

안녕하세요~
단단한 개발자가 되고픈
신입개발자 박현수 입니다.

Phone : 010 - 2170 - 1306

Email : sweetsound95@gmail.com



- 인적사항

이름 : 박현수(PARK HYUN SOO)

생년월일 : 1995.01.20

Phone : 010-2170-1306

Email : sweetsound95@gmail.com

- 학적

용인대학교 경영정보학과(2015 ~ 2021)

- 사용가능 Skill

JAVA, Javascript, HTML, CSS, SQL

- 한마디

단단한 개발자를 넘어, 단단한 사나이가 되고픈 신입개발자

DANVESTING PROJECT

개요

머릿말

코로나19 바이러스 이슈가 나왔을때,
여러 개발자들이 코로나19 확진자 수 통계프로그램, 코로나19 확진자 동선 알림 프로그램
코로나19 잔여백신 알림 프로그램 등등을 만드는 것을 보면서,
사회 이슈와 관련있는 프로그램을 만들고 싶었습니다.

코로나19 이슈와 함께, 부동산 폭등과 주식투자 열풍이 불면서
“영끌 빚투 ” 라는 용어가 생겨날 정도로 사회적 이슈가 되는 것을 보고,
investing.com을 모티브로 한, 주식 시세 확인 사이트를 만들기로 결정했습니다.

주식 시세 데이터를 사용하려면, 필연적으로 외부REST API를 사용해야 했습니다.
우연의 일치로 그 당시 미국의 주식 매매 서비스 Robinhood를 사용하는 투자자들이
기관투자자들의 게임스탑(GameStop)공매도에 대응해, 주식을 대량 매수해서 화제가 되었습니다.

Robinhood라는 서비스를 검색하던 중에, 해당 서비스가 미국주식 정보 조회 API를 무료로 제공한다는 것을 알게되었고
Robinhood API를 사용해서 주식 시세 확인 사이트를 구현했습니다.

프로젝트 정보

- Github 주소

<https://github.com/HSNURcat>

- 구현에 사용된 것들

Spring framework, Spring boot, Mybatis, JSTL, Bootstrap
jQuery, JAVA, Javascript, HTML, CSS, Robinhood API
Jsoup, Gson, Google Chart API

- 라이선스



Github 접속 QR-Code

VIEW 기획

- Oven을 사용해서 view를 기획했습니다.

<https://ovenapp.io/view/UmMs7pVmuHIFhAWODSfPs42pEi0o5teg/>



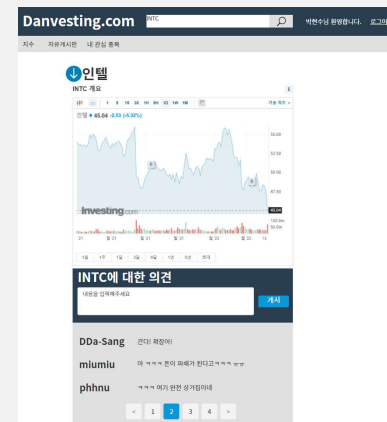
로그인 화면

The login screen for Danvesting.com. It features a dark blue header with the site name. Below it, a light gray box contains two input fields for 'ID' and 'PW'. At the bottom of the box, there is a link for '사용자 등록' (User Registration) and a dark blue '로그인' (Login) button.

main 화면



종목 정보 화면



DB 기획

- Google spread sheet를 사용해서 DB설계를 진행했습니다.

이 설계를 통해 컬럼 타입, null허용여부, 각 컬럼에 대한 설명을 입력한 다음
이 설계를 참고하여 테이블을 생성했습니다.

접속주소

https://docs.google.com/spreadsheets/d/1TZ5owEAYrdZrynSVB5nclCb-YuPhLwtDK49aidd3Y_0/edit?usp=sharing

테이블 이름 : post				
설명 : 일반사용자가 자유게시판에 작성한 게시물을 저장하는 테이블				
컬럼명	타입	NULL가능 여부	auto_increament	설명
id	int	No	Yes	Primary Key
writerId	int	N	N	게시물 작성 사용자 (user 테이블 PK)
nickName	varchar(32)	N	N	게시물 작성자 닉네임(user 테이블 닉네임)
title	varchar(256)	N	N	게시물 제목
content	text	N	N	게시물 내용
imagePath	varchar(256)	Y	N	첨부 이미지, 접근 가능한 이미지 경로
createdAt	timestamp	N	N	생성날짜
updatedAt	timestamp	N	N	수정날짜



Google spread sheet 접속
QR-Code

URL 기획

- Google spread sheet를 사용해서 URL설계를 진행했습니다.
이 설계를 통해 View페이지의 URL과 각 기능의 API URL을 지정하고,
어떤 값을 파라미터로 넘길 것인지, 어떤 형태의 리턴값을 json형식으로 받을 것인지
지정했습니다.

