한 걸음 더

추가로 공부해보면 좋을 것들

▼ Flask - url에서 데이터를 받기

👉 필요한 경우, url의 일부 값을 변수로 가져올 수도 있습니다.

즉, 아래와 같이 입력한 상태에서,
http://localhost:5000/profile/sparta

이라고 브라우저에 치면,
Hi! sparta

라고 출력하게 됩니다.

```python
@app.route("/profile/<username>") def get_profile(username): return "Hi! " + username
```

Python

▼ Flask - render_template으로 데이터를 주기

👉 render_template으로 html 파일을 건네줄 때, 데이터를 미리 줄 수 도 있습니다.
이 방법을 쓰면, 화면 로딩 후 ajax 요청을 하지 않아도 됩니다. (물론, 해도 됩니다.^^;)

아래와 같은 app.py, index.html을 만들어두고, 실습해보세요!
http://localhost:5000/sparta 라고 치면,

Hi! 내 이름은 sparta입니다. 라고 나오게 될 것입니다.

```python
from flask import Flask, render_template app = Flask(__name__)
@app.route('/<myname>') def home(myname): return
render_template('index.html',name=myname) if __name__ == '__main__':
app.run('0.0.0.0',port=5000,debug=True)
```

Python

```html
<html> <head> <title>스파르타코딩클럽의 페이지</title> <!-- head 안에 들어 갈 어떤 요소들을 여기에 넣습니다 --> </head> <body> Hi! <h1>내 이름은 {{ name }}입니다.</h1> </body> </html>
```

HTML

웹 개발할때 사용하기 좋은 무료 API가 정리된 깃헙(클릭해서 이동)>>

참고해보면 좋은 사이트

- Codepen: 다양한 프론트엔드 예시를 살펴볼 수 있습니다! 검색창에 원하는 대상을 입력해보세요
  - 참고: https://gocoder.tistory.com/130

파이썬으로 할 수 있는 다양한 것들

"또 무얼 할 수 있죠?" 하는 분들을 위해 준비한 재밌는 것들!
여기선 키워드만 알려드릴게요. 구글링을 통해 공부해보세요!

▼ Selenium - 브라우저 제어

👉 웹스크래핑을 해야하는데, 네이버 로그인을 꼭 해야한다면?

👉 혼자 공부해보기: (링크1), ... 스스로 찾기!

▼ 상세설명+코드 예시

- 패키지 설치하기

```
selenium
```

Python

- chrome driver 설치하기

  크롬브라우저를 조작하려면, 제어해줄 수 있는 파일이 필요합니다. 아래 링크에서 드라이버를 다운 받습니다.

  다운로드:
  https://sites.google.com/a/chromium.org/chromedriver/downloads

- 1) 페이스북 창 열기

```
from selenium import webdriver from
selenium.webdriver.common.keys import Keys path =
"C:/Users/bumky/Desktop/develop/chromedriver" driver =
webdriver.Chrome(path) driver.get("http://www.facebook.org")
```

Python

- 2) 페이스북에 로그인해보기

```
from selenium import webdriver from
selenium.webdriver.common.keys import Keys usr = '본인아이디'
pwd = '본인비밀번호' path =
"C:/Users/bumky/Desktop/develop/chromedriver" driver =
webdriver.Chrome(path) driver.get("http://www.facebook.org")
assert "Facebook" in driver.title elem =
driver.find_element_by_id("email") elem.send_keys(usr) elem =
driver.find_element_by_id("pass") elem.send_keys(pwd)
elem.send_keys(Keys.RETURN)
```

Python

- 3) 로그인하고, esc 눌러보기

```
from selenium import webdriver from
selenium.webdriver.common.keys import Keys usr = '본인아이디'
pwd = '본인비밀번호' path =
"C:/Users/bumky/Desktop/develop/chromedriver" driver =
webdriver.Chrome(path) driver.get("http://www.facebook.org")
assert "Facebook" in driver.title elem =
driver.find_element_by_id("email") elem.send_keys(usr) elem =
driver.find_element_by_id("pass") elem.send_keys(pwd)
elem.send_keys(Keys.RETURN) driver.implicitly_wait(5)
driver.find_element_by_tag_name('body').send_keys(Keys.ESCAPE)
driver.implicitly_wait(2)
```

