AI_05_오승원_Section3

Property	
상태	

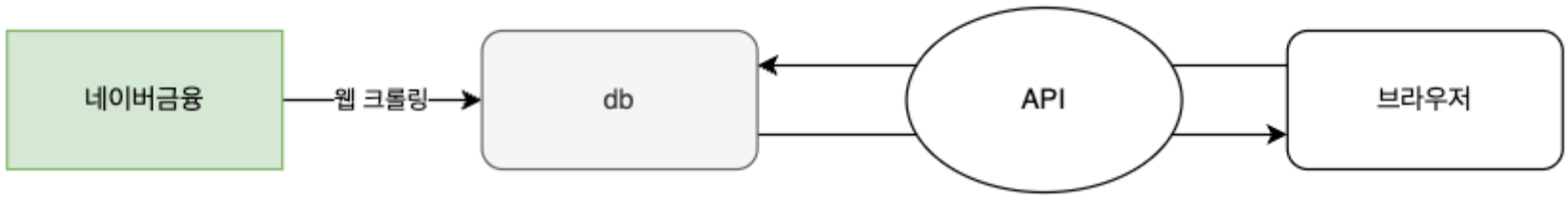
I. 서비스 소개

1. 서비스명 : 주식관리 웹

1) 서비스 설명

- 해당 웹은 현재 국내 주식 시장의 일별 시세를 가져와 내가 원하는 정보를 얻고, 내 자산의 현황도 알 수 있는 서비스로 만들고자 했음.

2) 파이프라인



- 네이버금융의 일별 시세를 웹 크롤링 방식으로 가져와 내 컴퓨터 내 db에 데이터 입력하고 기획한 API를 통해 브라우저를 통해 간단히 주식 정보를 알아보고자 함.

3) 서비스기획

- 상장한 기업들의 주가 정보를 가져와 쉽게 내가 원하는 정보(상장회사 코드, 종가,시가,저가,고가, 거래량)을 db에 저장하여 python을 통해 내가 원하는 방식으로 데이터 분석이 가능하도록 하고자 함.
- 내가 보유한 주식을 넣으면 자동으로 총금액을 계산해주고 얼마나 떨어졌는지 올랐는지 알려주는 서비스를 기획함.

II.서비스 시현

1.웹 크롤링

- 네이버금융의 일별 시세를 웹 크롤링 방식으로 가져옴.

일별시세

날짜	종가	전일비	시가	고가	저가	거래량
2022.01.25	319,000	▼ 9,500	329,500	331,500	318,500	567,171
2022.01.24	328,500	▼ 4,500	333,000	334,000	327,500	494,977
2022.01.21	333,000	▼ 2,000	330,500	340,500	330,000	580,928
2022.01.20	335,000	▲ 2,500	331,000	336,500	326,500	695,553
2022.01.19	332,500	▼ 5,500	330,000	334,000	328,000	749,338
2022.01.18	338,000	▼ 5,500	346,500	348,000	335,500	441,573
2022.01.17	343,500	▲ 1,500	340,500	346,500	339,500	418,789
2022.01.14	342,000	▼ 6,000	341,000	342,500	335,500	504,907
2022.01.13	348,000	▲ 2,500	348,000	352,000	344,000	677,226
2022.01.12	345,500	▲ 10,500	339,500	352,500	339,500	1,047,713

