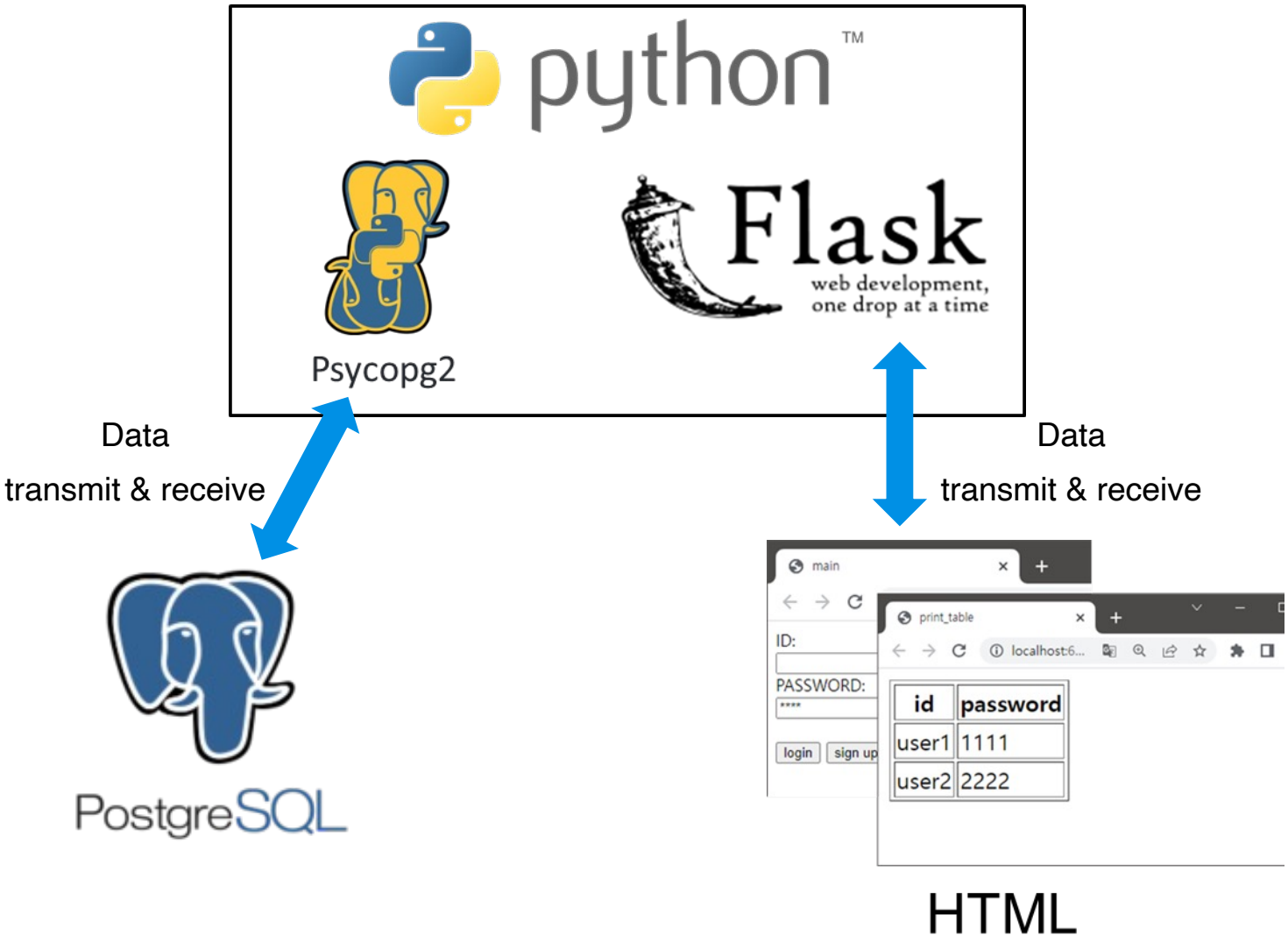
KOREA UNIVERSITY
DATABASE LAB

DB Application Programming - Lab

Tutorial

Overview


python
Psycopg2
Flask
web development,
one drop at a time
Data
transmit & receive
Data
transmit & receive
PostgreSQL
main
print_table
ID:
PASSWORD:
login
sign up
id
password
user1
1111
user2
2222
HTML

Connect to PostgreSQL with Psycopg2

1. Create new database “tutorial”

```
postgres=# create database tutorial;
CREATE DATABASE
postgres=# \c tutorial
You are now connected to database "tutorial" as user "term".
tutorial=#
```

2. Open psycopg2_test1.py and set parameters

psycopg2_test1.py

```
1  import psycopg2
2
3  connect = psycopg2.connect("dbname=tutorial user=term password=term")
```

your PostgreSQL password

Write data to PostgreSQL

psycopg2_test1.py

```python
import psycopg2

connect = psycopg2.connect("dbname=tutorial user=term password=term")
cur = connect.cursor()  # create cursor

cur.execute("CREATE TABLE users (id varchar(20) not null, password varchar(20) not null, primary key(id));")
cur.execute("INSERT INTO users VALUES('dbstudent0', '0000');")

id = 'database'
password = 'postgres'
cur.execute("INSERT INTO users VALUES('{}', '{}');".format( *args: id, password))

connect.commit()  # you must use connect.commit() when write data to PostgreSQL
```