Python

- 4) esc 후에 포스팅 목록 불러오기

```
html = driver.page_source soup = BeautifulSoup(html,
'html.parser') posts = soup.select('div.text_exposed_root') for
post in posts: print (post.text)
```

Python

- 5) 10번 내리고, 포스팅 목록 불러오기

```
for i in range(10):
driver.find_element_by_tag_name('body').send_keys(Keys.END)
driver.implicitly_wait(10) time.sleep(5)
```

Python

- 6) 완성된 코드

```
from selenium import webdriver from
selenium.webdriver.common.keys import Keys import time from bs4
import BeautifulSoup usr = '본인아이디' pwd = '본인비밀번호' path
= "C:/Users/bumky/Desktop/develop/chromedriver" driver =
webdriver.Chrome(path) driver.get("http://www.facebook.org")
assert "Facebook" in driver.title elem =
driver.find_element_by_id("email") elem.send_keys(usr) elem =
driver.find_element_by_id("pass") elem.send_keys(pwd)
elem.send_keys(Keys.RETURN) driver.implicitly_wait(5)
driver.find_element_by_tag_name('body').send_keys(Keys.ESCAPE)
driver.implicitly_wait(5)
driver.find_element_by_tag_name('body').send_keys(Keys.ESCAPE)
for i in range(5):
driver.find_element_by_tag_name('body').send_keys(Keys.END)
driver.implicitly_wait(3) time.sleep(5) html =
driver.page_source soup = BeautifulSoup(html, 'html.parser')
```

```
posts = soup.select('div.text_exposed_root') for post in posts:
print(post.text)
```

Python

▼ 셀레니움과 Beautifulsoup의 연결!

👉 1) 셀레니움으로 브라우저를 열고,
2) HTML을 받아온 다음
3) Beautifulsoup으로 우리가 원하는 정보면 가지고 오면 가장 좋겠죠?

아래 코드를 참고해보세요. 빨간 음영 부분이 핵심입니다!

```
from selenium import webdriver driver = webdriver.Chrome('C:\\Users\\bu
from bs4 import BeautifulSoup keyword = '코끼리' url = 'https://www.goo
q='+keyword+'&hl=ko&sxsrf=ACYBGNSTW5YFeVU0I4abA6H_bXsmwJ-
gag:1582014089814&source=lnms&tbm=isch&sa=X&ved=2ahUKEwj7kune1drnAhXaA
driver.get(url) req = driver.page_source soup = BeautifulSoup(req, 'htt
div.islrc > div') for count, image in enumerate(images): img = image.s
if count == 5: break driver.close()
```

Python

▼ Telegram Bot - 텔레그램 봇

👉 텔레그램 봇을 쉽게 만들 수 있어요! (정말 쉬워요. 돈 워리!)

👉 혼자 공부해보기: (링크1), (링크2), ... 스스로 찾기!

▼ 코드 예시

```
from telegram.ext import Updater, MessageHandler, Filters # import
modules my_token = '950898370:AAEa9ZtiDT9beSOGWb9QhHplqGtkgAOKEWo'
print('start telegram chat bot') # message reply function def
get_message(bot, update): if '안녕' in update.message.text:
update.message.reply_text('그래') elif '뭐해' in
update.message.text: update.message.reply_text('그냥 있지') else:
update.message.reply_text('뭐라는거야') updater = Updater(my_token)
message_handler = MessageHandler(Filters.text, get_message)
updater.dispatcher.add_handler(message_handler)
updater.start_polling(timeout=3, clean=True) updater.idle()
```