- beatufulsoup4과 pymysql등을 활용하여 db에 테이블을 만들고 주식 정보를 입력하는 방법을 진행함(약4년 정보 업데이트에 약 5시간 정도 걸림)

```python
def read_naver(self, code, company, pages_to_fetch):
    """네이버에서 주식 시세를 읽어서 데이터프레임으로 반환"""
    try:
        url = f"http://finance.naver.com/item/sise_day.nhn?code={code}"
        html = BeautifulSoup(requests.get(url,
                                          headers={'User-agent': "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/
                                          537.36 (KHTML, like Gecko) Chrome/97.0.4692.71 Safari/537.36"}).text, "lxml")
        pgrr = html.find("td", class_="pgRR")
        if pgrr is None:
            return None
        s = str(pgrr.a["href"]).split('=')
        lastpage = s[-1]
        df = pd.DataFrame()
        pages = min(int(lastpage), pages_to_fetch)
        for page in range(1, pages + 1):
            pg_url = '{}&page={}'.format(url, page)
            df = df.append(pd.read_html(requests.get(pg_url,
                                                     headers={'User-agent': "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
                                                     AppleWebKit/537.36 (KHTML, like Gecko) Chrome/97.0.4692.71 Safari/537.
                                                     36"}).text)[0])
            tmnow = datetime.now().strftime('%Y-%m-%d %H:%M')
            print('[{}] {} ({}) : {:04d}/{:04d} pages are downloading...'.
                  format(tmnow, company, code, page, pages), end="\r")
        df = df.rename(columns={'날짜': 'date', '종가': 'close', '전일비': 'diff'
            , '시가': 'open', '고가': 'high', '저가': 'low', '거래량': 'volume'})
        df['date'] = df['date'].replace('.', '-')
        df = df.dropna()
        df[['close', 'diff', 'open', 'high', 'low', 'volume']] = df[['close',
                                                                     'diff', 'open', 'high', 'low',
                                                                     'volume']].astype(int)
        df = df[['date', 'open', 'high', 'low', 'close', 'diff', 'volume']]
    except Exception as e:
        print('Exception occured :', str(e))
        return None
    return df
```

- db에 입력된 내용

investar.daily_price: 1,569,401 행 (층) (대략적), 제한 수: 1,000 다음

code	date	open	high	low	close	diff	volume
000020	2018-01-03	9,900	10,250	9,820	10,000	130	268,220
000020	2018-01-04	10,050	10,050	9,680	9,750	250	161,342
000020	2018-01-05	9,750	9,980	9,750	9,910	160	116,604
000020	2018-01-08	10,000	10,150	9,940	9,950	40	158,326
000020	2018-01-09	9,950	10,050	9,690	9,770	180	184,742
000020	2018-01-10	9,800	9,910	9,770	9,840	70	89,778
000020	2018-01-11	9,840	10,350	9,840	10,300	460	317,974
000020	2018-01-12	10,350	10,450	10,100	10,250	50	311,381
000020	2018-01-15	10,300	10,450	10,100	10,350	100	269,179
000020	2018-01-16	10,350	10,350	10,000	10,050	300	216,119
000020	2018-01-17	10,100	10,100	9,830	9,900	150	224,108
000020	2018-01-18	9,990	10,250	9,910	10,100	200	234,830
000020	2018-01-19	10,150	10,850	10,150	10,550	450	496,024
000020	2018-01-22	10,450	10,800	10,350	10,500	50	241,650
000020	2018-01-23	10,500	10,700	10,350	10,500	0	157,733
000020	2018-01-24	10,650	10,650	10,400	10,500	0	187,411
000020	2018-01-25	10,500	10,600	10,350	10,500	0	179,397
000020	2018-01-26	10,450	10,900	10,400	10,750	250	251,422
000020	2018-01-29	10,700	10,750	10,500	10,550	200	245,952
000020	2018-01-30	10,500	11,000	10,300	10,800	250	526,654
000020	2018-01-31	10,850	11,300	10,650	10,800	0	485,972
000020	2018-02-01	10,850	10,900	10,450	10,450	350	298,372
000020	2018-02-02	10,450	10,500	10,000	10,300	150	294,961
000020	2018-02-05	9,900	10,650	9,900	10,050	250	313,533
000020	2018-02-06	9,620	10,000	9,530	10,000	50	305,434
000020	2018-02-07	10,150	10,750	10,000	10,050	50	588,716
000020	2018-02-08	10,150	10,300	10,000	10,250	200	180,650
000020	2018-02-09	9,850	10,200	9,850	10,050	200	166,414
000020	2018-02-12	10,300	11,050	10,300	10,900	850	673,670
000020	2018-02-13	11,100	11,200	10,400	10,450	450	343,493
000020	2018-02-14	10,450	10,600	10,150	10,500	50	197,459
000020	2018-02-19	10,550	10,800	10,500	10,650	150	197,794

