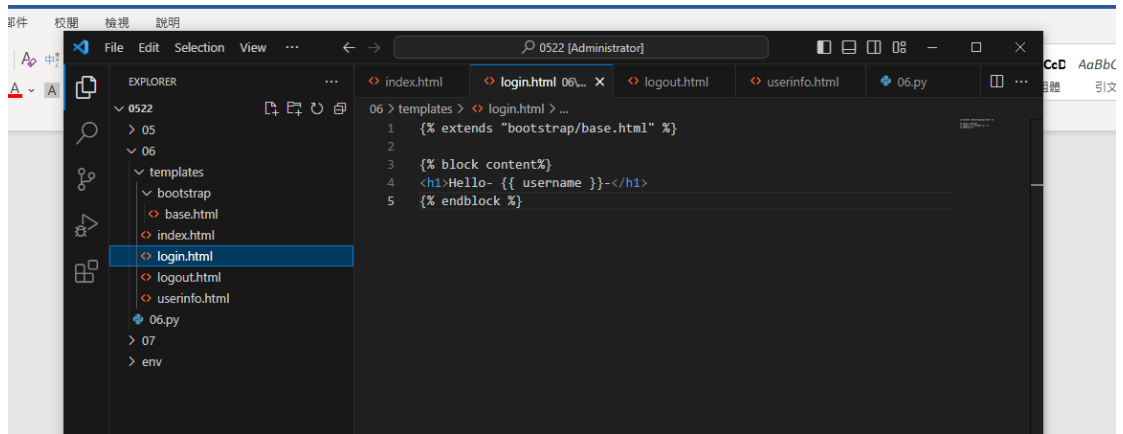


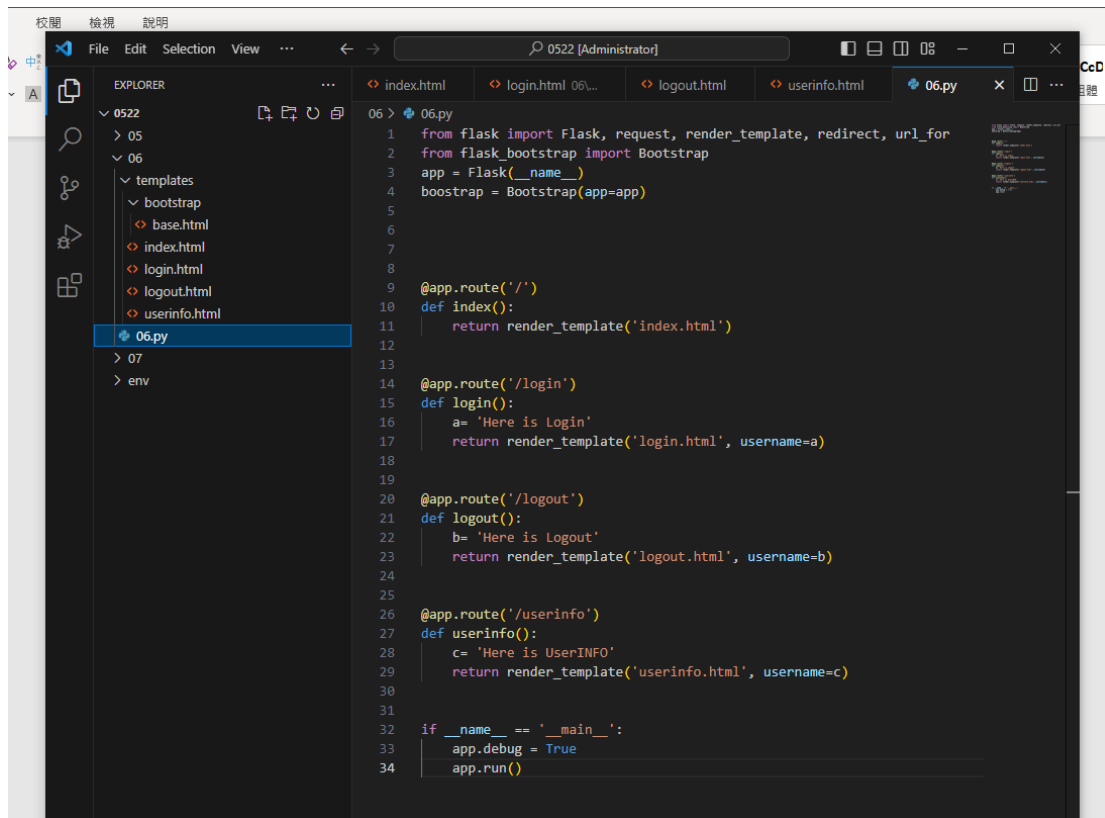




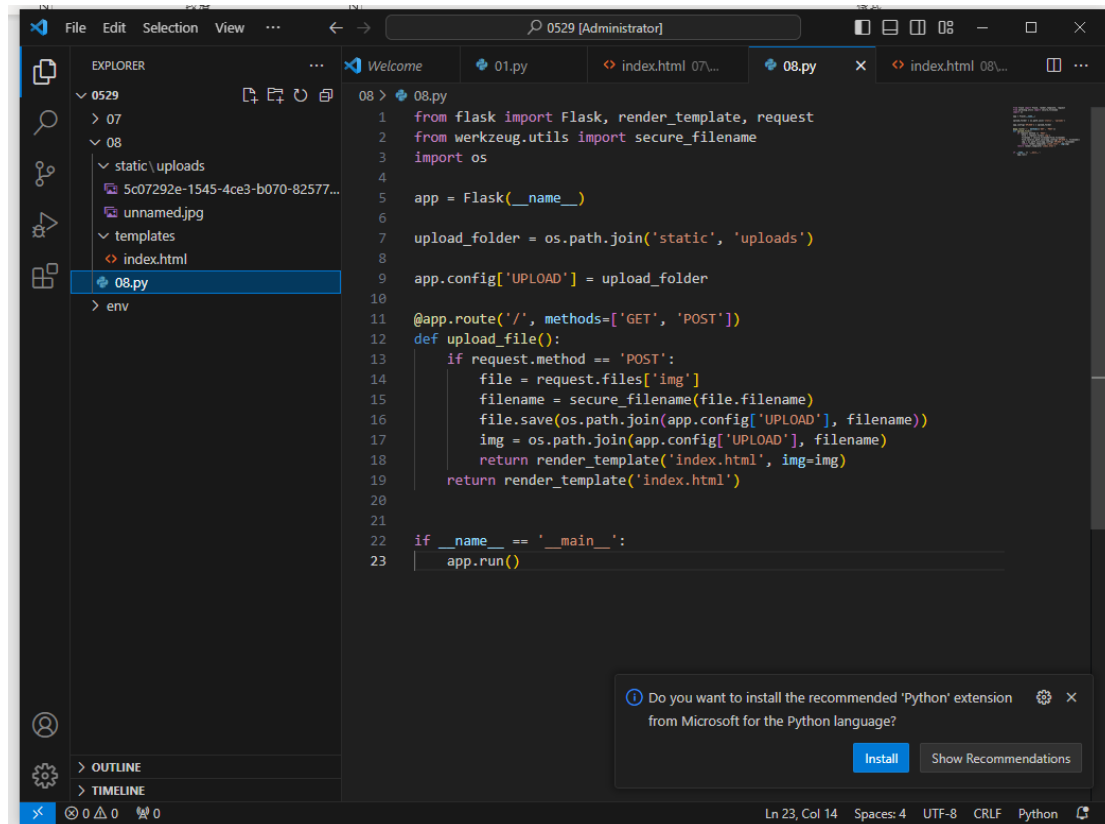
```
06 > templates > bootstrap > base.html > html
1 <!doctype html>
2 <html lang="en">
3 <head>
4 <!-- Required meta tags -->
5 <meta charset="utf-8">
6 <meta name="viewport" content="width=device-width, initial-scale=1,
7
8 <!-- Bootstrap CSS -->
9 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bo
10 <title>{% block title %} Harden {% endblock %}</title>
11 </head>
12 <body>
13
14
15 {% block navbar %}
16 <nav class="navbar navbar-expand-lg bg-body-tertiary">
17 <div class="container-fluid">
18 <a class="navbar-brand" href="{{ url_for('index') }}">Navbar</a>
19 <button class="navbar-toggler" type="button" data-bs-toggle="collaps
20 <span class="navbar-toggler-icon"></span>
21 </button>
22 <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
23 <div class="navbar-nav">
24 <a class="nav-link active" aria-current="page" href="{{ url_for(
25 <a class="nav-link" href="{{ url_for('logout') }}">LOGOUT</a>
26 <a class="nav-link" href="{{ url_for('login') }}">LOGIN</a>
27
28 </div>
29
30 </div>
31
32 </div>
33 </nav>
34 </div>
35 {% endblock %}
36
37 <div class="content">
38 <div class="content">
39 <div class="content">
40 <div class="content">
41
42
43 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootst
44 </body>
45 </html>
```