Run

```
tutorial=# \d
       List of relations
 Schema | Name  | Type  | Owner
--------+-------+-------+-------
 public | users | table | term
(1 row)

tutorial=# select * from users;
     id     | password
------------+----------
 dbstudent0 | 0000
 database   | postgres
(2 rows)
```

Read data from PostgreSQL

Open psycopg2_test2.py and set parameters

psycopg2_test2.py

```
1   import psycopg2
2
3   connect = psycopg2.connect("dbname=tutorial user=term password=term")
4   cur = connect.cursor()  # create cursor
5
6   cur.execute("SELECT * FROM users;")
7   result = cur.fetchall()  # you must use cur.fetchall() when read data from PostgreSQL
8   print(result)
9   print("=====")
10  print(result[0])
11  print("=====")
12  print(result[0][0])
```

Run

```
[('dbstudent0', '0000'), ('database', 'postgres')]
=====
('dbstudent0', '0000')
=====
dbstudent0
```

Basic Flask App

flask_test1.py

```
1   from flask import Flask
2
3   app = Flask(__name__)
4
5   ①@app.route('/')
6   ②def main():
7       return "Hello World!"
8
9   # @app.route('/bye')
10  # def bye():
11  #     return "Goodbye World!"
12
13  if __name__ == '__main__':
14      app.run()
```

Open and run “flask_test1.py” file

① URL routing

② What to do in ①'s URL

Basic Flask App

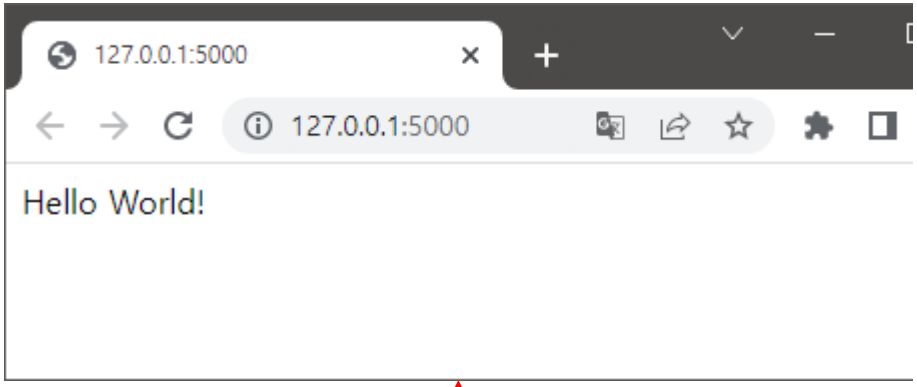
flask_test1.py

```python
from flask import Flask

app = Flask(__name__)

@app.route('/')
def main():
    return "Hello World!"

# @app.route('/bye')
# def bye():
#     return "Goodbye World!"

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

Run

```
 * Serving Flask app 'flask_test1'
 * Debug mode: off
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
```

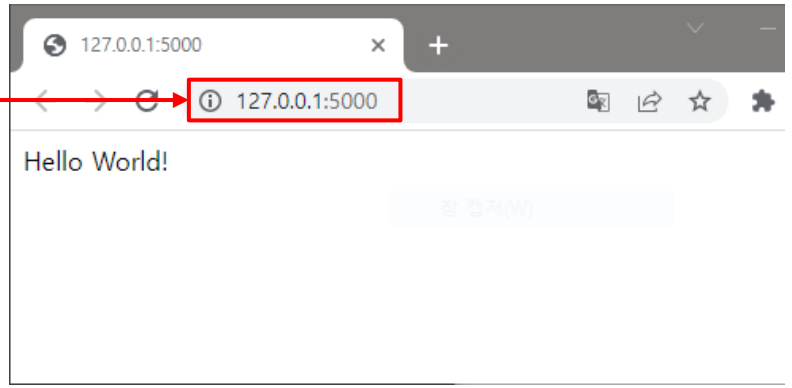
Click

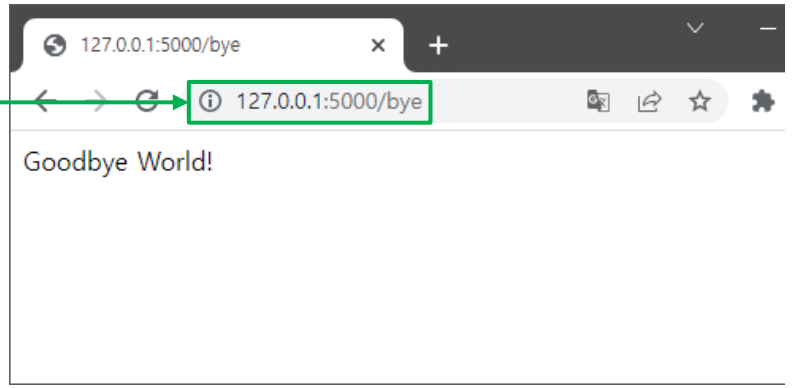


Basic Flask App

flask_test1.py

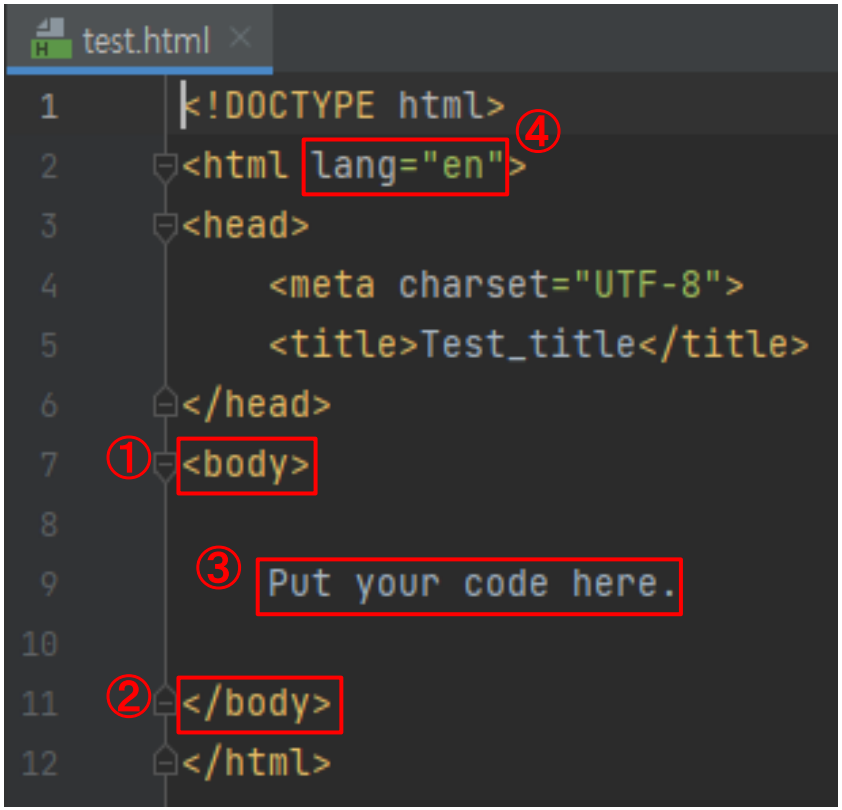
```python
from flask import Flask