investar.company_info: 2,486 행 (층) (대략적), 제한 수: 1,00

code	company	last_update
000890	보해양조	2022-01-23
000910	유니온	2022-01-23
000950	전방	2022-01-23
000970	한국주철관공업	2022-01-23
000990	DB하이텍	2022-01-23
001000	신라섬유	2022-01-23
001020	페이퍼코리아	2022-01-23
001040	CJ	2022-01-23
001060	JW중외제약	2022-01-23
001070	대한방직	2022-01-23
001080	만호제강	2022-01-23
001120	LX인터내셔널	2022-01-23
001130	대한제분	2022-01-23
001140	국보	2022-01-23
001200	유진증권	2022-01-23
001210	금호전기	2022-01-23
001230	동국제강	2022-01-23
001250	GS글로벌	2022-01-23
001260	남광토건	2022-01-23
001270	부국증권	2022-01-23
001290	상상인증권	2022-01-23
001340	백광산업	2022-01-23
001360	삼성제약	2022-01-23
001380	SG글로벌	2022-01-23
001390	KG케미칼	2022-01-23
001420	태원물산	2022-01-23
001430	세아베스틸	2022-01-23
001440	대한전선	2022-01-23
001450	현대해상	2022-01-23
001460	BYC	2022-01-23
001470	삼부토건	2022-01-23
001500	현대차증권	2022-01-23

2.API 서비스 구현

- 주가 데이터를 가져오는 애플리케이션 파이썬 파일을 완성 후 이를 API로 만드려고 시도했으나 쉽지 않았음.
- 단순히 주식 코드와 수를 넣으면 잔고가 나오게 하는 화면까지만 진행함.

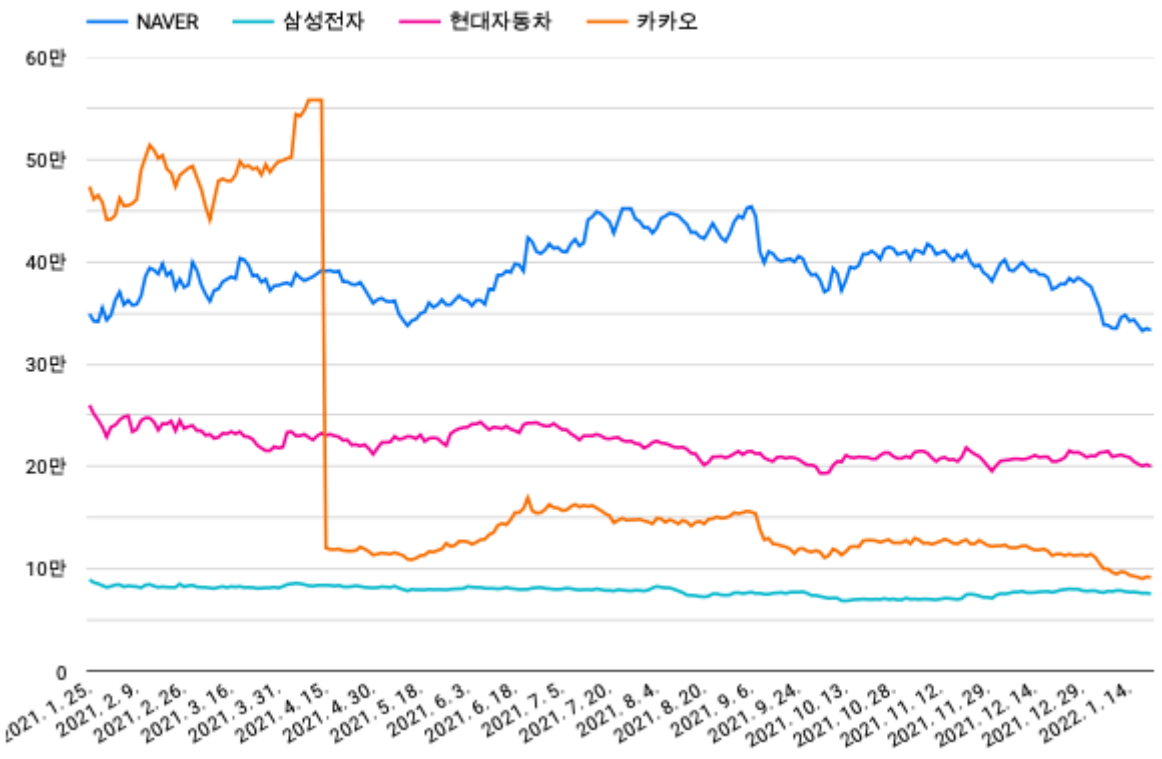
localhost:8000/balance/?035420=20&005930=10&005380=12&035720=15