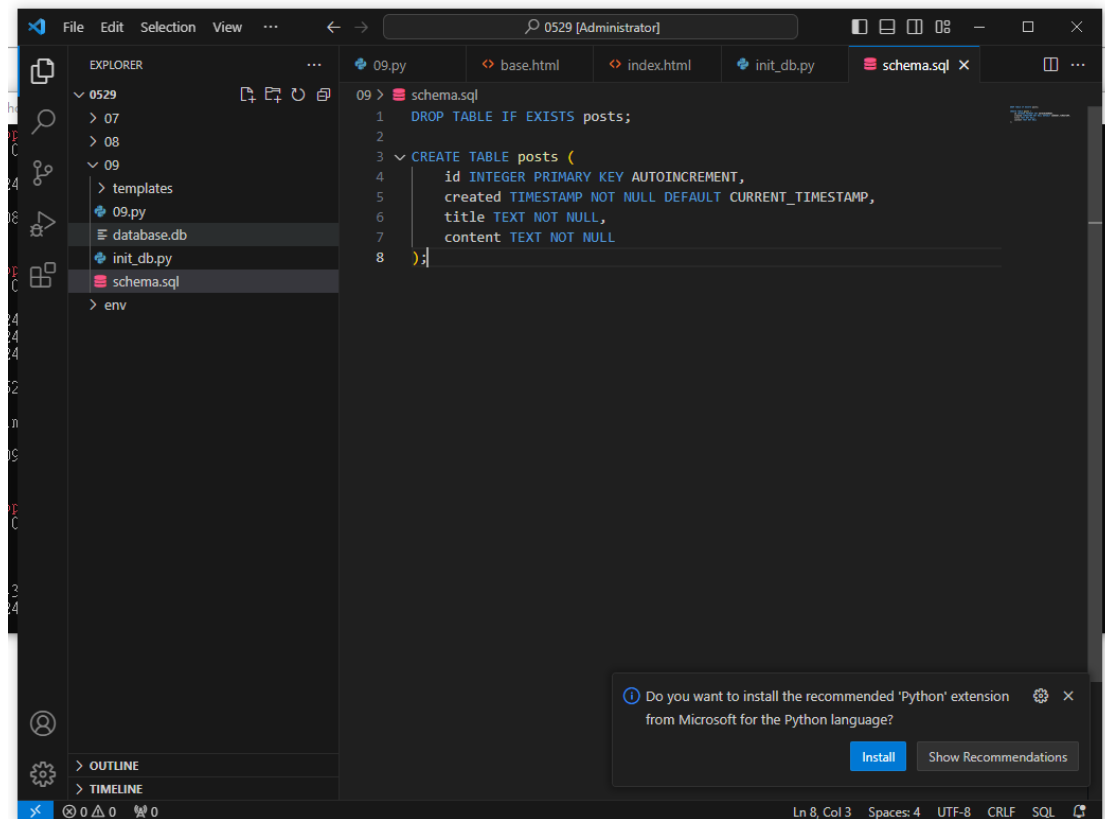
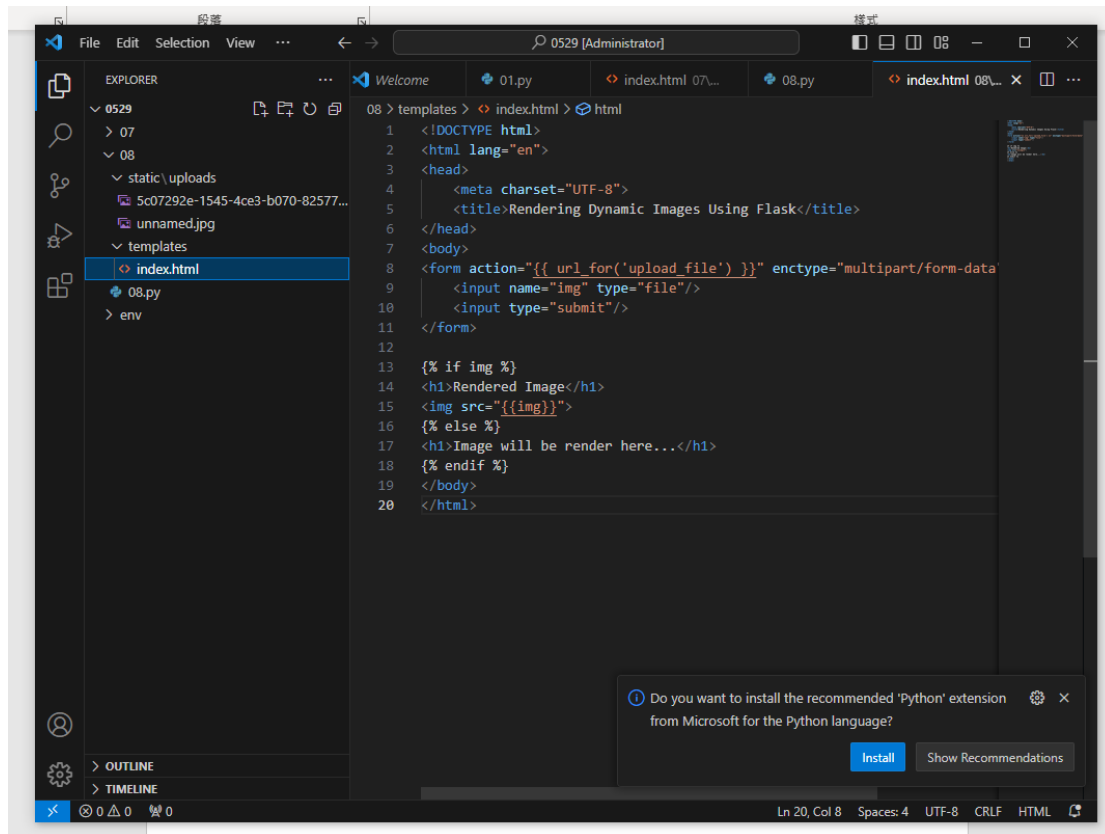
```
06 > templates > index.html > ...
1 {% extends "bootstrap/base.html" %}
2
3 {% block content %}
4 <h1>Hello, world!</h1>
5 {% endblock %}
```