app = Flask(__name__)

@app.route('/')
def main():
    return "Hello World!"

@app.route('/bye')
def bye():
    return "Goodbye World!"

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





Basic HTML Format

- In the HTML syntax, most elements are written with a start tag and an end tag, with the content in between



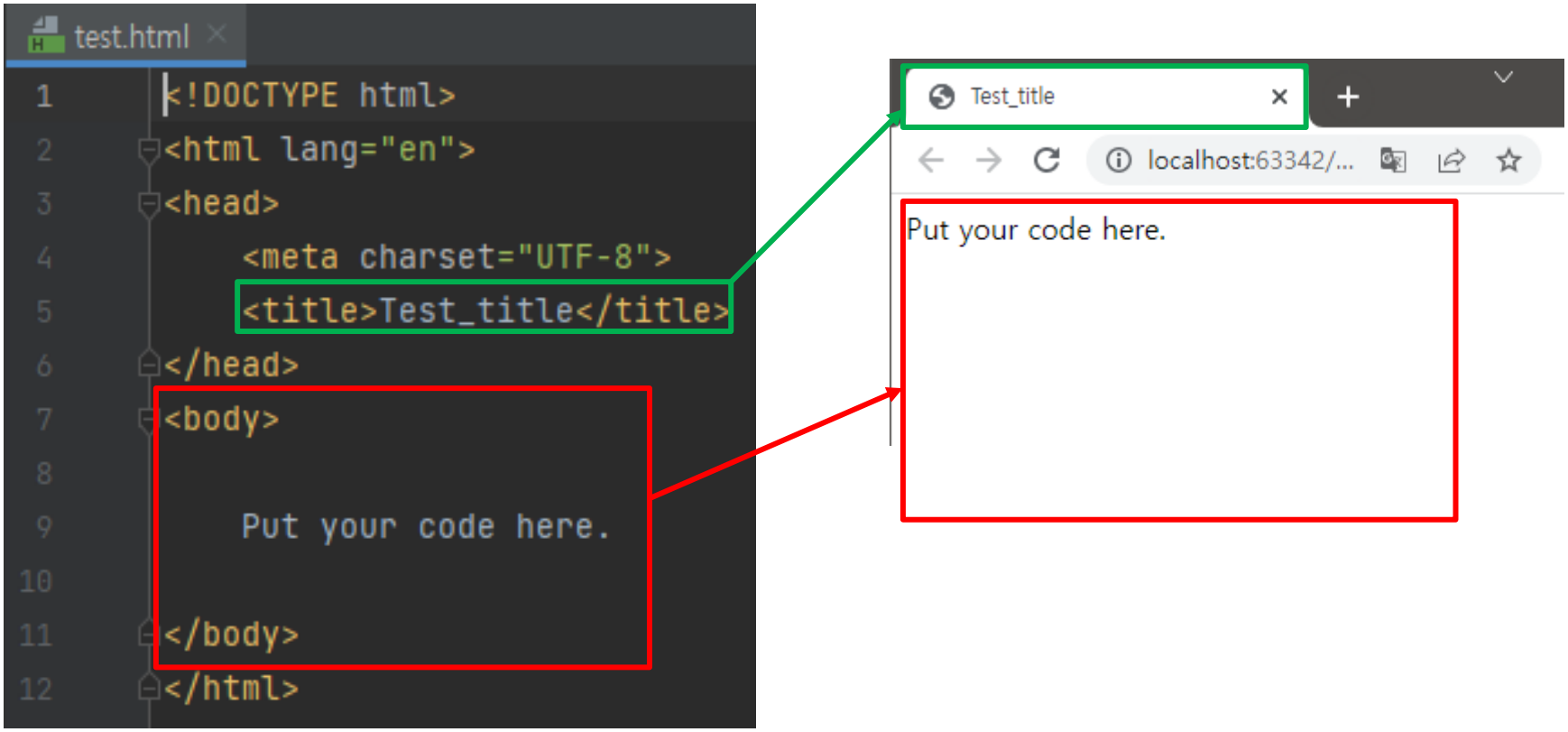
① Start tag

② End tag

③ Content

④ Attribute

Basic HTML Format



HTML Tags

-
: Insert a single line break
- <form></form>: Defines an HTML form for user input
- <label></label>: Defines a label for an <input> element
- <input>: Defines an input control

