

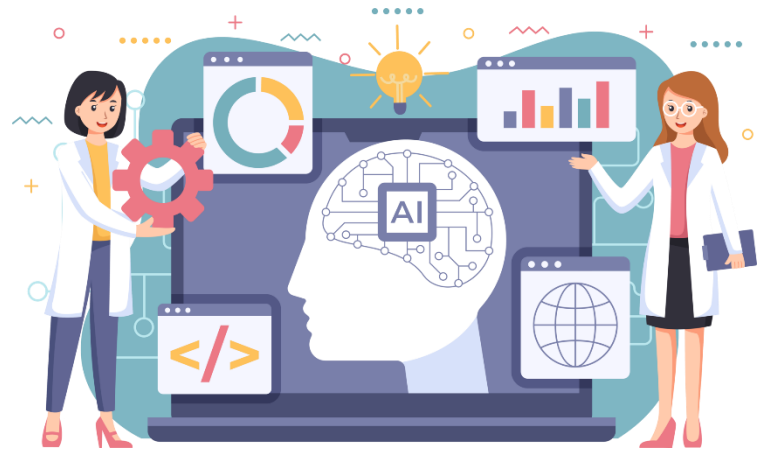


데이터 시각화 2

인공지능 RPA – 6주차

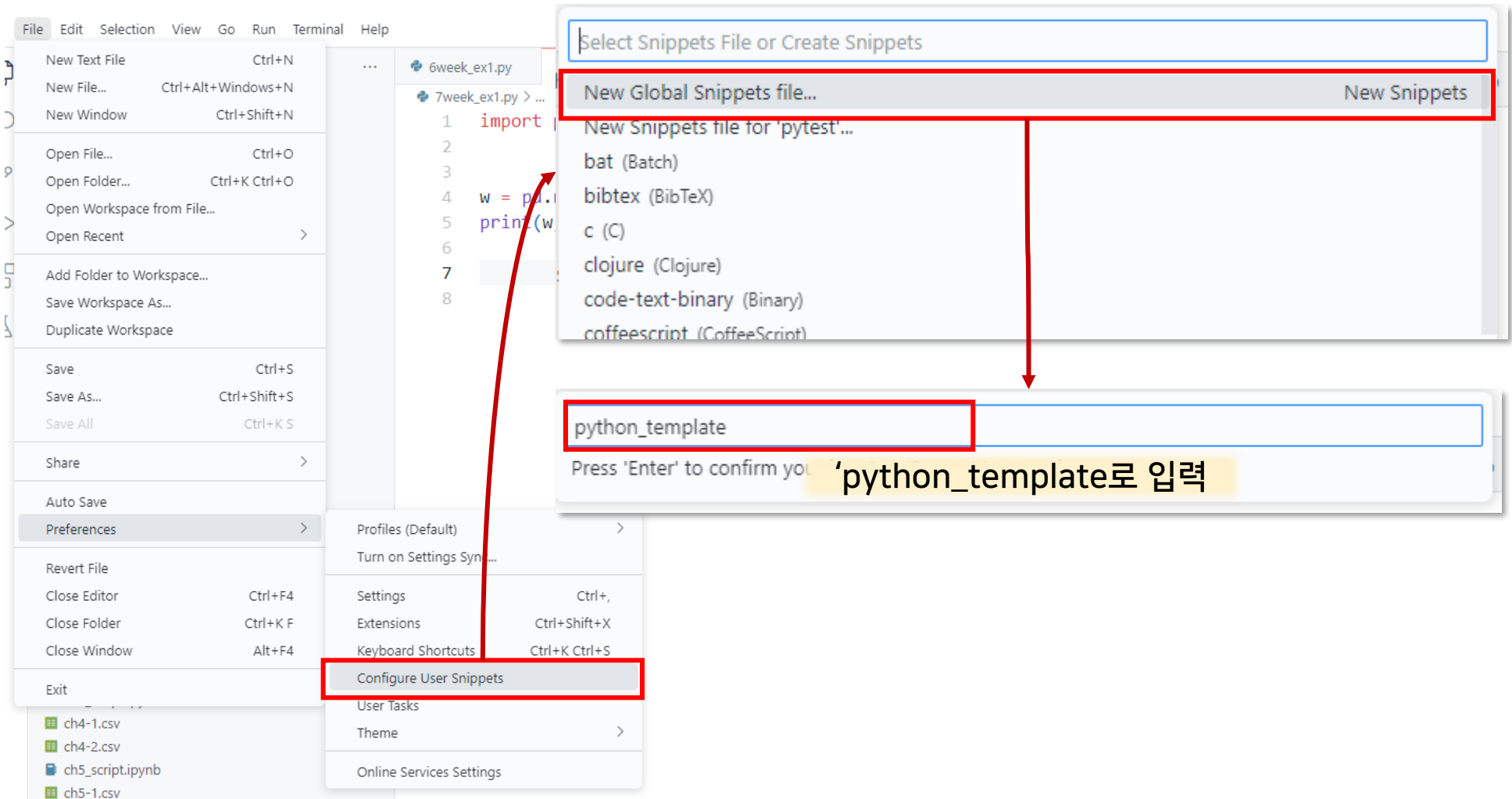
학습 내용

1. 데이터 시각화
2. FastAPI - form 다루기



인공지능 이론 및 실습

_템플릿 등록하기



템플릿 등록하기

```
python_template.code-snippets × 6week_ex1.py 7week_ex1.py ch5-1.csv
C: > Users > mail > AppData > Roaming > Code > User > snippets > python_template.code-snippets
1  {
2      // Place your global snippets here. Each snippet is d
3      // description. Add comma separated ids of the langua
4      // is left empty or omitted, the snippet gets applied
5      // used to trigger the snippet and the body will be e
6      // $1, $2 for tab stops, $0 for the final cursor posi
7      // Placeholders with the same ids are connected.
8      // Example:
9      // "Print to console": {
10     //   "scope": "javascript,typescript",
11     //   "prefix": "내용 입력",
12     //   "body": [
13     //     "console.log('$1');",
14     //     "$2"
15     //   ],
16     //   "description": "Log output to console"