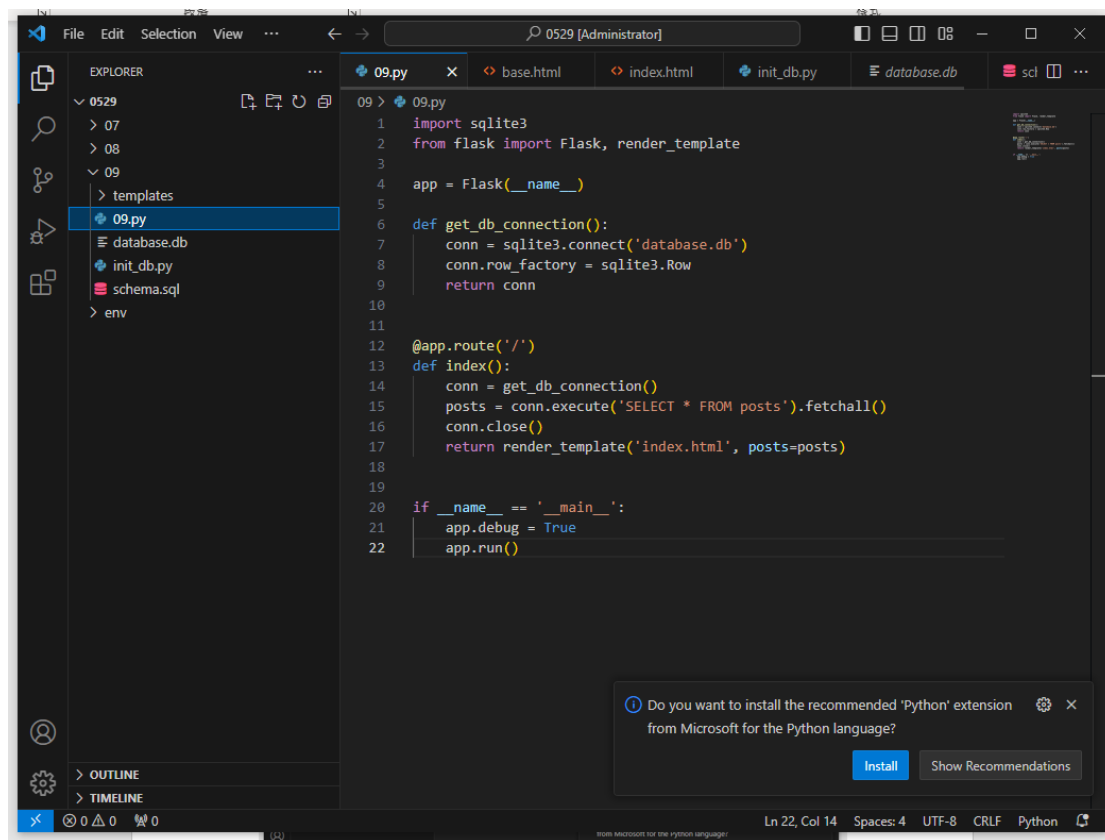
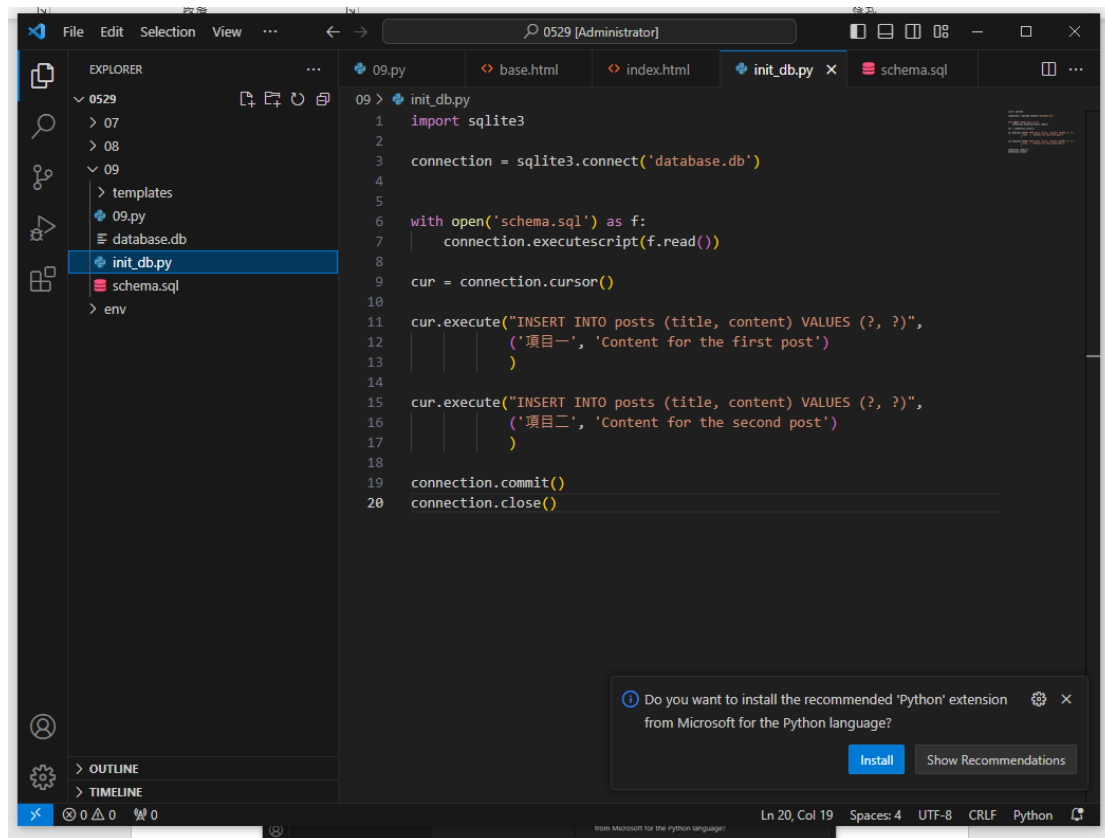


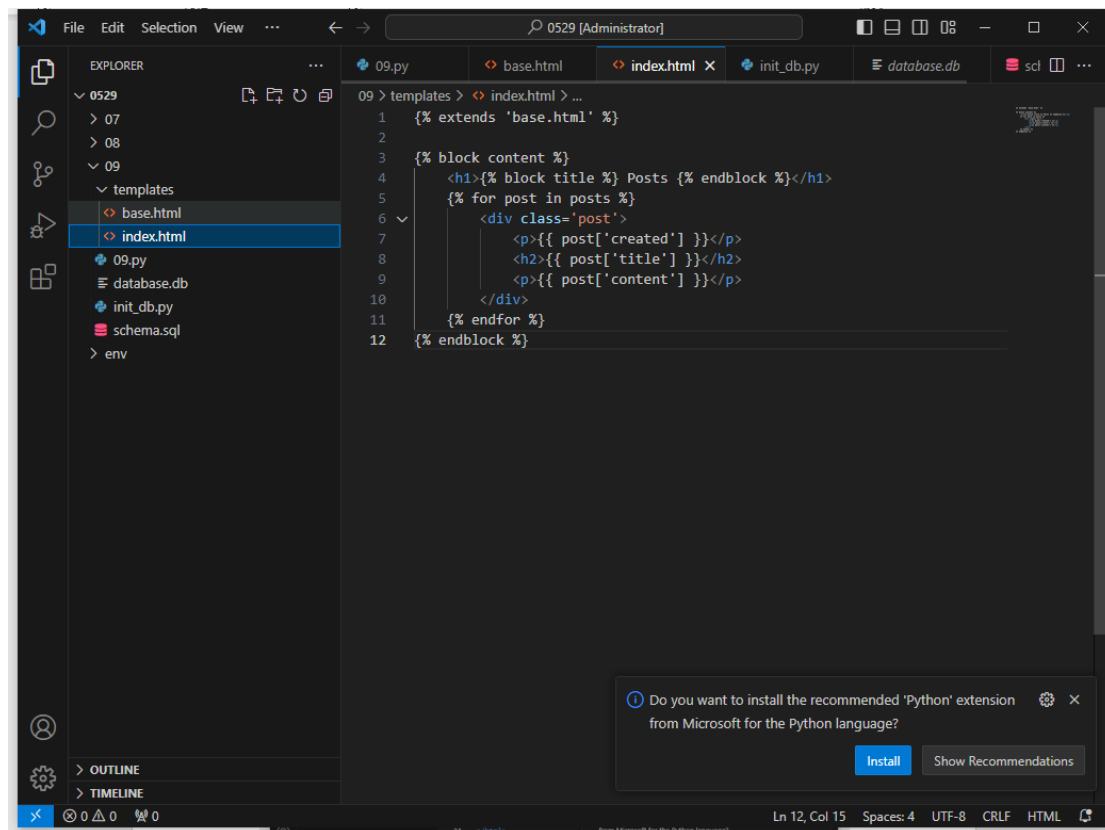
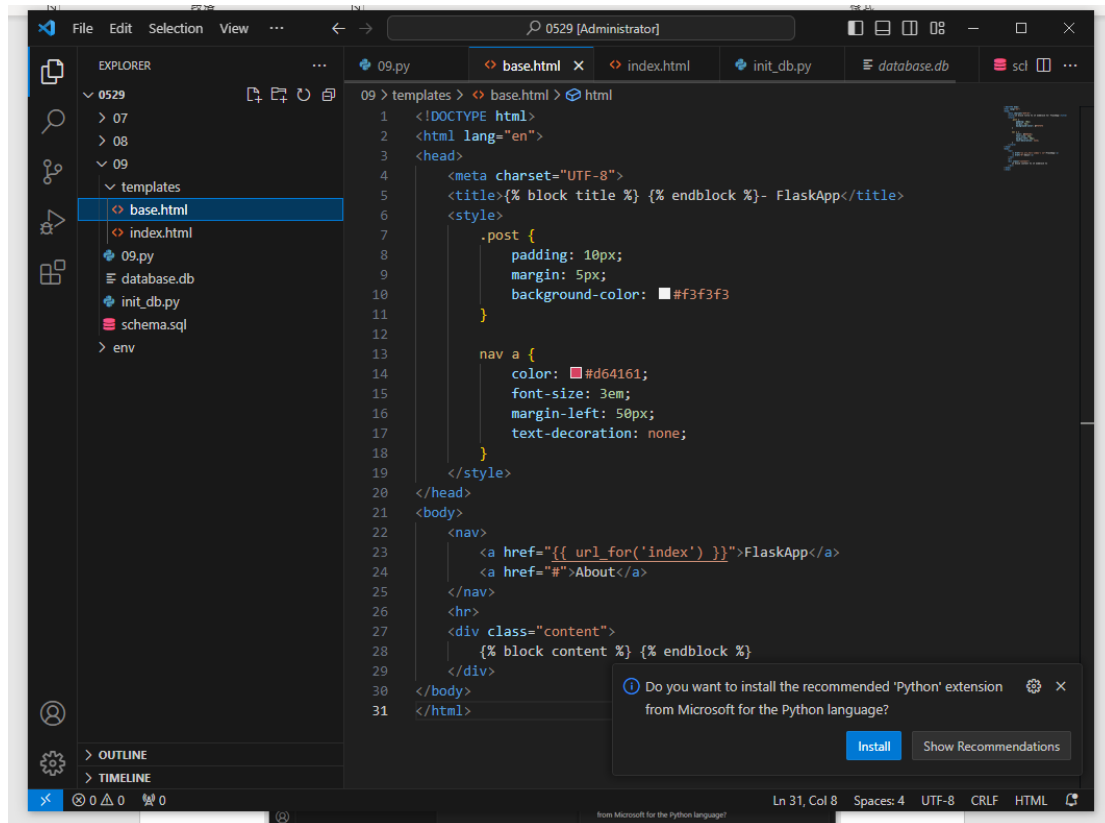