Python

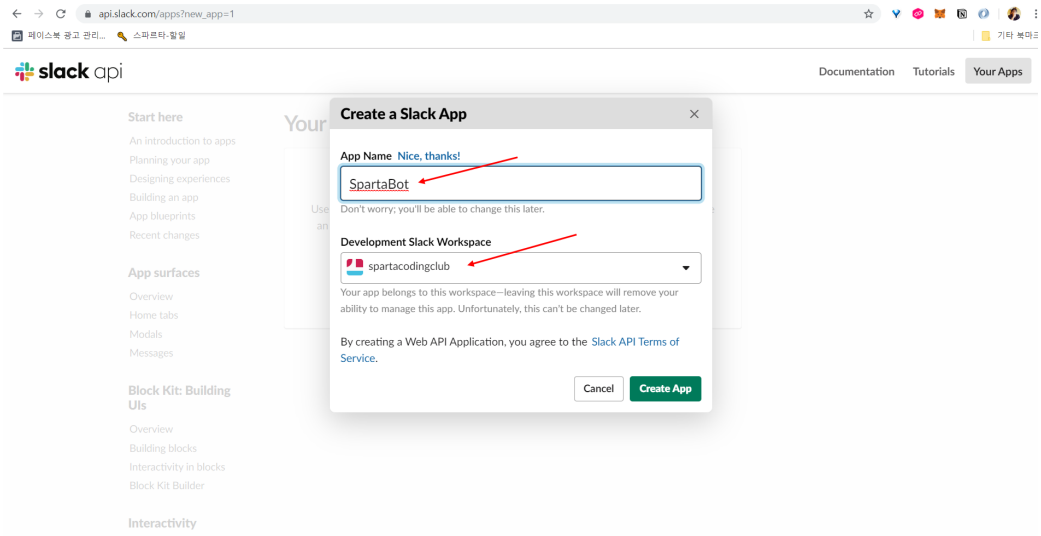
▼ Slack Bot - 슬랙 봇

👉 Telegram은 익숙하지 않다고요? 그럼 슬랙은 어떠신지요!

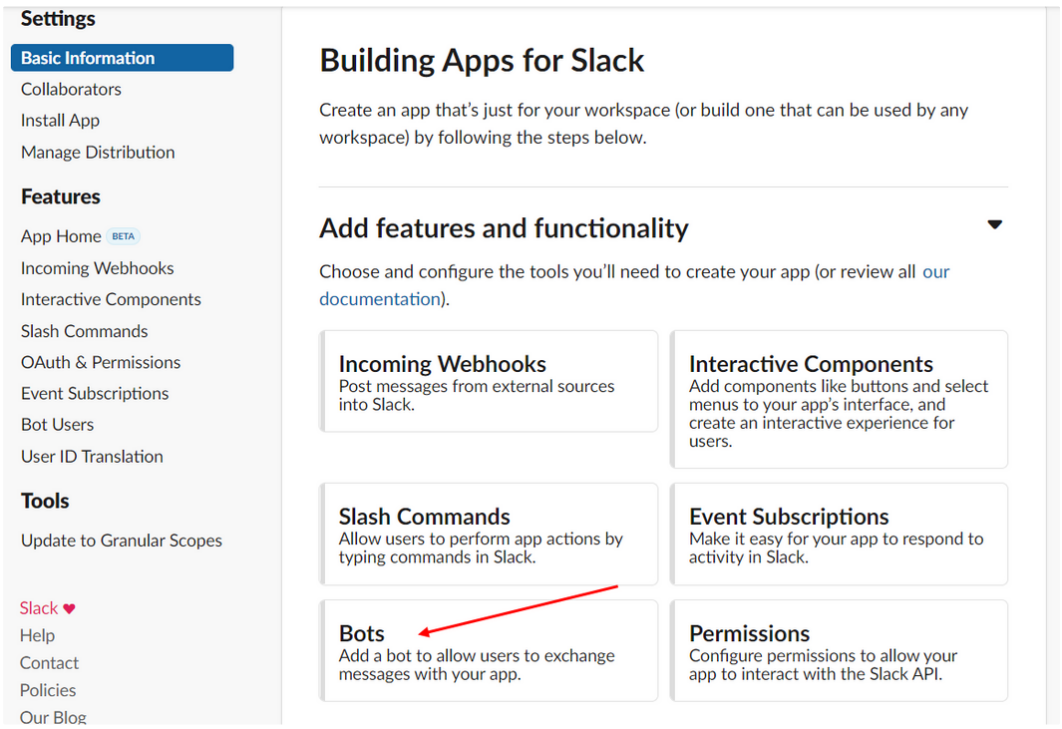
👉 혼자 공부해보기: (링크1), ... 스스로 찾기!