접속주소

https://docs.google.com/spreadsheets/d/1TZ5owEAYrdZrynSVB5nclCb-YuPhLwtDK49aidd3Y_0/edit?usp=sharing



Google spread sheet 접속
QR-Code

View URL 설계			
제목	url	parameter	설명
로그인	/user/sign_in_view		로그인 화면
회원가입	/user/sign_up_view		회원가입 화면
회원정보 수정 전 회원확인	/user/member/check_member		
회원정보 수정	/user/member/change_user_info		
내 목록	/user/my_stock_list		내가 저장한 목록
메인화면	/main		
게시물 작성	/post/content/create_view		컨텐츠 생성 화면
게시물 수정	/post/content/rewrite_view		컨텐츠 수정 화면
자유게시판 목록	/post/content/board		컨텐츠 목록 화면
게시물 상세	/post/list_detail_view		컨텐츠 상세 화면
종목현황	/post/stock_detail_view		종목 현황/정보
분석 칼럼	/post/column_detail_view		전문가 분석 칼럼
분석 칼럼 리스트	/post/column_list		전문가 분석 칼럼 리스트

2) 회원가입			
URL : /user/sign_up			
method : post			
parameter			
parameter 이름	데이터 타입	NULL 여부	설명
loginId	String	N	사용자 ID(e-mail)
password	String	N	사용자 비밀번호
userName	String	N	사용자 본명
nickName	String	N	사용자 활동명
응답값(JSON)			
성공시	{"result", "success"}		
실패시	{"result", "failure"}		

DANVESTING PROJECT
TROUBLE_SHOOTING & 개선할 점

프로그램 전체 TROUBLE SHOOTING & 부족한 점

이번 프로젝트를 진행하면서, 느낀 점을 아이작 뉴턴의 말을 인용해 한 마디로 표현하자면
“나는 거인의 어깨에 올라탄 난쟁이다.”입니다.

기억나지 않을 정도로 많은 크고작은 에러사항을 마주치고 trouble shooting을 하는 과정에서,
여러 개발자들이 옛날에 남긴 기록(stackoverflow를 포함한 구글 검색 결과)들을 참조해가며
혼자 프로젝트를 진행해 나갔지만, 나 혼자 프로젝트를 만든 것 같지 않은 느낌을 받았습니다.

Jsoup, Gson, RobinhoodAPI, GoogleChartAPI과 같은 라이브러리와 API를 개발해 놓은
여러 훌륭한 개발자 분들(거인들)과, 여러 에러를 잡는 방법을 기록해 놓은 많은 개발자 분들(거인들)이 없었다면
아예 이번 프로젝트 결과물이 없었을 것입니다.

다시 말해, 거인의 어깨에 올라탔기 때문에 미약한 실력으로 여기까지 프로젝트를 끌고 온 것임을 느꼈습니다.
또한 여러 거인들에 비교해, 파악조차 불가능할 정도로 많은 부족함을 갖고 있는 자신을 보게 되었습니다.