0529



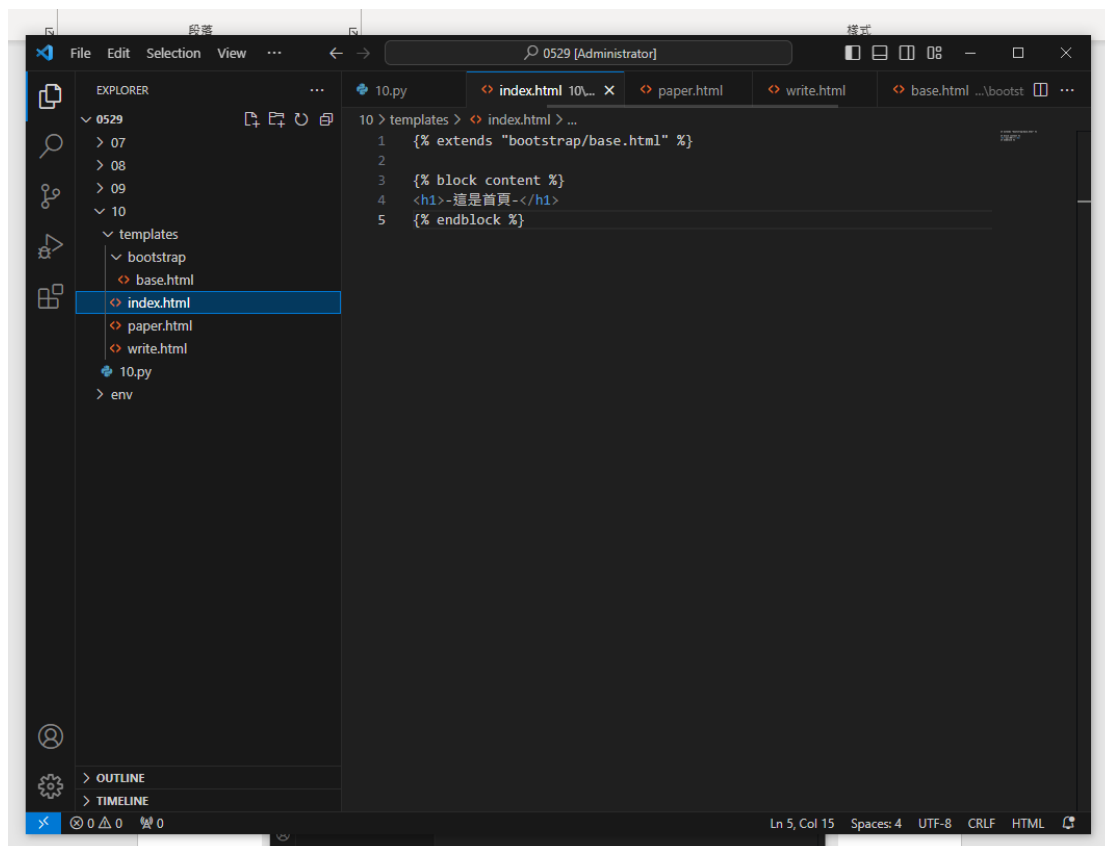
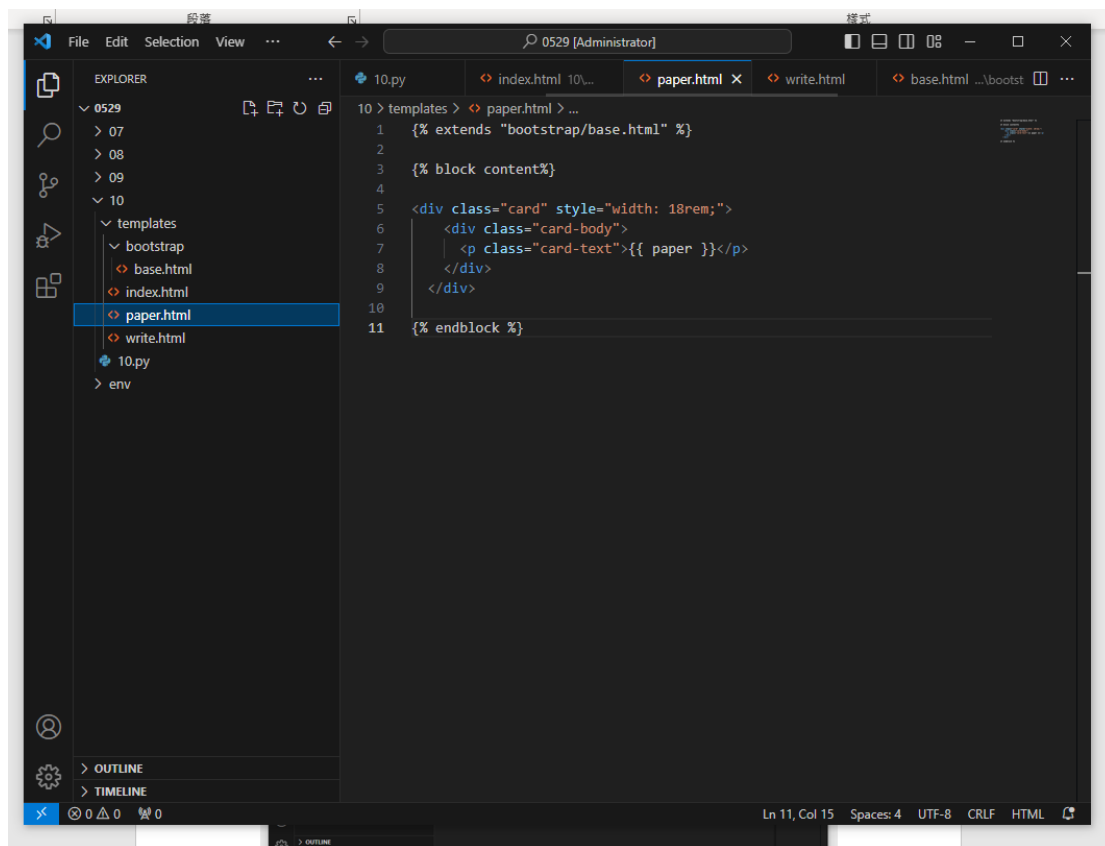







```
1 from flask import Flask, redirect, url_for, render_template, request, session
2 from flask_bootstrap import Bootstrap
3 app = Flask(__name__)
4 app.secret_key = "#230dec61-fee8-4ef2-a791-36f9e680c9fc"
5 bootstrap = Bootstrap(app=app)
6
7 @app.route('/')
8 def index():
9     return render_template('index.html')
10
11 @app.route("/write", methods=["POST", "GET"])
12 def write():
13     if request.method == "POST":
14         paper = request.form["nm"]
15         session["paper"] = paper
16         return redirect(url_for("paper"))
17     else:
18         return render_template("write.html")
19
20
21 @app.route("/paper")
22 def paper():
23     if "paper" in session:
24         paper = session["paper"]
25         return render_template("paper.html", paper=paper)
26     else:
27         return redirect(url_for("write"))
28
29
30 if __name__ == '__main__':
31     app.run()
```

```
1 {% extends "bootstrap/base.html" %}
2
3 {% block content %}
4 <form action="#" method="post">
5     <div class="form-group">
6         <label for="username">内容:</label>
7         <input id="username" class="form-control" type="text" name="nm" />
8     </div>
9     <div class="form-group">
10         <button type="submit" class="btn btn-dark">Submit</button>
11     </div>
12 </form>
13 {% endblock %}
```