17     // }
18 }
```

```
{
  "Use Pandas": {
    "scope": "python",
    "prefix": "py3",
    "body": [
      "import pandas as pd",
      "",
      "df = pd.read_csv('$1') ",
      "print(df, end='\\n\\n')",
      ""
    ],
    "description": "python start"
  }
}
```



_데이터 시각화

1. matplotlib 라이브러리 불러오기

- `pip install matplotlib`
- `import matplotlib.pyplot as plt`

2. 데이터 불러오기

- `pd.read_csv('ch4-1.csv')`

3. 차트(그래프) 함수 사용

- Bar 차트
`plt.bar()`
- 상자 차트
`plt.boxplot()`



데이터 시각화

```
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.font_manager as font_manager
```

```
def addtext(x,y):
    for i in range(len(x)):
        plt.text(i,y[i]+0.5,y[i], ha = 'center')
```

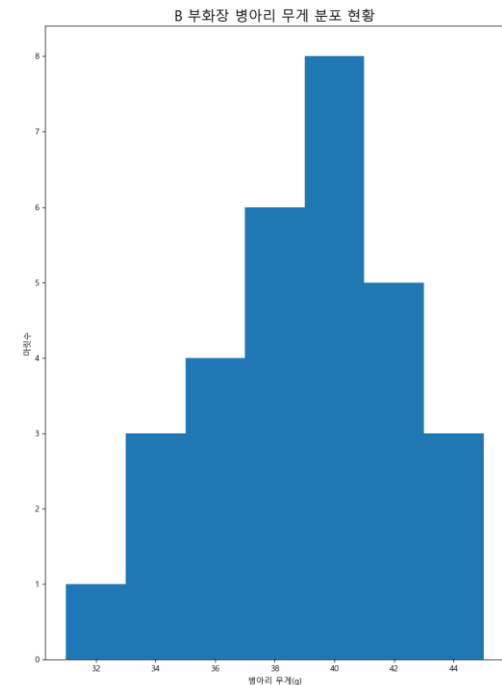
```
hat = pd.read_csv('ch4-1.csv') # hat 변수에 데이터셋 입력
print(hat, end="\n\n")
print(hat.head(), end="\n\n") # 위에서 부터 5개 데이터 확인
```

```
font_path = "malgun.ttf"
font_name = font_manager.FontProperties(fname=font_path).get_name()
plt.rc('font', family=font_name)
```