▼ 1) 토큰 발급받기

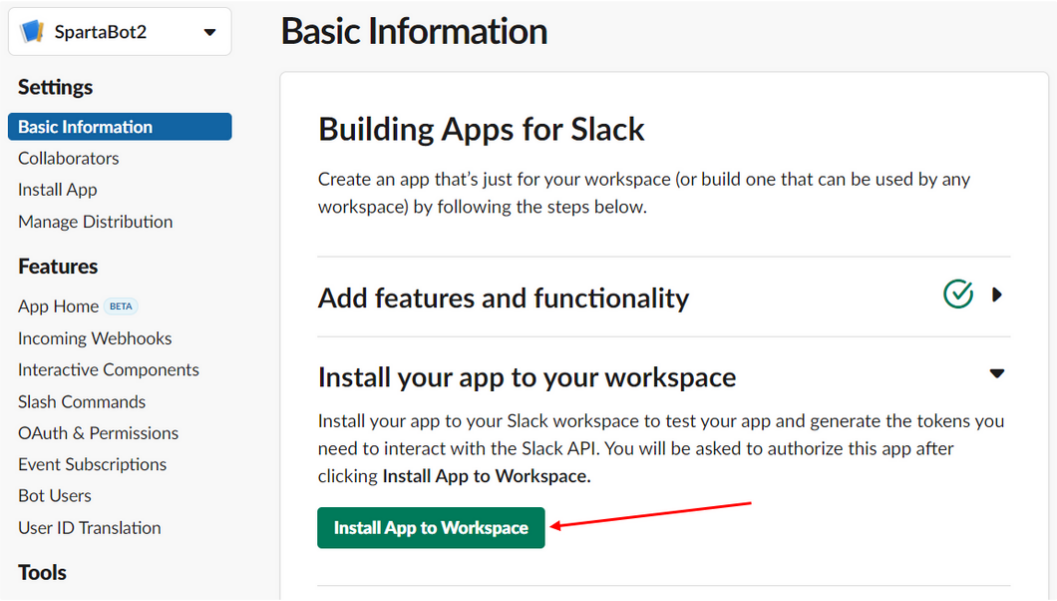
1. Slack App 페이지에 접속해서, Create an App를 클릭하고 봇 이름과 공유할 그룹을 입력



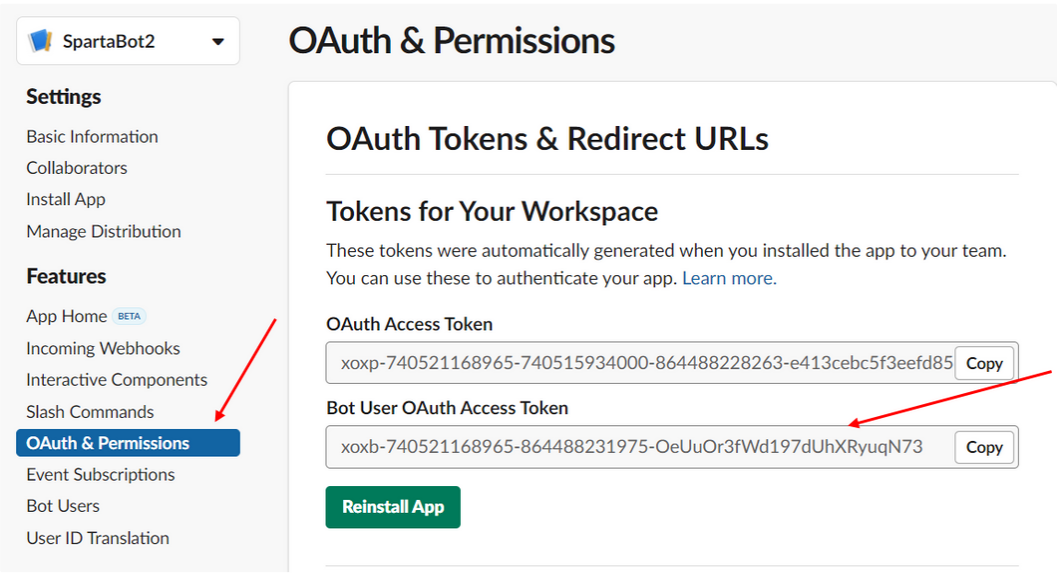
2. Basic Information에서, Bots를 클릭해 기능을 줍니다.