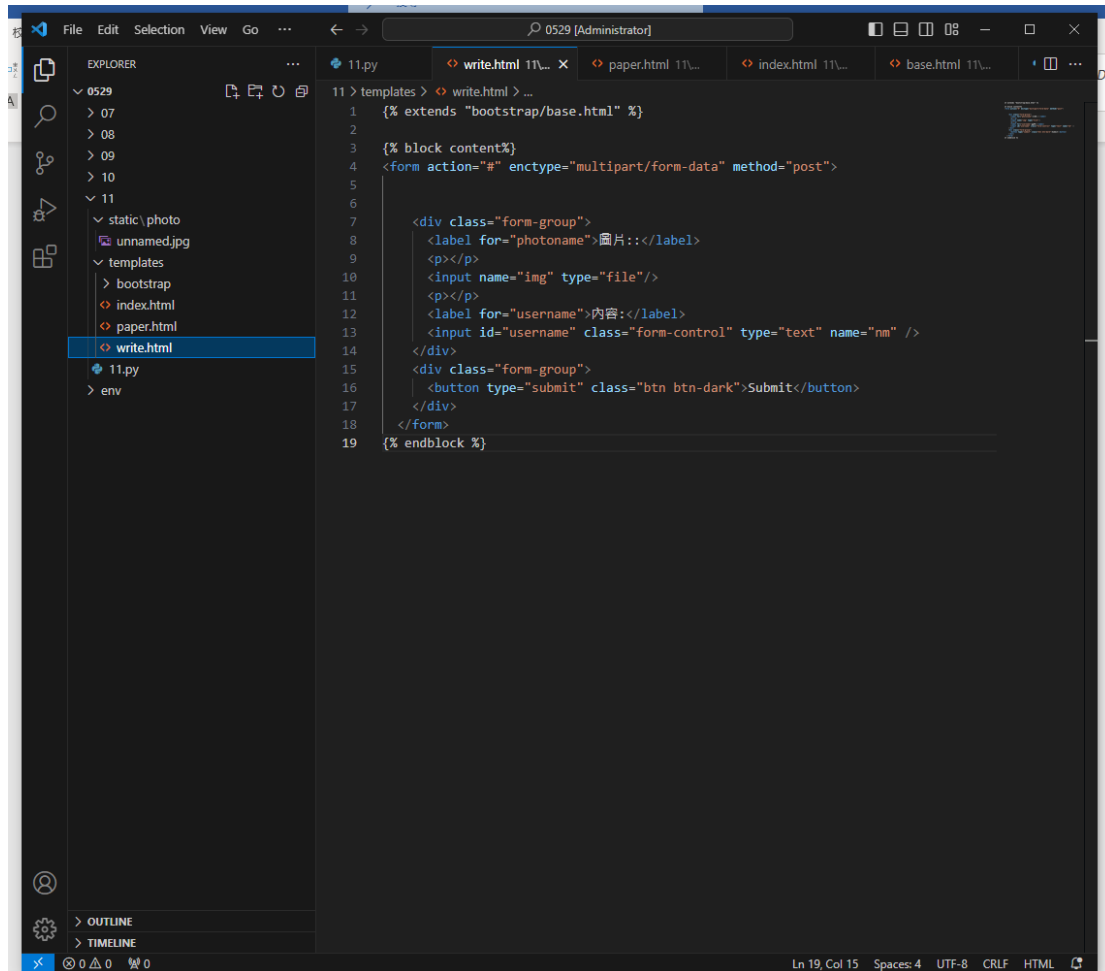


This screenshot shows the Visual Studio Code editor with the file explorer on the left displaying a project structure. The file explorer shows a directory named '0529' containing subdirectories '07', '08', '09', '10', and 'env'. The '10' directory contains 'templates', which in turn contains 'bootstrap'. The 'bootstrap' directory contains 'base.html', 'index.html', 'paper.html', and 'write.html'. The 'base.html' file is selected and its content is displayed in the main editor area. The code is an HTML template using Jinja2 syntax, featuring a navigation bar with a link to 'write.html' and a content block placeholder. Bootstrap CSS and JavaScript are included at the bottom.

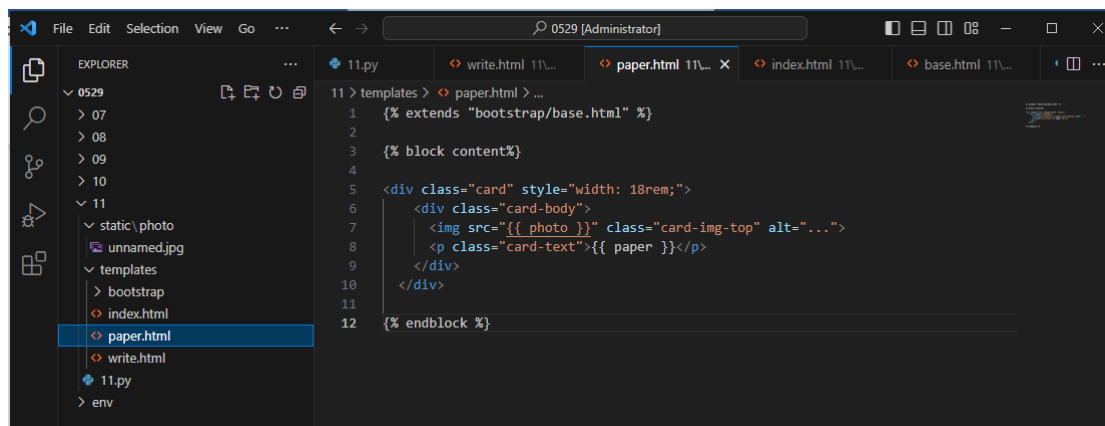
```
25 <a class="nav-link active" href="{{ url_for('write') }}">寫文章</a>
26
27 </div>
28
29
30
31 </div>
32 </div>
33 </nav>
34 {% endblock %}
35
36 <div class="content">
37     {% block content %}
38     {% endblock content %}
39 </div>
40
41
42 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.js"></script>
43 </body>
44 </html>
```

This screenshot shows the Visual Studio Code editor with the file explorer on the left displaying a project structure. The file explorer shows a directory named '0529' containing subdirectories '07', '08', '09', '10', and '11'. The '11' directory contains 'static', 'photo', 'unnamed.jpg', and 'templates'. The '11.py' file is selected and its content is displayed in the main editor area. The code is a Python Flask application that implements a simple file upload and display system. It uses Flask, Flask-Bootstrap, and Werkzeug. The application has three routes: a root route that renders 'index.html', a '/write' route that handles file uploads and saves them to a 'photo' directory, and a '/paper' route that renders 'paper.html' with the uploaded photo and filename.