_데이터 시각화 : 히스토그램 그리기

```
# 히스토그램 그리기  
plt.figure(figsize=(10, 17)) # 그래프 크기 지정  
plt.hist(hat.weight, bins = 7)  
plt.title('B 부화장 병아리 무게 분포 현황', fontsize = 17)  
plt.xlabel('병아리 무게(g)')  
plt.ylabel('마릿수')  
  
# sns.distplot(hat.weight) # 라인 히스토그램으로 보기  
  
plt.show()
```

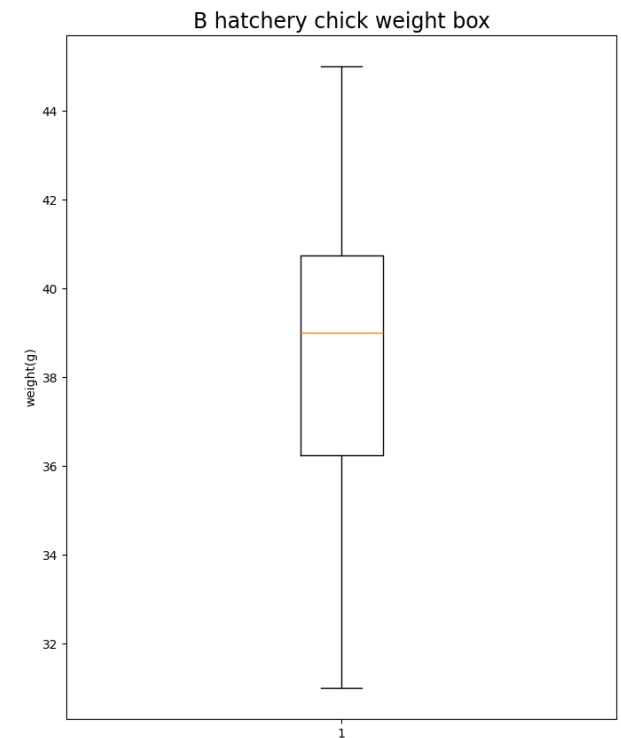


_데이터 시각화 : 상자그래프 그리기

```
import pandas as pd
import matplotlib.pyplot as plt

hat = pd.read_csv('ch4-2.csv') # hat 변수에 데이터셋 입력
print(hat.describe(), end="\n\n")

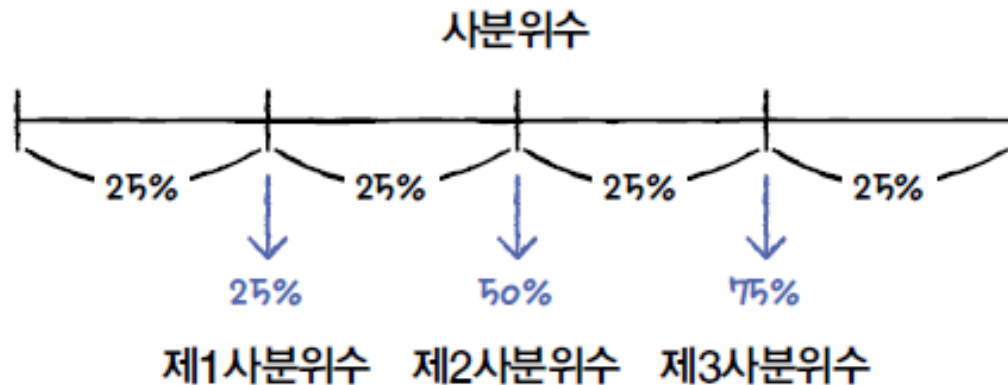
# 상자그림 그리기
plt.figure(figsize=(8, 10))
plt.boxplot(hat.weight)
plt.title('B hatchery chick weight box', fontsize =17)
plt.ylabel('weight(g)')
plt.show()
```