3. 다시 Basic Information에서, workspace에 앱(=봇)을 추가합니다.

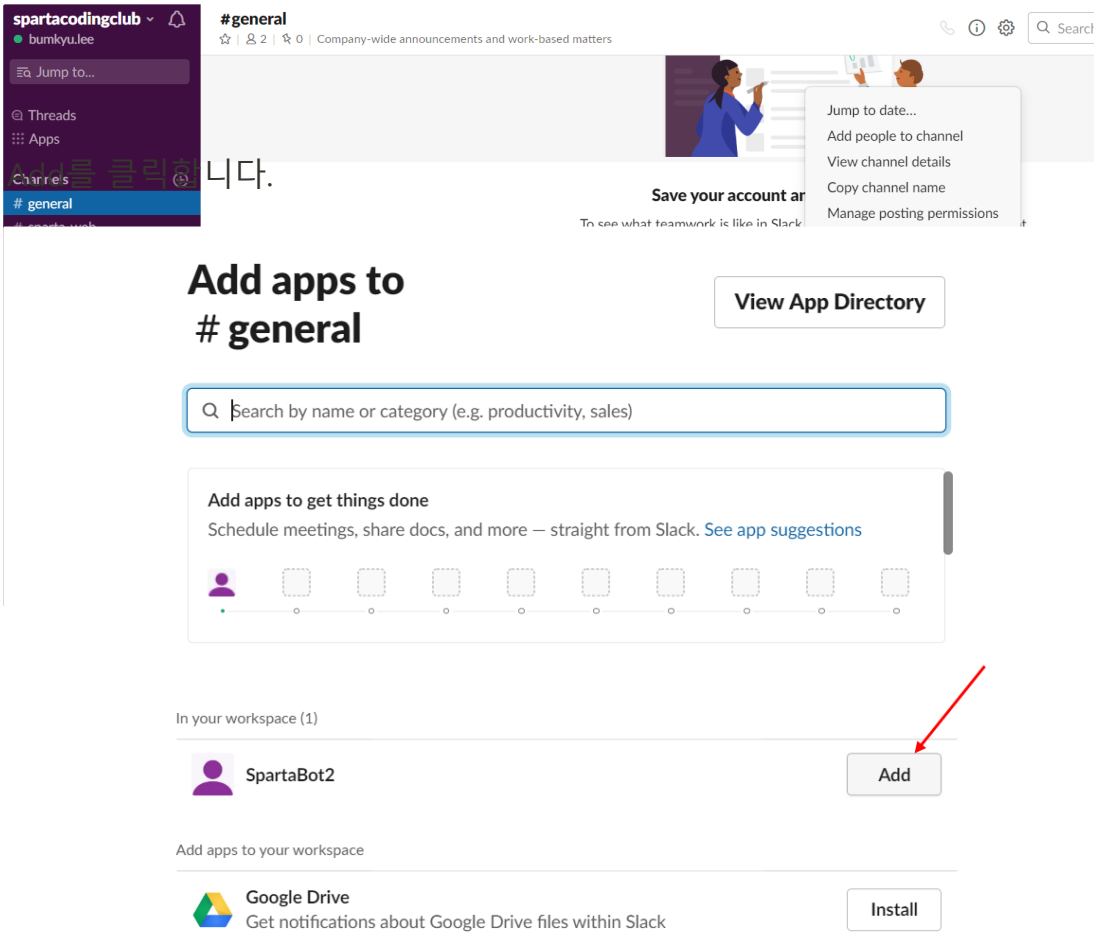


4. OAuth & Permissions에서 토큰을 복사해둡니다.



▼ 2) 봇을 팀 채널에 초대하기

슬랙 웹 또는 앱에서 Add an app을 클릭합니다.



Add를 클릭합니다.

▼ 3) 봇이 메시지 보내게 하기

💡 인스톨 할 패키지: slackclient

```
import slack slack_token = 'xoxb-740521168965-864488231975-
OeUuOr3fWd197dUhXRyuqN73' client =
slack.WebClient(token=slack_token)
client.chat_postMessage(channel="#general", text="Hello world!")
```

Python

▼ 4) 코드 예시 - 묻는 말에 대답하기

👉 print (data) 로, 어떤 정보들을 불러올 수 있는지 확인해보세요
chat_postMessage 위에, 특정 기능을 수행하도록 코딩을 해둘 수 도 있습니다.
(예를 들어 증권사API를 통해 주식을 사고 팔 수게 할 수도 있겠죠?)