- <table></table>: Defines a table
- <thead></thead>: Groups the header content in a table
- <th></th>: Defines a header cell in a table
- <tbody></tbody>: Groups the body content in a table
- <tr></tr>: Defines a row in a table
- <td></td>: Defines a cell in a table

- More information: https://www.w3schools.com/tags/ref_byfunc.asp

<form> Tag's Attribute

Attribute	Value	Description
accept-charset	*character_set*	Specifies the character encodings that are to be used for the form submission
action	*URL*	Specifies where to send the form-data when a form is submitted
autocomplete	on off	Specifies whether a form should have autocomplete on or off
enctype	application/x-www-form-urlencoded multipart/form-data text/plain	Specifies how the form-data should be encoded when submitting it to the server (only for method="post")
method	get post	Specifies the HTTP method to use when sending form-data

https://www.w3schools.com/tags/tag_form.asp

<input> Tag's Attribute

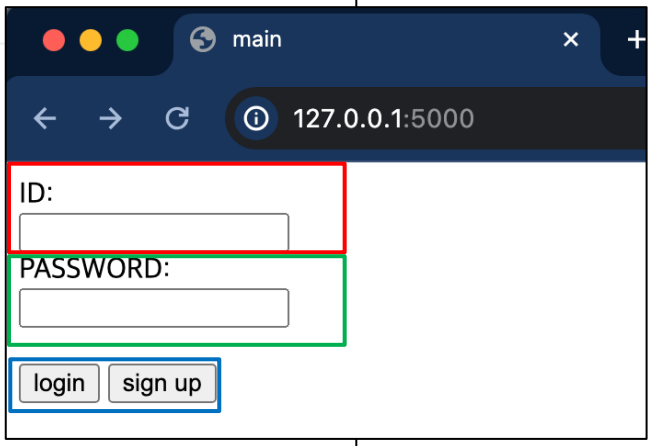
name	*text*	Specifies the name of an <input> element
pattern	*regexp*	Specifies a regular expression that an <input> element's value is checked against
type	button checkbox color date datetime-local email file hidden image month number password radio range reset search submit tel text time url week	Specifies the type <input> element to display

https://www.w3schools.com/tags/tag_input.asp

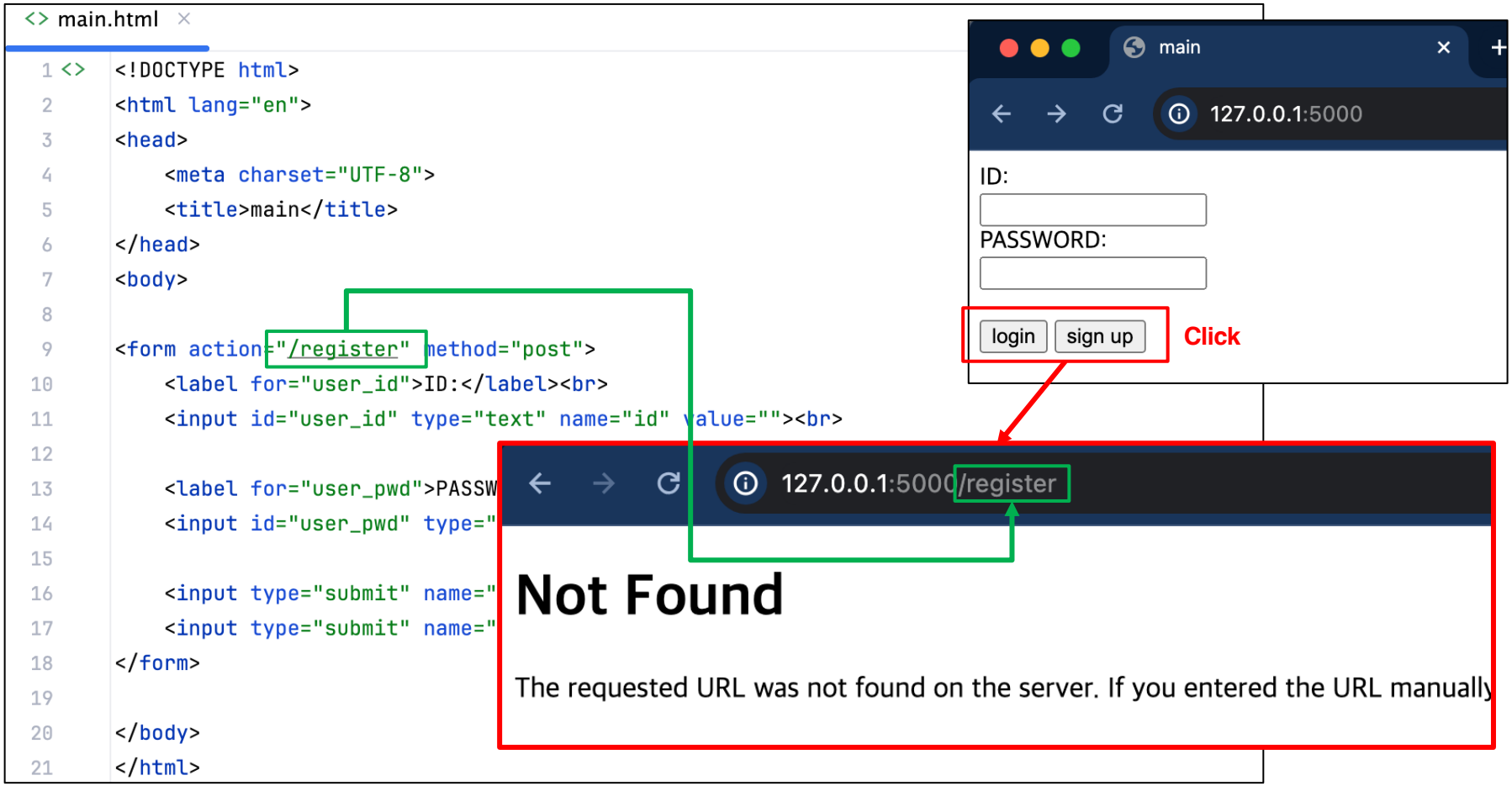
HTML Example (main.html)