_데이터 시각화 : 상자그래프 그리기

분위수

- 사분위수(quartile)는 순서대로 정렬된 데이터를 네 구간으로 나눔
 - 사분위수는 3개가 나오고 각각 25%, 50%, 75%에 해당
 - 제1사분위수 - 25%에 해당하는 값
 - 제2사분위수 - 중앙값
 - 제3사분위수 - 75%에 해당하는 값



RPA 실습

_FastAPI

이름, 학번, 학과 정보를 보내면 "___학과 ___님 (___) 반갑습니다" 를 출력하는 RestAPI

- API : /user (method는 GET)
- 파라미터 : name, studentcode, major
- 출력 : { "msg" : "의료정보과 홍길동님(202223) 반갑습니다" }

<http://127.0.0.1:8000/info?major=의료정보과&name=홍길동&studentcode=202223>

The screenshot shows a REST client interface with a GET request to `http://127.0.0.1:8000/info?major=의료정보과&name=홍길동&studentcode=202223`. The request parameters are listed in a table below.

Key	Value	Description
<input type="checkbox"/> name	홍길동	Description
<input type="checkbox"/> major	의료정보과	Description
<input type="checkbox"/> studentcode	202223	Description
<input type="checkbox"/> Key	Value	Description

The response is shown in the 'Response' section, indicating a status of 200 OK. The response body is a JSON object:

```
{  
  "msg": "의료정보과 홍길동님 (202223) 반갑습니다."  
}
```



FastAPI

이름, 학번, 학과 정보를 보내면 "___학과 ___님 (___) 반갑습니다" 를 출력하는 RestAPI

- API : /user (method는 GET)
- 파라미터 : name, studentcode, major
- 출력 : { "msg" : "의료정보과 홍길동님(202223) 반갑습니다" }

@app.get("/user")

```
async def read_user(name: str = None, studentcode: str = None, major: str = None):  
    return {"msg": f"{major} {name}님 ({studentcode}) 반갑습니다."}
```



_FastAPI

VS Code Extension에서 확인 KeyRunner - API Client 설치



KeyRunner - API Client v1.0.56
KeyRunner [keyrunner.app](#) | 5,274 | ★★★★★ (3)
The Zero Trust API Client
[Disable](#) [Uninstall](#) ⚙️

1 New

GET ex1 GET Untitled Req

4week / ex1

GET http://127.0.0.1:8000

2 URL 입력

Params 0 Authorization Headers 5 Body Scripts Tests Code Snippet Bulk Edit

Key	Value	Description
Key	Value	Description

2-1 파라미터 입력

Response Time: 38 ms Content length: 17 Status: 200 OK

Body Headers 4 Cookies 0 Decoded JWT Test Results

```
{ 1 Items  
  Hello: "World"  
}
```

3-1 결과 확인

4 저장

3 호출 - 실행



_FastAPI

1. Fast API 단독 실행 코드

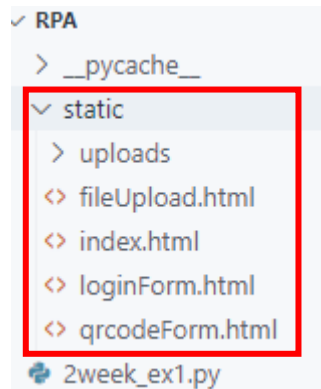
```
if __name__ == '__main__':  
    import uvicorn  
    uvicorn.run(app, host='127.0.0.1', port=8000, log_level="info")
```

2. HTML 파일 서비스 설정

```
from fastapi.staticfiles import StaticFiles  
app.mount("/", StaticFiles(directory="static", html=True), name="static")
```

static 폴더를 만들고 폴더안에 html 파일 만들기

```
<!DOCTYPE html>  
<html lang="en">  
<body>  
<div> Hello World </div>  
</body>  
</html>
```