👉 slackclient 패키지를 인스톨하세요!

```
import slack @slack.RTMClient.run_on(event='message') def
say_hello(**payload): data = payload['data'] # 받은메시지의 모든 정보
if 'bot_id' in data: # 봇이 보낸 메시지 일때는 패스 return channel_id
= data['channel'] # 받은메시지의 채널 정보(어느 채널에서 왔는가?)
web_client = payload['web_client'] # 받은메시지에 응답할 때 필요한 정
보 chat = data.get('text', []) # 받은메시지의 내용 if 'hello' in
chat: # 만약 chat에 hello라는 단어가 포함돼있으면
web_client.chat_postMessage( # 답을 한다 channel=channel_id,
text='외국인이세요?' ) elif '안녕' in chat: # 만약 chat에 안녕이라는
단어가 포함돼있으면 web_client.chat_postMessage( # 답을 한다
channel=channel_id, text='한국인이세요?' ) else: # 이도저도 아니면
web_client.chat_postMessage( # 답을 한다 channel=channel_id,
text='무슨 말인지 모르겠어요!' ) slack_token = 'xoxb-740521168965-
864488231975-OeUuOr3fWd197dUhXRyuqN73' rtm_client =
slack.RTMClient(token=slack_token) rtm_client.start()
```

JavaScript

▼ pyautogui - 키보드, 마우스 제어

👉 파이썬으로 키보드, 마우스를 제어할 수 있어요! → 매크로를 만들 수 있다는 사실!

👉 혼자 공부해보기: (링크1), ... 스스로 찾기!

▼ gspread - 구글스프레드시트 연결

👉 구글스프레드 시트 연결도 물론 가능하죠! 읽기, 쓰기!

👉 혼자 공부해보기: (링크1), ... 스스로 찾기!

▼ gmail 보내기 - 메일 보내기!

👉 안되는 게 없습니다! gmail, naver 메일 등 쉽게 보낼 수 있어요

👉 혼자 공부해보기: (링크1), ... 스스로 찾기!

1. 파이썬으로 메일도 보내나요?!

   네, 할 수 있습니다! 이처럼 누구나 많이 쓰는 동작들은 대부분 라이브러리(미리 짜여진 작은 프로그램)로 작성되어 있으니, 앞으로 여러분이 필요한 것이 있다면 구글에 검색해보세요! 거의 대부분 존재할 거에요 😎

2. 기본 코드

```
import smtplib from email.mime.multipart import MIMEMultipart from email.mime.text import MIMEText me = "내 이메일 주소를 입력하세요." my_password = "내 이메일 비밀번호를 입력하세요." you = "상대방 이메일을 입력하세요" ## 여기서부터 코드를 작성하세요. ## 여기에서 코드 작성이 끝납니다. # Gmail 관련 필요한 정보를 획득합니다. s = smtplib.SMTP_SSL('smtp.gmail.com') # Gmail에 로그인합니다. s.login(me, my_password) # 메일을 전송합니다. s.sendmail(me, you, msg.as_string()) # 프로그램을 종료합니다. s.quit()
```

Python

3. 완성 코드

```
import smtplib from email.mime.multipart import MIMEMultipart from email.mime.text import MIMEText me = "bumkyu.lee@gmail.com" my_password = "vcpdxrnjxegajhdn" you = "bk.lee@spartacodingclub.kr" ## 여기서부터 코드를 작성하세요. msg = MIMEMultipart('alternative') msg['Subject'] = "Alert" msg['From'] = me msg['To'] = you html = '<html><body><p>Hi, I have the following alerts for you!</p></body></html>' part2 = MIMEText(html, 'html') msg.attach(part2) ## 여기에서 코드 작성이 끝납니다. # Send the message via gmail's regular server, over SSL - passwords are being sent, afterall s = smtplib.SMTP_SSL('smtp.gmail.com') s.login(me, my_password) s.sendmail(me, you, msg.as_string()) s.quit()
```

Python

▼ [모르고 넘어가도 무방하지만, 궁금하면 살펴보세요! - SMTP, MINE란?]

```
import smtplib from email.mime.multipart import MIMEMultipart from
```

```python
email.mime.text import MIMEText
```

Python

❓ 위에서 쓰인 smtplib, mime 이런 것은 무엇인가요? 🤔

👉 이메일 전송 규약과 그에 관한 확장 프로토콜(약속)인 SMTP, MIME에 관한 라이브러리입니다. 이게 무슨 소리인가하면요!

