掲示板を作ろう

この教材でできること

・htmlの連携ができる

・DBとのやり取りができる

どんな教材？

DBに入っている内容をweb上で出力したり、編集したりできるようになる。

目次

① コードを書く前の準備をしよう

②完成系を確認してみよう

③コードを書いてみよう

④答え合わせをしてみよう

目次

① コードを書く前の準備をしよう

このページでは環境設定などの準備をするためのページ

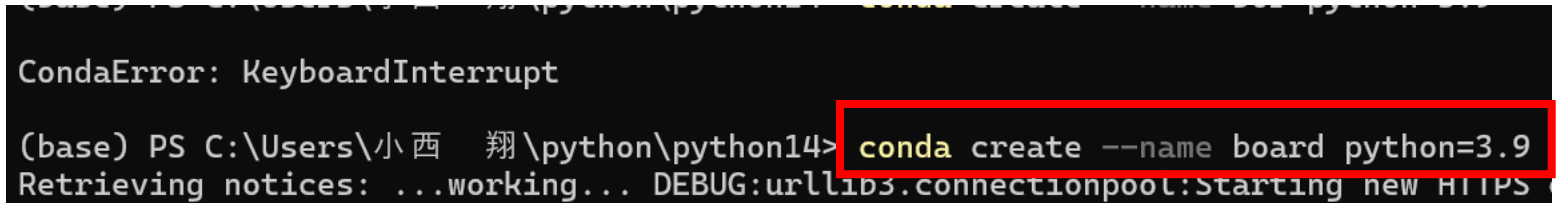
① コードを書く前の準備をしよう

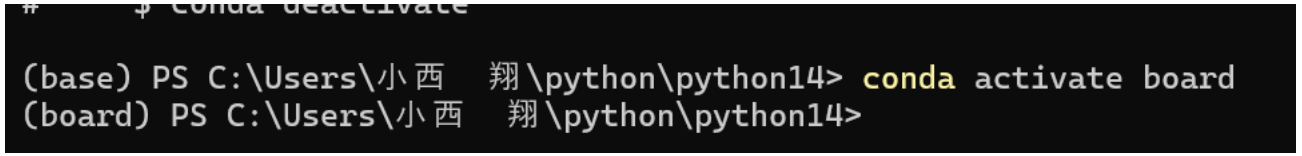
このページでは掲示板の情報を保存するDBを準備する

	#	名前	タイプ	照合順序	属性	Null	デフォルト値	コメント	その他	操作
☐	1	id	int(5)			いいえ	なし			変更 削除 その他
☐	2	title	varchar(30)	utf8mb4_general_ci		いいえ	なし			変更 削除 その他
☐	3	content	varchar(50)	utf8mb4_general_ci		いいえ	なし			変更 削除 その他
☐	4	created	date			いいえ	なし			変更 削除 その他

① コードを書く前の準備をしよう

このページでは環境設定や使うファイル、ライブラリなどの準備を行う





① コードを書く前の準備をしよう

ライブラリは必要であればその都度追加しよう！

vscodeを開いて、コードを書いていこう！

目次

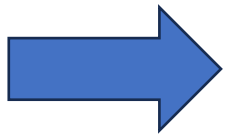
②完成系を確認してみよう

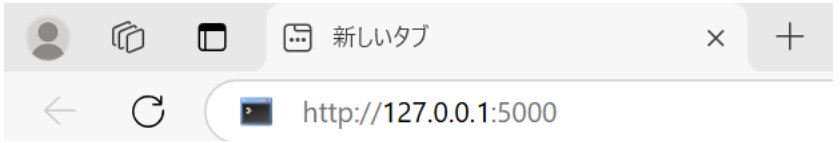
このページでは完成系を見てどのようなコードを書いたらいいかをイメージしてみよう

②完成系を確認してみよう

httpの後の文字をコピーしてwebに張り付けてエンターキーを押す

```
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do n
tion deployment. Use a production WSGI serv
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 399-545-312
```