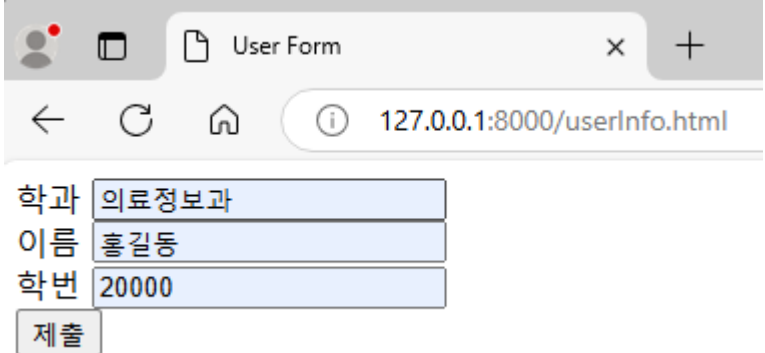


_FastAPI

3. User Form 만들기

- Form 엘리먼트 추가

```
<form action="/user" method="post">  
  <span>학과</span>  
  <input type="text" name="major"><br>  
  <span>이름</span>  
  <input type="text" name="name"><br>  
  <span>학번</span>  
  <input type="text" name="studentcode" ><br>  
  <input type="submit" name="submit" value="제출">  
</form>
```



A screenshot of a web browser window. The title bar shows 'User Form'. The address bar shows '127.0.0.1:8000/userInfo.html'. The form contains three text input fields labeled '학과' (Major), '이름' (Name), and '학번' (Student ID), with values '의료정보과', '홍길동', and '20000' respectively. Below the inputs is a '제출' (Submit) button.

학과	의료정보과
이름	홍길동
학번	20000
<input type="submit" value="제출"/>	



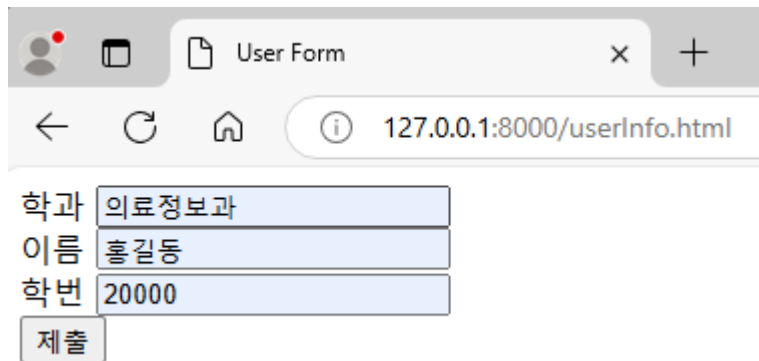
_FastAPI

4. Form 요청에 대응하는 API 만들기

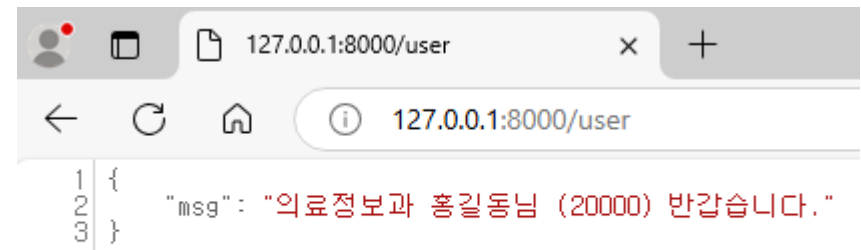
```
from fastapi import Form
```

```
@app.post("/user")
```

```
async def read_user_form(name: str = Form(...), studentcode: str = Form(...), major: str = Form(...)):  
    return {"msg": f"{major} {name}님 ({studentcode}) 반갑습니다."}
```



학과	의료정보과
이름	홍길동
학번	20000
제출	



```
1 {  
2   "msg": "의료정보과 홍길동님 (20000) 반갑습니다."  
3 }
```