main.html

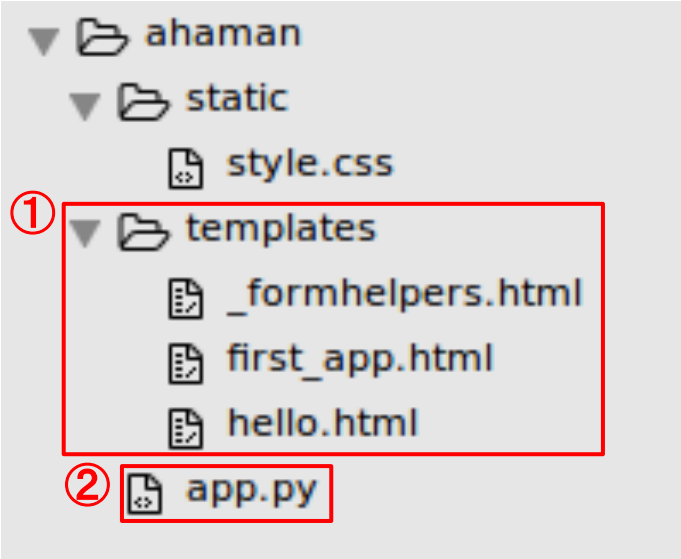
```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>main</title>
6  </head>
7  <body>
8
9  <form action="/register" method="post">
10     <label for="user_id">ID:</label><br>
11     <input id="user_id" type="text" name="id" value=""><br>
12
13     <label for="user_pwd">PASSWORD:</label><br>
14     <input id="user_pwd" type="password" name="password" value=""><br><br>
15
16     <input type="submit" name="send" value="login">
17     <input type="submit" name="send" value="sign up">
18 </form>
19
20 </body>
21 </html>
```



HTML Example (main.html)



Directory Structure



① Folder for HTML file

(The folder name must be 'templates')