②完成系を確認してみよう

htmlで投稿できるページにする

②完成系を確認してみよう

タイトルと内容を入力して投稿ボタンを押す

②完成系を確認してみよう

新しい投稿が下に出てくるようにする

目次

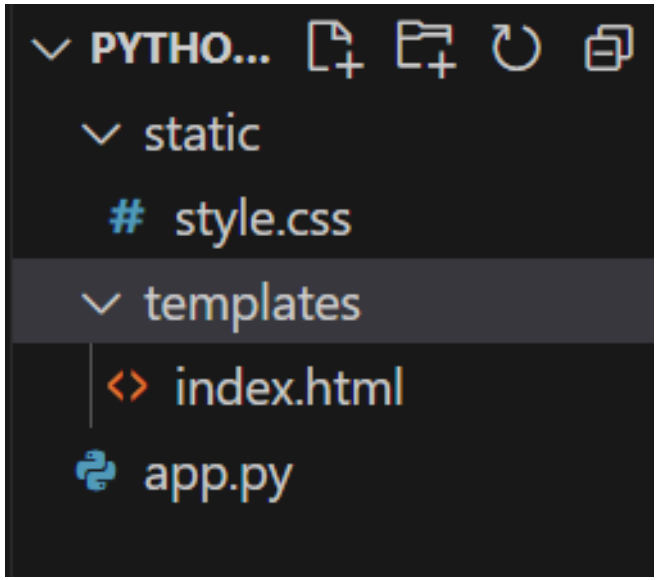
③コードを書いてみよう

このページではどのようなコードを書いたらいいかヒントが書かれているよ

③コードを書いてみよう

ファイル、フォルダーは写真のようになっているよ

staticの中にcssコード、templatesの中にhtmlコードが入っているよ



③コードを書いてみよう

htmlのコードはまずformタグを作り、投稿のタイトル、内容とボタンを作ろう

③コードを書いてみよう

次に投稿一覧でPythonの内容から送られた変数をhtmlで
表示するようにする

③コードを書いてみよう

PythonではDBとの連携をする

③コードを書いてみよう

DBの内容を読み取り、変数にしてhtmlに送る

③コードを書いてみよう

POSTで送られてきた内容からDBに保存する

③コードを書いてみよう

デザインが気になったら自分でcssのコードを書いてみよう

目次

① コードを書く前の準備をしよう

②完成系を確認してみよう

③コードを書いてみよう

④答え合わせをしてみよう

④答え合わせをしてみよう

答えのコードを確認してどのようにしているか見れるよ

④答え合わせをしてみよう

htmlのヘッドの部分と入力部分を書いていこう

templates > index.html > ...

```html
<!DOCTYPE html>
<html lang="ja">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>掲示板</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
    <h1>掲示板</h1>

    <form action="/add" method="POST">
        <input type="text" name="title" placeholder="タイトル" required>
        <textarea name="content" placeholder="内容" required></textarea>
        <button type="submit">投稿</button>
    </form>

    <h2>投稿一覧</h2>
    <ul>
        {% for post in posts %}
            <li>
                <h3>{{ post.title }}</h3>
                <p>{{ post.content }}</p>
                <small>{{ post.created.strftime('%Y-%m-%d %H:%M') }}</small>
            </li>
        {% endfor %}
    </ul>
</body>
</html>
```

④答え合わせをしてみよう

ライブラリを入力しよう

必要であればその都度インストールをしよう

app.py > ...

```
1   from flask import Flask, render_template, request, redirect
2   import pymysql
3   from datetime import datetime
4
```

④答え合わせをしてみよう

DBの連携をしよう

```
4
5   app = Flask(__name__)
6
7   # データベース接続設定
8   def get_db_connection():
9       connection = pymysql.connect(
10          host='localhost',
11          user='root',
12          password='',  # パスワードが設定されている場合はここに記入
13          database='python_test',
14          charset='utf8mb4',
15          cursorclass=pymysql.cursors.DictCursor
16      )
17      return connection
```

④答え合わせをしてみよう

DBの内容を読み取り、変数に代入しよう

```
18
19  # 投稿一覧の表示
20  @app.route('/')
21  def index():
22      connection = get_db_connection()
23      try:
24          with connection.cursor() as cursor:
25              cursor.execute("SELECT * FROM board ORDER BY created DESC")
26              posts = cursor.fetchall()
27          return render_template('index.html', posts=posts)
28      finally:
29          connection.close()
```

④答え合わせをしてみよう

投稿の内容をDBに保存しよう

```
31  # 新規投稿の追加
32  @app.route('/add', methods=['POST'])
33  def add_post():
34      title = request.form['title']
35      content = request.form['content']
36      created = datetime.now()
37
38      connection = get_db_connection()
39      try:
40          with connection.cursor() as cursor:
41              sql = "INSERT INTO board (title, content, created) VALUES (%s, %s, %s)"
42              cursor.execute(sql, (title, content, created))
43              connection.commit()
44          return redirect('/')
45      finally:
46          connection.close()
47
48  if __name__ == '__main__':
49      app.run(debug=True)
50
```

④答え合わせをしてみよう

余力があればcssでデザインを作ろう

```css
body {
    font-family: Arial, sans-serif;
    background-color: #f4f4f4;
    margin: 0;
    padding: 20px;
}

h1 {
    text-align: center;
    color: #333;
}

form {
    background: #fff;
    padding: 20px;
    margin-bottom: 20px;
    border-radius: 5px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}

form input, form textarea {
    width: 100%;
    padding: 10px;
    margin: 10px 0;
    border: 1px solid #ccc;
    border-radius: 3px;
}
```

```css
form button {
    background-color: #007BFF;
    color: white;
    border: none;
    padding: 10px 15px;
    border-radius: 3px;
    cursor: pointer;
}

form button:hover {
    background-color: #0056b3;
}

ul {
    list-style-type: none;
    padding: 0;
}

ul li {
    background: #fff;
    margin-bottom: 15px;
    padding: 15px;
    border-radius: 5px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}
```

```css
ul li h3 {
    margin: 0 0 10px 0;
    color: #007BFF;
}

ul li p {
    margin: 0 0 10px 0;
    color: #333;
}

ul li small {
    color: #666;
    font-size: 0.9em;
}
```

お疲れさまでした

テキストは終了です。
あとは自分なりにアレンジを付け加えていこう！