HTTP에 대해 들어보셨나요? HTTP가 우리가 일반적으로 아는 '웹'을 구성하는 프로토콜, 그러니까 하나의 약속입니다. HTML, CSS, Javascript가 무엇이고 그것을 어떻게 주고받을 것인지, 인증 처리는 어떻게 할 것인지에 대해 광범위한 규약이라고 할 수 있지요.

하지만 우리가 아는 '인터넷'은 오로지 HTTP로 구성되어있지 않습니다.
→ FTP(File Transfer Protocol): 파일을 송수신하기 위한 프로토콜
→ SMTP(Simple Mail Transfer Protocol): 글자로만 구성된 이메일을 주고받는 프로토콜
→ MIME(Multi-purpose Internet Mail Extensions): 이메일이 다른 종류의 데이터도 포함할 수 있도록 도와주는 프로토콜

정리하자면, 이메일과 관련해서는!
글자로'만' 구성된 이메일을 주고받기 위해 반드시 지켜야 하는 규칙인 SMTP와 더불어, 이메일에 글자 뿐만 아니라 이미지, 동영상 등을 주고받을 수 있도록 도와주는 규칙인 MIME이 존재합니다. 이 규칙을 준수하기 위해 코드 상단에 위 세 줄이 포함된 것이구요.

더 자세한 내용은 다음 링크를 살펴보세요.
(링크 1), (링크 2), ...

▼ PyJWT - 로그인 구현하기

👉 편의를 위해, 1~4주차에서 배운 수준으로 로그인을 구현해두었습니다.
아래 깃 주소를 clone 받아 코드를 살펴봐주세요!

주석으로 상세 설명을 적어두었습니다.
https://github.com/bumkyulee/login_prac

👉 회원가입은 id / pw 를 받아 DB에 저장하고,
로그인은 id / pw 를 받아 DB에 있는 것과 맞춰보는 과정이지만,

로그인하고 요청하는 API들(예 - 회원정보 확인, 내 글 목록 확인 등)에서는
어떻게 이 API가 로그인된 회원에게서 온 것이라고 확인할 수 있을까요?

로그인은 크게 아래와 같은 방법으로 구현된답니다.
토큰을 만드는 방법은 여러가지가 있는데, JWT(링크)를 이용하면 편리하답니다.

스파르타의 예시 코드를 살펴봐주세요!

1. **회원가입** : id / pw 를 받아서 DB에 저장한다. 단, pw는 암호화해서 저장한다.
2. **로그인** : id / pw를 받아서, pw를 암호화 시킨 다음에 DB에 있는 정보와 비교한다. 성공하면 '몇시간짜리 토큰'을 준다.
3. **로그인 이후**: 모든 API를 요청할 때, '몇시간짜리 토큰'을 같이 준다. 서버에서는 동작을 수행하기 전에, 해당 토큰이 유효한 것인지를 판단하고, API의 본래 역할을 수행한다.

👉 혼자 공부해보기: (링크1), (링크2), (링크3) ... 스스로 찾기!

▼ Schedule - 주기적으로 실행하기

👉 주기적으로 함수를 실행해야 한다면? (매일 아침 10시 or 10초마다 등) schedule 라이브러리를 활용해보세요!

👉 혼자 공부해보기: (링크1), ... 스스로 찾기!

▼ 코드

```python
import schedule def job(): print('여기에 할 일을 넣기') def run(): schedule.every().day.at('09:00').do(job) # 매일 09:00 마다 job 함수를 실행 while True: schedule.run_pending() if __name__ == "__main__": run()
```

Python

... and more! 상상할 수 있는 모든 것 모두!