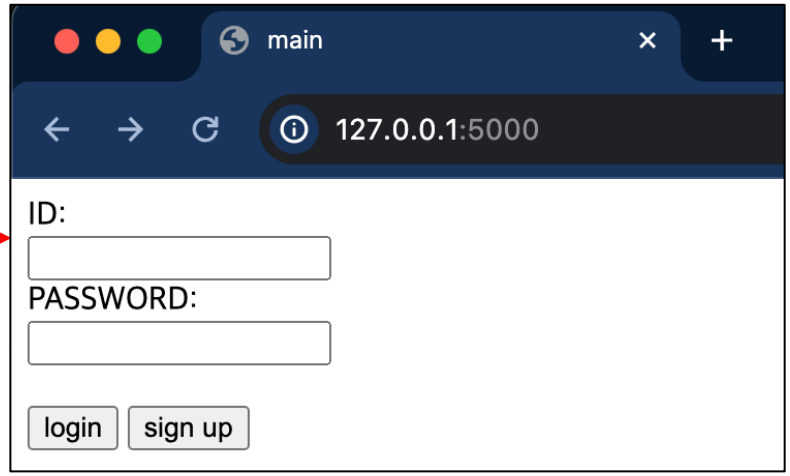
② Flask's main python file

Render HTML

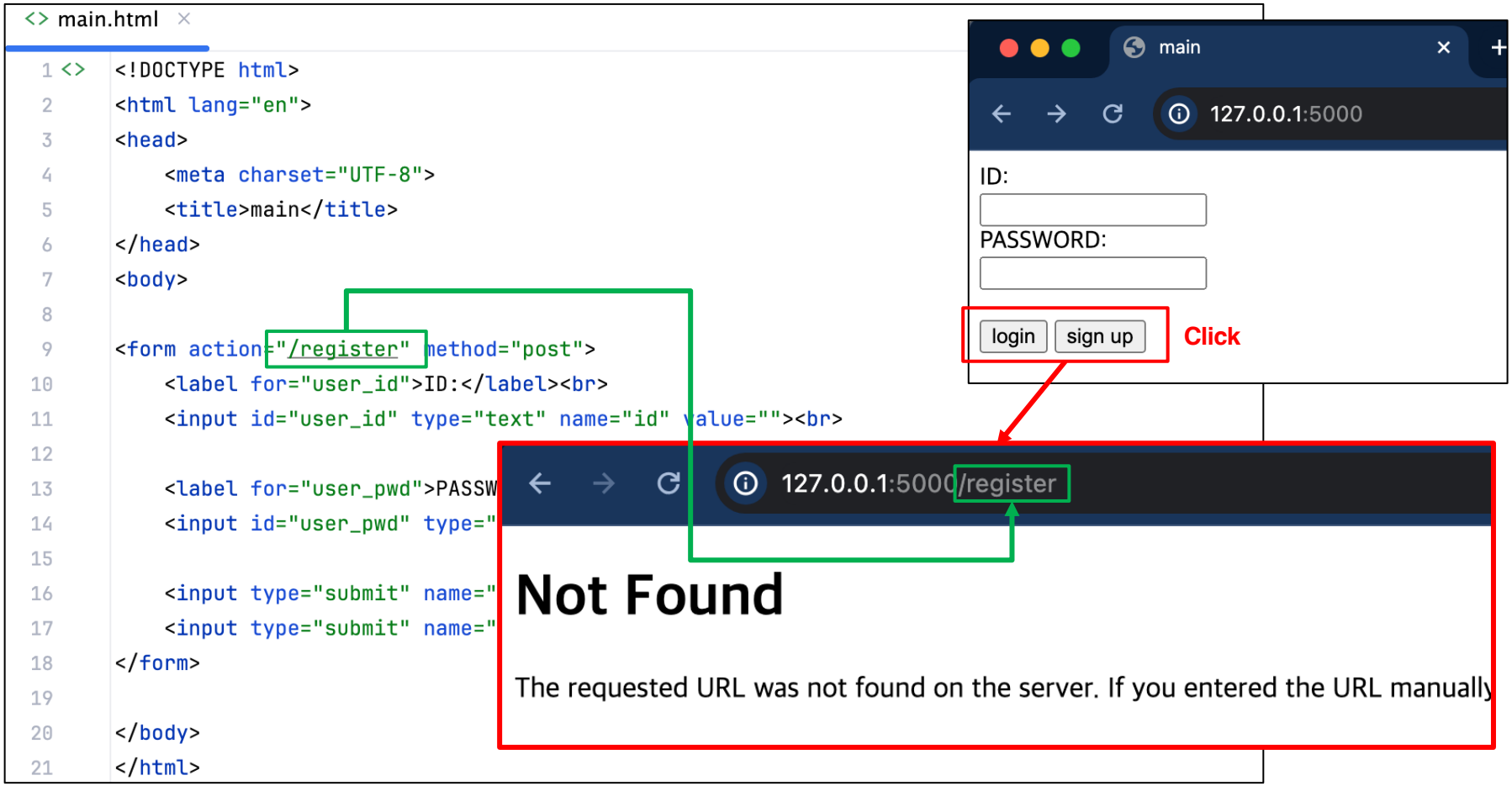
Run “app_test1.py”

app_test1.py

```
1   from flask import Flask, render_template
2
3   app = Flask(__name__)
4
5   @app.route('/')
6 <> def main():
7       return render_template("main.html")
8
```



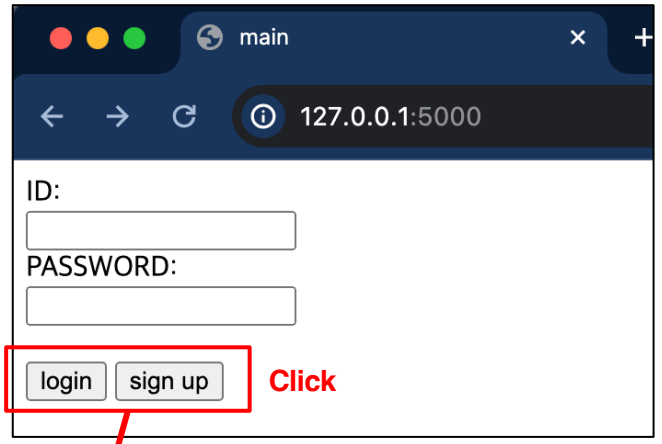
HTML Example (main.html)

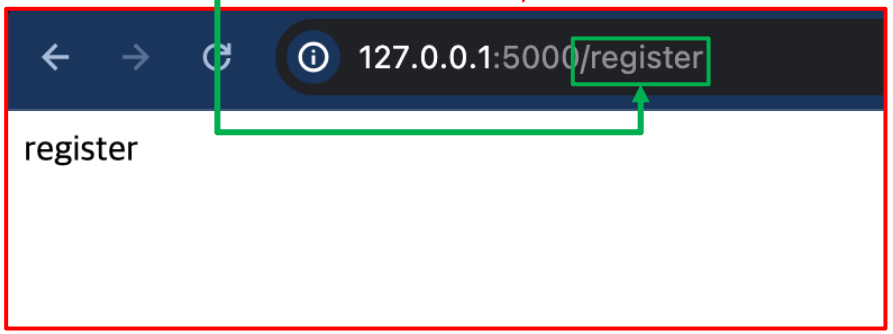


HTML Example (main.html)

app_test1.py

```python
from flask import Flask, render_template

app = Flask(__name__)

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

@app.route( rule: '/register', methods=['post'])
def register():
    return "register"
```