앱 Bookmarks 파이썬으로 배우는... 주식 글쓰기 TomatoTimer 즐겨찾기

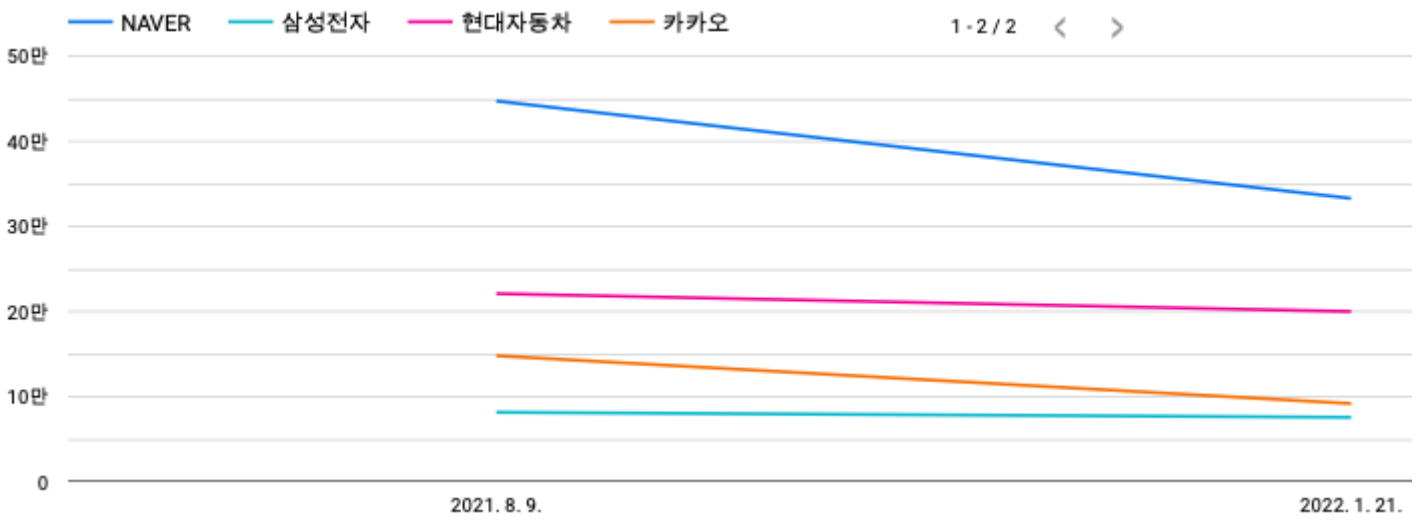
종목명	종목코드	현재가	주식수	등락률	평가금액
NAVER	035420	321,000	20	-2.28%	6,420,000
삼성전자	005930	73,800	10	-1.73%	738,000
현대차	005380	191,500	12	-2.79%	2,298,000
카카오	035720	88,300	15	-1.89%	1,324,500
계좌 잔고			10,780,500		

3.대시보드 구현

- 구글 데이터스튜디오를 활용함
- 1) 가상의 개인 포트폴리오를 정해 2021.01.25~ 2022.01.21까지의 주가의 상승과 하락을 알아보고자함. (네이버20주, 삼성전자10주, 현대자동차12주, 카카오 15주)
  - *카카오는 주식분할이 이루어져 55만8000원(지난 9일 종가)에서 11만1600원으로 내려감



- 매수시점을 21년 8월 9일로 잡고 현재까지의 주가 변동을 살펴봄
  - 전체적으로 하락하는 모습을 보여줌.



- 구매 당시 개인 보유 주식 총액은 14, 637,000 원이었나 현재 기준 11,193,000 원으로 떨어짐.

- 네이버의 보유 비중이 높아 하락폭이 더 크게 나타남.

III.마무리발언

- flask구현의 어려움
  - 데이터를 모으고 이를 db에 저장하고 서비스 구현을 위한 코딩까지는 큰 무리없이 진행했으나 flask를 통한 서비스 구현에는 어려움을 느낌.
  - 좀 더 시간을 두고 차근차근 공부해야 할 것으로 보임
- 데이터스튜디오의 파워풀한 시각화 능력
  - flask구현으로 시간이 없어 바쁘게 진행했음에도 만들고자 하는 것들을 쉽게 만들 수 있었음. 시각화 툴의 필요성에 대해서 많이 느낄 수 있었음.
- 해당 프로젝트는 '파이썬 증권 데이터 분석'을 많이 참고했습니다.