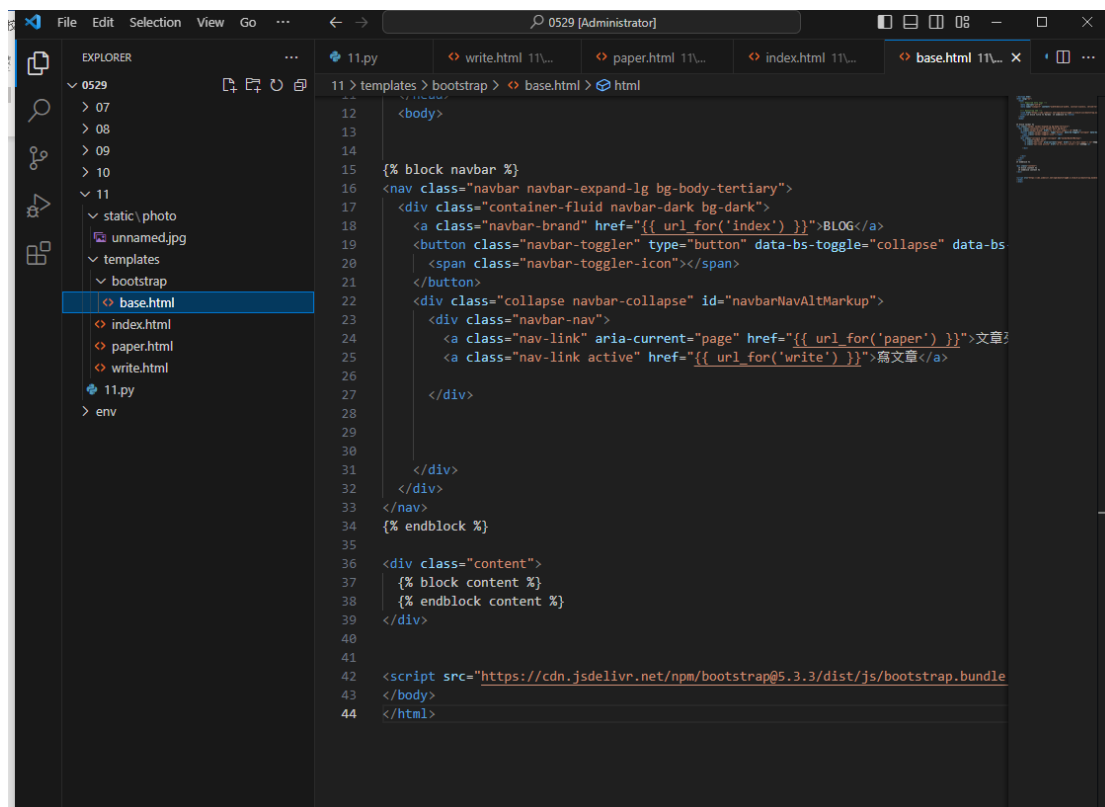
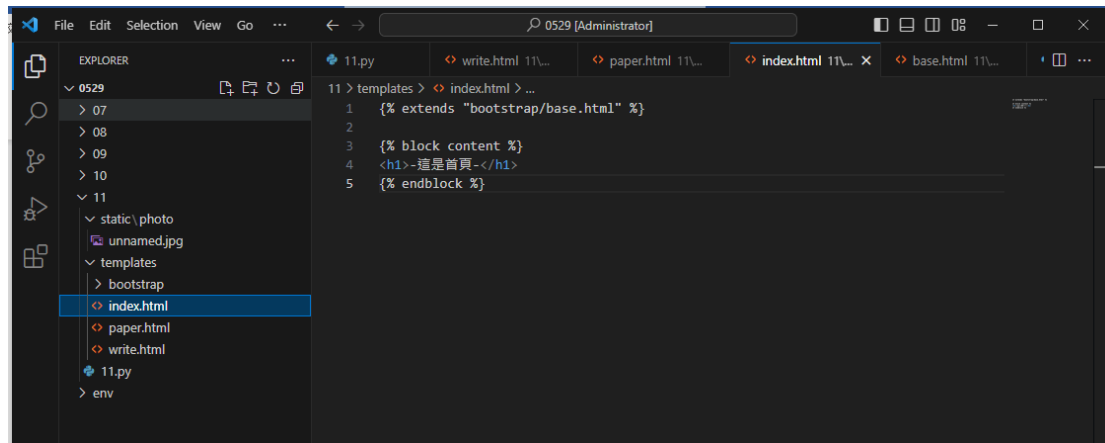
```
1 from flask import Flask, redirect, url_for, render_template, request, session
2 from flask_bootstrap import Bootstrap
3 from werkzeug.utils import secure_filename
4 import os
5
6 app = Flask(__name__)
7 app.secret_key = "#230dec61-fec8-4ef2-a791-36f9e680c9fc"
8 bootstrap = Bootstrap(app=app)
9 img = ''
10 upload_folder = os.path.join('static', 'photo')
11 app.config['UPLOAD'] = upload_folder
12 @app.route('/')
13 def index():
14     return render_template('index.html')
15
16 @app.route("/write", methods=["POST", "GET"])
17 def write():
18     global img
19     if request.method == "POST":
20         paper = request.form["nm"]
21         file = request.files['img']
22         filename = secure_filename(file.filename)
23         file.save(os.path.join(app.config['UPLOAD'], filename))
24         img = os.path.join(app.config['UPLOAD'], filename)
25         session["paper"] = paper
26         session["photo"] = filename
27         return redirect(url_for("paper"))
28     else:
29         return render_template("write.html")
30
31
32 @app.route("/paper")
33 def paper():
34     if "paper" or "photo" in session:
35         paper = session["paper"]
36         photo = session["photo"]
37         img = os.path.join(app.config['UPLOAD'], photo)
38         return render_template("paper.html", paper=paper, photo=img)
39     else:
40         return redirect(url_for("write"))
41
42
43
44 if __name__ == '__main__':
45     app.run()
```



```
11 > templates > write.html > ...
1  {% extends "bootstrap/base.html" %}
2
3  {% block content%}
4  <form action="#" enctype="multipart/form-data" method="post">
5
6
7      <div class="form-group">
8          <label for="photoname">图片:</label>
9          <p></p>
10         <input name="img" type="file"/>
11         <p></p>
12         <label for="username">内容:</label>
13         <input id="username" class="form-control" type="text" name="nm" />
14     </div>
15     <div class="form-group">
16         <button type="submit" class="btn btn-dark">Submit</button>
17     </div>
18 </form>
19 {% endblock %}
```



```
11 > templates > paper.html > ...
1  {% extends "bootstrap/base.html" %}
2
3  {% block content%}
4
5  <div class="card" style="width: 18rem;">
6      <div class="card-body">
7          
8          <p class="card-text">{{ paper }}</p>
9      </div>
10 </div>
11
12 {% endblock %}
```



```

1  from flask import Flask, render_template, request, url_for, flash, redirect, abort
2  import sqlite3
3  import datetime
4  from werkzeug.utils import secure_filename
5
6  import os
7  app = Flask(__name__)
8
9  app.secret_key = 'super secret key'
10 app.config['SESSION_TYPE'] = 'filesystem'
11 upload_folder = os.path.join('static', 'uploads')
12 app.config['UPLOAD'] = upload_folder
13 def get_db_connection():
14     conn = sqlite3.connect('database.db')
15     conn.row_factory = sqlite3.Row
16     return conn
17
18 Mnow = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")
19 @app.route('/')
20 def index():
21     conn = get_db_connection()
22     posts = conn.execute('SELECT * FROM COMPANY ORDER BY time DESC').fetchall()
23     conn.close()
24     return render_template('index.html', posts=posts)
25
26 @app.route('/create', methods=('GET', 'POST'))
27 def create():
28
29     if request.method == 'POST':
30         id = request.form['id']
31         name = request.form['name']
32         content = request.form['content']
33         image = request.files['image']
34         filename = secure_filename(image.filename)
35
36         if not id:
37             flash('id無填')
38         elif not name :
39             flash('name無填')
40         elif not content :
41             flash('content無填')
42         elif not filename:
43             nopic='no.jpg'
44             conn = get_db_connection()
45             conn.execute('INSERT INTO COMPANY (id, name, content, image, time ) VALUES
46                 (id, name, content, nopic, Mnow )')

```

The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with files like 07, 08, 09, templates, 09.py, database.db, init_db.py, schema.sql, 10, 11, 12, static/uploads, a.jpg, b.jpg, c.jpg, d.jpg, no.jpg, templates, base.html, create.html, edit.html, index.html, 12.py, database.db, env, OUTLINE, and TIMELINE. The code editor shows the following Python code:

```
46         conn.commit()(id, name, content, nopic, Mnow ))
47     conn.close()
48     return redirect(url_for('index'))
49
50     else:
51         image.save(os.path.join(app.config['UPLOAD'], filename))
52         conn = get_db_connection()
53         conn.execute('INSERT INTO COMPANY (id, name, content, image, time ) VALUES
54             (id, name, content, filename, Mnow ))
55         conn.commit()
56         conn.close()
57         return redirect(url_for('index'))
58
59     return render_template('create.html')
60
61
62 def get_post(post_id):
63     conn = get_db_connection()
64     post = conn.execute('SELECT * FROM COMPANY WHERE id = ?',
65                         (post_id,)).fetchone()
66     conn.close()
67     if post is None:
68         abort(404)
69     return post
70
71 @app.route('/<int:id>/edit/', methods=('GET', 'POST'))
72 def edit(id):
73
74     post = get_post(id)
75
76     if request.method == 'POST':
77         name = request.form['name']
78         content = request.form['content']
79         image = request.files['image']
80         filename = secure_filename(image.filename)
81
82         print(filename)
83         if not filename:
84             conn = get_db_connection()
85             conn.execute('UPDATE COMPANY SET name = ?, content = ?'
86                         ' WHERE id = ?',
87                         ( name, content, id))
88
89             conn.commit()
90             conn.close()
91             return redirect(url_for('index'))
```