Receive Data from HTML

- Import request to receive data from HTML
- Open “app_test2.py”

main.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>main</title>
</head>
<body>

<form action="/register" method="post">
    <label for="user_id">ID:</label><br>
    <input id="user_id" type="text" name="id" value=""><br>

    <label for="user_pwd">PASSWORD:</label><br>
    <input id="user_pwd" type="text" name="password" value="****"><br><br>

    <input type="submit" name="send" value="login">
    <input type="submit" name="send" value="sign up">
</form>

</body>
</html>
```

app_test2.py

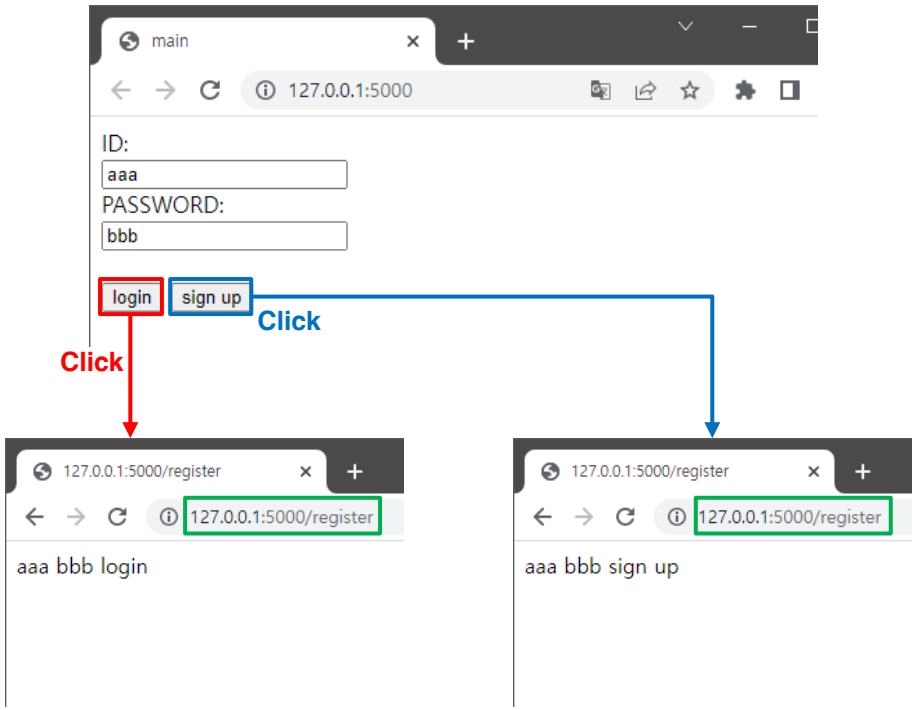
```
from flask import Flask, render_template, request

app = Flask(__name__)

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

@app.route( rule: '/register', methods=['post'])
def register():
    id = request.form["id"]
    password = request.form["password"]
    send = request.form["send"]

    return id + " " + password + " " + send

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

Receive Data from HTML

- Import request to receive data from HTML
- Run “app_test2.py”

app_test2.py

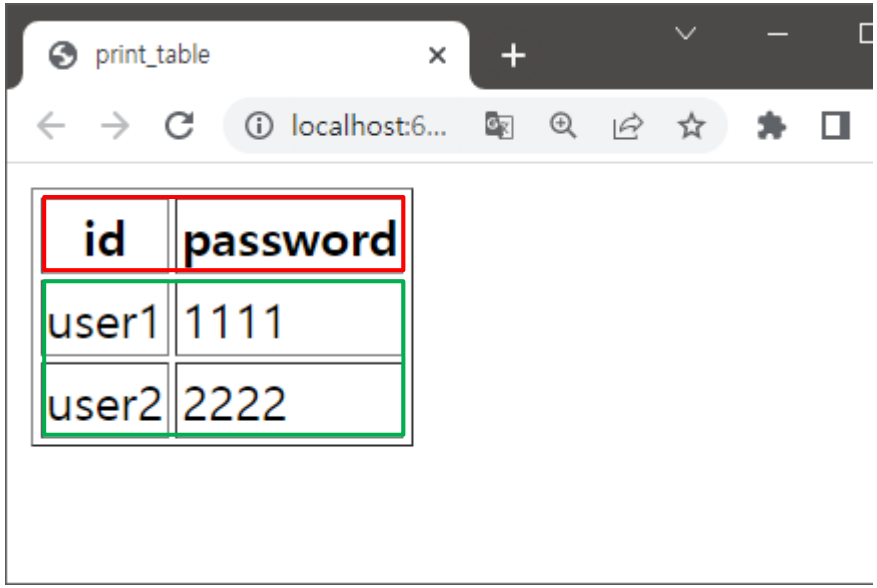
```
1   from flask import Flask, render_template, request
2
3   app = Flask(__name__)
4
5   @app.route('/')
6   def main():
7       return render_template("main.html")
8
9   @app.route( rule: '/register', methods=['post'])
10  def register():
11      id = request.form["id"]
12      password = request.form["password"]
13      send = request.form["send"]
14
15      return id + " " + password + " " + send
16
17  if __name__ == '__main__':
18      app.run()
```



HTML Example (print_table.html)

print_table.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>print_table</title>
</head>
<body>

<table border="1">
    <thead>
        <th>id</th>
        <th>password</th>
    </thead>