**Trouble
Shooting**

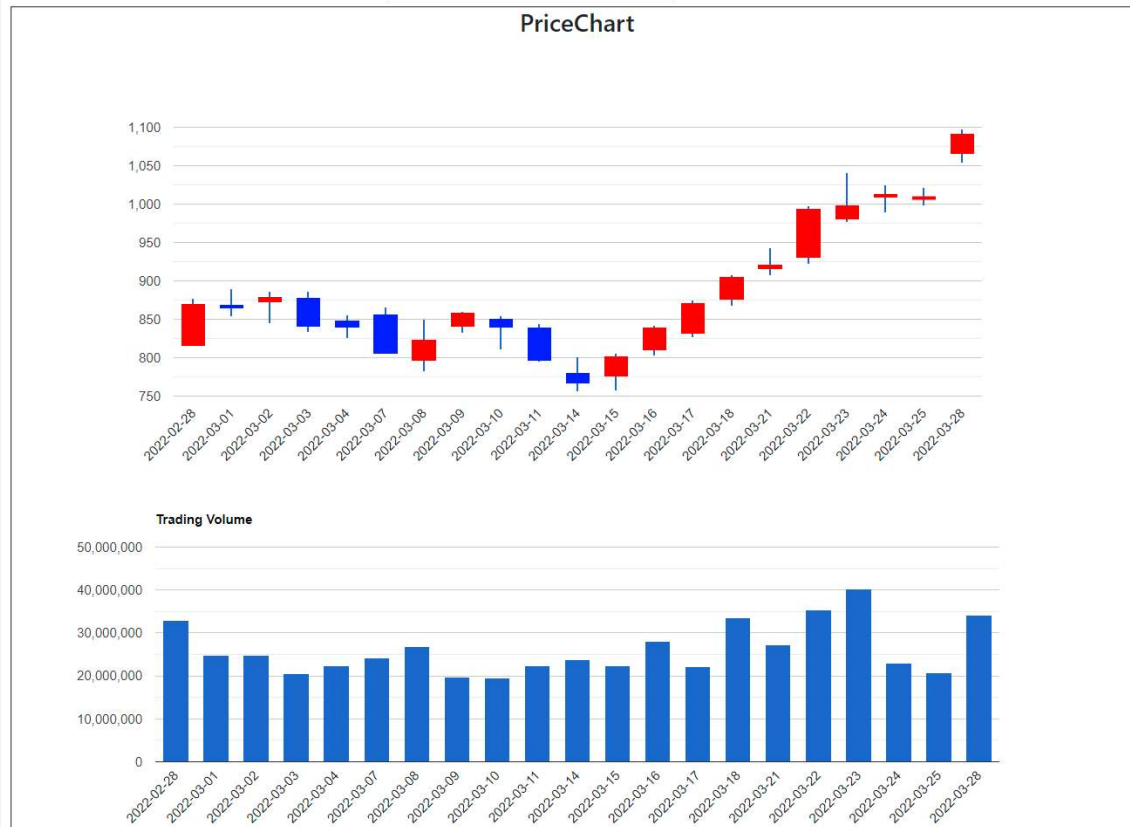
개선할 점

**Trouble
Shooting**

개선할 점

Ticker : TSLA

Search From To today(2022-3-29).



Trouble Shooting

RestTemplate를 사용해 json데이터를 가져와서, String변수에 저장하는 것에는 성공했으나,

궁극적으로 구글차트API를 사용해서 차트를 표기하려면, 단순 문자열이 아니라 Key-value형식의 데이터로 가공해야 했습니다.

따라서 Gson 라이브러리를 사용해서 RestTemplate으로 가져온 String형 데이터를 가공했고

Gson라이브러리를 사용하면서 Json데이터를 JsonObject클래스, JsonPrimitive, JsonArray클래스형 변수에 넣는 과정에서 살짝 문제를 겪었으나 따로 test코드를 작성, 문제를 해결했습니다.


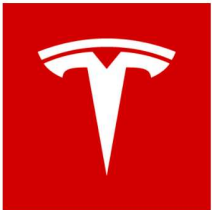
또한 Gson으로 가공한 데이터를 구글 차트로 적용하는 과정에서 UnixTimestamp형식의 시간을 <년-월-일>형식으로 변형하는 문제는 stack overflow 및 구글 검색을 통해 해결했으며,



Gson으로 가공한 데이터를 각각의 key-value값으로 떼어내서, 구글차트API에 적용시키는 방법은 구글차트API document 및 구글 검색을 참고하여 완성했습니다.

부족한 점

다른 주식관련 서비스 처럼 <1일>, <1주일>, <1달>, <3개월>, <6개월>, <1년>, <5년> 버튼으로 표기기간 조정하도록 개선해야겠습니다.

오늘 해당 종목이 하락중인 상태인지, 상승중인 상태인지를 티커 옆에 표기하도록 개선해야겠습니다.