The status bar at the bottom shows 'Ln 117, Col 14', 'Spaces: 4', 'UTF-8', 'CRLF', and 'Python'.

```

12.py
def edit(id):
    post = get_post(id)

    if request.method == 'POST':
        name = request.form['name']
        content = request.form['content']
        image = request.files['image']
        filename = secure_filename(image.filename)

        print(filename)
        if not filename:
            conn = get_db_connection()
            conn.execute('UPDATE COMPANY SET name = ?, content = ?'
                          ' WHERE id = ?',
                          ( name, content, id))

            conn.commit()
            conn.close()
            return redirect(url_for('index'))

        else:
            image.save(os.path.join(app.config['UPLOAD'], filename))
            conn = get_db_connection()
            conn.execute('UPDATE COMPANY SET name = ?, content = ?, image = ?'
                          ' WHERE id = ?',
                          ( name, content, filename, id))

            conn.commit()
            conn.close()
            return redirect(url_for('index'))

    return render_template('edit.html', post=post)

@app.route('/<int:id>/delete/', methods=('POST',))
def delete(id):
    post = get_post(id)
    conn = get_db_connection()
    conn.execute('DELETE FROM COMPANY WHERE id = ?', (id,))
    conn.commit()
    conn.close()
    return redirect(url_for('index'))

if __name__ == '__main__':
    app.run()

```

Ln 117, Col 14 Spaces: 4 UTF-8 CRLF Python

File Edit Selection View Go ... 0529 [Administrator]

EXPLORER

- 0529
 - 07
 - 08
 - 09
 - templates
 - 09.py
 - database.db
 - init_db.py
 - schema.sql
 - 10
 - 11
 - 12
 - static\uploads
 - a.jpg
 - b.jpg
 - c.jpg
 - d.jpg
 - no.jpg
 - templates
 - base.html
 - create.html
 - edit.html
 - index.html
 - 12.py
 - database.db
 - env

12 > templates > index.html > ...

```
1 {% extends 'base.html' %}
2 {% block content %}
3     <div class="container">
4         <h1>{% block title %} -文章列表- {% endblock %}</h1>
5         <table class="table table-striped table-hover">
6             <tr class="table-dark">
7                 <td>編號</td>
8                 <td>標題</td>
9                 <td>內容</td>
10                <td>圖片</td>
11                <td>編輯</td>
12                <td>刪除</td>
13            </tr>
14            {% for post in posts %}
15            <tr class="table-primary">
16                <td>{{ post['id'] }}</td>
17                <td>{{ post['name'] }}</td>
18                <td>{{ post['content'] }}</td>
19                <td></td>
20                <td><a href="{{ url_for('edit', id=post['id']) }}" class="link-dark">編輯</a>
21                <td><form action="{{ url_for('delete', id=post['id']) }}" method="POST">
22                    <input type="submit" value="刪除"
23                    onclick="return confirm('確定刪除?')"
24                    class="btn btn-primary mb-3">
25                </form></td>
26            </tr>
27            {% endfor %}
28        </table>
29    </div>
30 {% endblock %}
```

File Edit Selection View Go ... 0529 [Administrator]

EXPLORER

- 0529
 - 07
 - 08
 - 09
 - templates
 - 09.py
 - database.db
 - init_db.py
 - schema.sql
 - 10
 - 11
 - 12
 - static\uploads
 - a.jpg
 - b.jpg
 - c.jpg
 - d.jpg
 - no.jpg
 - templates
 - base.html
 - create.html
 - edit.html
 - index.html
 - 12.py
 - database.db
 - env

12 > templates > edit.html > ...

```
1 {% extends 'base.html' %}
2 {% block content %}
3     <div class="container">
4         <h1>{% block title %} -編輯- "{{ post['name'] }}" {% endblock %}</h1>
5         <form enctype="multipart/form-data" method="POST">
6             <div class="mb-3">
7                 <label for="name" class="form-label">標題</label>
8                 <input type="text" class="form-control" name="name" value="{{ request.form.name }}" />
9             </div>
10            <div class="mb-3">
11                <label for="content" class="form-label">內容</label>
12                <input type="text" class="form-control" name="content" value="{{ request.form.content }}" />
13            </div>
14            <div class="mb-3">
15                
16            </div>
17            <div class="mb-3">
18                <label for="image" class="form-label">圖片</label>
19                <input type="file" name="image" />
20            </div>
21            <div class="mb-3">
22                <button type="submit" class="btn btn-primary mb-3">Submit</button>
23            </div>
24        </form>
25    </div>
26 {% endblock %}
```

```
File Edit Selection View Go ... 0529 [Administrator] database.db 12 base.html 12... create.html x edit.html index.html 12...

EXPLORER
0529
> 07
> 08
> 09
  templates
    09.py
    database.db
    init_db.py
    schema.sql
  10
  11
  12
    static\uploads
      a.jpg
      b.jpg
      c.jpg
      d.jpg
      no.jpg
    templates
      base.html
      create.html
      edit.html
      index.html
      12.py
      database.db
    env

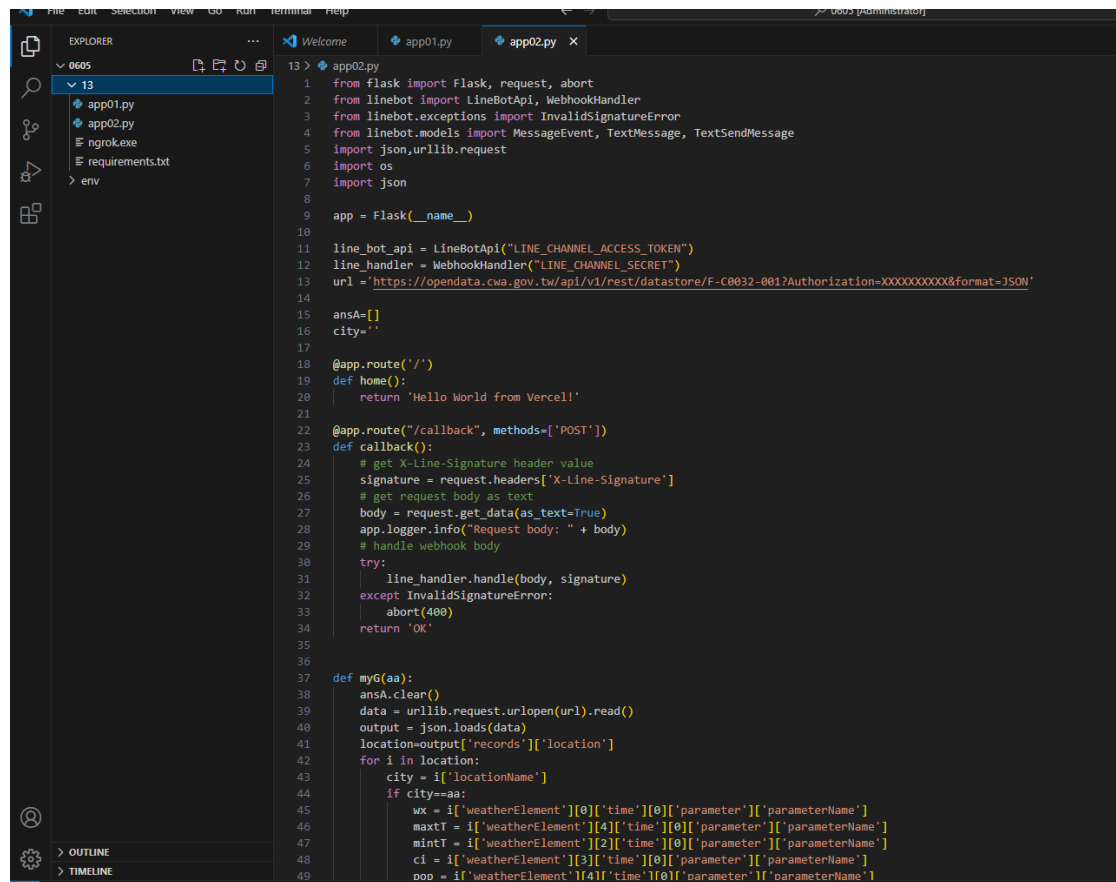
12 > templates > create.html > ...
1  {% extends 'base.html' %}
2  {% block content %}
3
4  {% with messages = get_flashed_messages() %}
5  {% if messages %}
6  <ul class="flash">
7      {% for message in messages %}
8          <li>{{ message }}</li>
9      {% endfor %}
10 </ul>
11 {% endif %}
12 {% endwith %}
13
14
15 <div class="container">
16 <h1>{{ block title %}} - 新增文章</h1>
17 <form enctype="multipart/form-data" method="POST">
18     <div class="mb-3">
19         <label for="id" class="form-label">編號</label>
20         <input type="text" class="form-control" name="id" value="{{ request.f"
21     </div>
22     <div class="mb-3">
23         <label for="name" class="form-label">標題</label>
24         <input type="text" class="form-control" name="name" value="{{ request."
25     </div>
26     <div class="mb-3">
27         <label for="content" class="form-label">內容</label>
28         <input type="text" class="form-control" name="content" value="{{ reque"
29     </div>
30     <div class="mb-3">
31         <label for="image">圖片</label>
32         <input name="image" type="file"/>
33     </div>
34     <div class="mb-3">
35         <button type="submit" class="btn btn-primary mb-3">Submit</button>
36     </div>
37 </form>
38 </div>
39
40 {% endblock %}
```

```
File Edit Selection View Go ... 0529 [Administrator] database.db 12 base.html 12... create.html edit.html index.html 12...

EXPLORER
0529
> 07
> 08
> 09
  templates
    09.py
    database.db
    init_db.py
    schema.sql
  10
  11
  12
    static\uploads
      a.jpg
      b.jpg
      c.jpg
      d.jpg
      no.jpg
    templates
      base.html
      create.html
      edit.html
      index.html
      12.py
      database.db
    env

12 > templates > base.html > ...
2 <!DOCTYPE html>
3 <html>
4 <head>
5     <meta charset="UTF-8">
6     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet">
7     <title>{{ block title %}} DEMO</title>
8     <style>
9         .nav-link{
10             color: #255, 255, 255;
11         }
12     </style>
13 </head>
14 <body>
15
16     {% block navbar %}
17     <nav class="navbar navbar-expand-lg" style="background-color: #288a9b;">
18
19         <div class="container-fluid">
20             <a class="navbar-brand" href="{{ url_for('index') }}">-BLOG-</a>
21             <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav"
22                 <span class="navbar-toggler-icon"></span>
23             </button>
24             <div class="collapse navbar-collapse" id="navbarNav">
25                 <ul class="navbar-nav">
26                     <li class="nav-item">
27                         <a class="nav-link" href="{{ url_for('index') }}">文章列表</a>
28                     </li>
29                     <li class="nav-item">
30                         <a class="nav-link" href="{{ url_for('create') }}">新增文章</a>
31                     </li>
32                 </ul>
33             </div>
34         </div>
35     </nav>
36     {% endblock navbar %}
37
38     <div class="content">
39         {% block content %} {% endblock %}
40     </div>
41
42     {% block scripts %}
43     <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js" rel="stylesheet">
44     </script>
45     </body>
46 </html>
47 {% endblock doc %}
```