    <tbody>
        <tr>
            <td>user1</td>
            <td>1111</td>
        </tr>

        <tr>
            <td>user2</td>
            <td>2222</td>
        </tr>
    </tbody>
</table>

</body>
</html>
```



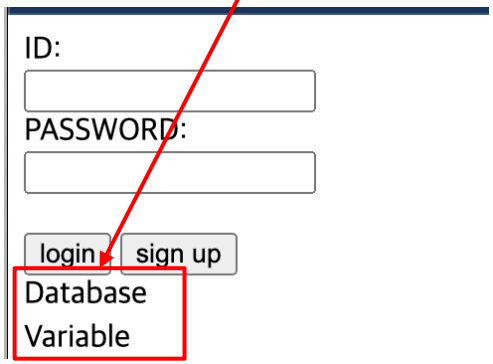
Send Data to HTML

app_test2.py

```python
from flask import Flask, render_template, request

app = Flask(__name__)

@app.route('/')
def main():
    return render_template( template_name_or_list: "main.html",
                            x="Database", y="Variable")

@app.route( rule: '/register', methods=['post'])
def register():
    id = request.form["id"]
    password = request.form["password"]
    send = request.form["send"]

    return id + " " + password + " " + send

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

main.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>main</title>
</head>
<body>

<form action="/register" method="post">
    <label for="user_id">ID:</label><br>
    <input id="user_id" type="text" name="id" value=""><br>

    <label for="user_pwd">PASSWORD:</label><br>
    <input id="user_pwd" type="password" name="password" value=""><br><br>

    <input type="submit" name="send" value="login">
    <input type="submit" name="send" value="sign up">
</form>

{{ x }} <br>
{{ y }}

{#<form action="/print_table" method="post">#}
{#    <input type="submit" name="send" value="print">#}
```



Display Query Result in HTML

- Open “app.py”

app.py × <> main.html ×

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>main</title>
</head>
<body>

<form action="/register" method="post">
    <label for="user_id">ID:</label><br>
    <input id="user_id" type="text" name="id" value=""><br>

    <label for="user_pwd">PASSWORD:</label><br>
    <input id="user_pwd" type="password" name="password" value=""><br><br>

    <input type="submit" name="send" value="login">
    <input type="submit" name="send" value="sign up">
</form>

<form action="/print_table" method="post">
    <input type="submit" name="send" value="print">
</form>

</body>
</html>
```

app.py × <> main.html

```python
import psycopg2
from flask import Flask, render_template, request

app = Flask(__name__)
connect = psycopg2.connect("dbname=tutorial user=postgres password=_____")
cur = connect.cursor()  # create cursor


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


@app.route( rule: '/return', methods=['post'])
def re_turn():
    return render_template("main.html")


@app.route( rule: '/print_table', methods=['post'])
def print_table():
    cur.execute("SELECT * FROM users;")
    result = cur.fetchall()

    return render_template( template_name_or_list: "print_table.html", users=result)


@app.route( rule: '/register', methods=['post'])
def register():
    id = request.form["id"]
    password = request.form["password"]
    send = request.form["send"]

    return id + " " + password + " " + send


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

Display Query Result in HTML

print_table.html

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>print_table</title>
6  </head>
7  <body>
8
9  <table border="1">
10     <thead>
11         <th>id</th>
12         <th>password</th>
13     </thead>
14
15     <tbody>
16         {% for user in users %}
17         <tr>
18             <td>{{ user[0] }}</td>
19             <td>{{ user[1] }}</td>
20         </tr>
21         {% endfor %}
22     </tbody>
23 </table>
24
25 </body>
26 </html>
```

```
19  @app.route( rule: '/print_table', methods=['post'])
20  def print_table():
21      cur.execute("SELECT * FROM users;")
22      result = cur.fetchall()
23
24      return render_template( template_name_or_list: "print_table.html", users=result)
```

users = [('dbstudent0', '0000'), ('database', 'postgres')]

Iteration 1 ➔ user = ('dbstudent0', '0000')

Iteration 2 ➔ user = ('database', 'postgres')

127.0.0.1:5000/print_table

id	password
dbstudent0	0000
database	postgres

Exercise

- Edit register function to make login and sign up are working (use sample_code.zip)

- For the ‘sign up’ button, add (id, password) into the “users” table
  - If there is same id in the “users” table, load “ID_collision.html”

- For the ‘login’ button,
  - If (id, password) is exist in the “users” table, load “login_success.html”
  - else, load “login_fail.html”

app.py × <> main.html

```python
import psycopg2
from flask import Flask, render_template, request

app = Flask(__name__)
connect = psycopg2.connect("dbname=tutorial user=postgres password=_____")
cur = connect.cursor()  # create cursor


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


@app.route( rule: '/return', methods=['post'])
def re_turn():
    return render_template("main.html")


@app.route( rule: '/print_table', methods=['post'])
def print_table():
    cur.execute("SELECT * FROM users;")
    result = cur.fetchall()

    return render_template( template_name_or_list: "print_table.html", users=result)


@app.route( rule: '/register', methods=['post'])
def register():
    id = request.form["id"]
    password = request.form["password"]
    send = request.form["send"]

    return id + " " + password + " " + send


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

Homework

- Complete today's exercise
- Submit your app.py on blackboard
  - 10:29:59, April 30th, 2024
  - **Only .py files** are accepted
  - **No late submission**

End of Lab