0605



```
13 > app02.py
1  from flask import Flask, request, abort
2  from linebot import LineBotApi, WebhookHandler
3  from linebot.exceptions import InvalidSignatureError
4  from linebot.models import MessageEvent, TextMessage, TextSendMessage
5  import json,urllib.request
6  import os
7  import json
8
9  app = Flask(__name__)
10
11  line_bot_api = LineBotApi("LINE_CHANNEL_ACCESS_TOKEN")
12  line_handler = WebhookHandler("LINE_CHANNEL_SECRET")
13  url = 'https://opendata.cwa.gov.tw/api/v1/rest/datastore/F-C0032-001?Authorization=XXXXXXXXXX&format=JSON'
14
15  ansA=[]
16  city=""
17
18  @app.route('/')
19  def home():
20      return 'Hello World from Vercel!'
21
22  @app.route("/callback", methods=['POST'])
23  def callback():
24      # get X-Line-Signature header value
25      signature = request.headers['X-Line-Signature']
26      # get request body as text
27      body = request.get_data(as_text=True)
28      app.logger.info("Request body: " + body)
29      # handle webhook body
30      try:
31          line_handler.handle(body, signature)
32      except InvalidSignatureError:
33          abort(400)
34      return 'OK'
35
36
37  def myG(aa):
38      ansA.clear()
39      data = urllib.request.urlopen(url).read()
40      output = json.loads(data)
41      location=output['records']['location']
42      for i in location:
43          city = i['locationName']
44          if city==aa:
45              wx = i['weatherElement'][0]['time'][0]['parameter']['parameterName']
46              maxtI = i['weatherElement'][4]['time'][0]['parameter']['parameterName']
47              mintI = i['weatherElement'][2]['time'][0]['parameter']['parameterName']
48              ci = i['weatherElement'][3]['time'][0]['parameter']['parameterName']
49              ooo = i['weatherElement'][4]['time'][0]['parameter']['parameterName']
```

```
51         ansA.append(city)
52         ansA.append(wx)
53         ansA.append(mintT)
54         ansA.append(maxtT)
55     return ansA
56
57
58
59 @line_handler.add(MessageEvent, message=TextMessage)
60 def handle_message(event):
61
62     if event.message.text == "1":
63         line_bot_api.reply_message(
64             event.reply_token,
65             TextSendMessage(text=str(myG('臺北市'))))
66
67     elif event.message.text == "2":
68         line_bot_api.reply_message(
69             event.reply_token,
70             TextSendMessage(text=str(myG('桃園市'))))
71
72     elif event.message.text == "3":
73         line_bot_api.reply_message(
74             event.reply_token,
75             TextSendMessage(text=str(myG('新竹市'))))
76
77     elif event.message.text == "4":
78         line_bot_api.reply_message(
79             event.reply_token,
80             TextSendMessage(text=str(myG('臺中市'))))
81
82     elif event.message.text == "5":
83         line_bot_api.reply_message(
84             event.reply_token,
85             TextSendMessage(text=str(myG('臺南市'))))
86
87     elif event.message.text == "6":
88         line_bot_api.reply_message(
89             event.reply_token,
90             TextSendMessage(text=str(myG('高雄市'))))
91
92     else:
93         line_bot_api.reply_message(
94             event.reply_token,
95             TextSendMessage(text="只有1-6的數字"))
96
97 if __name__ == "__main__":
98     app.run()
```

```
File Edit Selection View Go Run Terminal Help
0605 [A]

EXPLORER
0605
  13
    app01.py
    app02.py
    ngrok.exe
    requirements.txt
    env

OUTLINE
TIMELINE

13 > app01.py
1  from flask import Flask, request, abort
2  from linebot import LineBotApi, WebhookHandler
3  from linebot.exceptions import InvalidSignatureError
4  from linebot.models import MessageEvent, TextMessage, TextSendMessage
5  import random
6  app = Flask(__name__)
7
8  line_bot_api = LineBotApi('LINE_CHANNEL_ACCESS_TOKEN')
9  line_handler = WebhookHandler('LINE_CHANNEL_SECRET')
10
11 @app.route('/')
12 def home():
13     return 'Hello World'
14
15 @app.route("/callback", methods=['POST'])
16 def callback():
17     # get X-Line-Signature header value
18     signature = request.headers['X-Line-Signature']
19     # get request body as text
20     body = request.get_data(as_text=True)
21     app.logger.info("Request body: " + body)
22     # handle webhook body
23     try:
24         line_handler.handle(body, signature)
25     except InvalidSignatureError:
26         abort(400)
27     return 'OK'
28
29
30 def myG():
31     myW=['你好', '吃飽了嗎', '天氣如何']
32     return random.choice(myW)
33
34 @line_handler.add(MessageEvent, message=TextMessage)
35 def handle_message(event):
36     getA=event.message.text
37
38     if getA == '0' :
39         line_bot_api.reply_message(
40             event.reply_token,
41             TextSendMessage(text=str(myG())))
42
43     else:
44         line_bot_api.reply_message(
45             event.reply_token,
46             TextSendMessage(text="輸入0"))
47
48 if __name__ == "__main__":
49     app.run()
```