Name	Tesla, Inc. Common Stock
Ticker	TSLA
Logo	
Icon	
address	3500 DEER CREEK RD PALO ALTO CA 94304
Phone-number	650-681-5000
Company homepage	https://www.tesla.com
<p>Founded in 2003 and based in Palo Alto, California, Tesla is a vertically integrated sustainable energy company that also aims to transition the world to electric mobility by making electric vehicles. The company sells solar panels and solar roofs for energy generation plus batteries for stationary storage for residential and commercial properties including utilities. Tesla has multiple vehicles in its fleet, which include luxury and midsize sedans and crossover SUVs. The company also plans to begin selling more affordable sedans and small SUVs, a light truck, a semi truck, and a sports car. Global deliveries in 2021 were a little over 936,000 units.</p> <p>Company description</p>	
Add This stock in list	

Name	Invesco QQQ Trust, Series 1	Company description
Ticker	QQQ	
Logo		
Icon		
address		
Phone-number		
Company homepage		

Trouble Shooting

앞서 외부 API로부터 Json데이터를 끌어와서, 라이브러리를 사용해 가공하고, 차트를 출력하는 것을 구현했기 때문에 비교적 수월하게 구현해 나갈 수 있었습니다.

다만, RobinhoodAPI는 QQQ와 같은 ETF에 대한 로고(운영사 로고)와 주소(운영사 주소), description(설명)데이터를 따로 제공하지 않기 때문에,

사용자가 ETF를 검색했을 때, 에러패이지가 나오지 않도록 Null값에 대한 예외처리를 했어야 했습니다.

부족한 점

다른 주식관련 서비스처럼, 해당 종목의 분배율(배당률),당기순이익과 같은 재무정보를 표시하고 싶었지만, XBRL파일을 연동하는 방법을 찾지 못해서 구현하지 못했습니다.

ETF종목을 검색했을 때, 그냥 null값이 아닌 해당 운용사의 정보를 가져오도록 수정하던가, Null값인 목록이 출력이 안되도록 수정해야 겠습니다.

```
ec2-user@ip-172-31-12-179:/usr/local/tomcat
Using username "ec2-user".
Authenticating with public key "imported-openssh-key"
Last login: Sat Mar 26 05:01:51 2022 from [REDACTED]

 _ _ _ _ _
|_| ( _ _ _ /   Amazon Linux 2 AMI
_|_|_|_|_|_|_|_|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-12-179 ~]$ cd /usr/local/tomcat
[ec2-user@ip-172-31-12-179 tomcat]$ sudo bin/catalina.sh stop
Using CATALINA_BASE:   /usr/local/tomcat
Using CATALINA_HOME:   /usr/local/tomcat
Using CATALINA_TMPDIR: /usr/local/tomcat/temp
Using JRE_HOME:        /
Using CLASSPATH:        /usr/local/tomcat/bin/bootstrap.jar:/usr/local/tomcat/bin/tomcat-juli.jar
Using CATALINA_OPTS:
NOTE: Picked up JDK_JAVA_OPTIONS:  --add-opens=java.base/java.lang=ALL-UNNAMED --add-opens=java.base/java.io=ALL-UNNAMED --add-opens=java.base/java.util=ALL-UNNAMED --add-opens=java.base/java.util.concurrent=ALL-UNNAMED --add-opens=java.rmi/sun.rmi.transport=ALL-UNNAMED
Mar 29, 2022 12:21:13 PM org.apache.catalina.startup.Catalina stopServer
SEVERE: Could not contact [localhost:8005] (base port [8005] and offset [0]). Tomcat may not be running.
Mar 29, 2022 12:21:13 PM org.apache.catalina.startup.Catalina stopServer
SEVERE: Error stopping Catalina
java.net.ConnectException: Connection refused (Connection refused)
    at java.base/java.net.PlainSocketImpl.socketConnect(Native Method)
    at java.base/java.net.AbstractPlainSocketImpl.doConnect(AbstractPlainSocketImpl.java:412)
    at java.base/java.net.AbstractPlainSocketImpl.connectToAddress(AbstractPlainSocketImpl.java:255)
    at java.base/java.net.AbstractPlainSocketImpl.connect(AbstractPlainSocketImpl.java:237)
    at java.base/java.net.SocksSocketImpl.connect(SocksSocketImpl.java:392)
    at java.base/java.net.Socket.connect(Socket.java:609)
    at java.base/java.net.Socket.connect(Socket.java:558)
    at java.base/java.net.Socket.<init>(Socket.java:454)
    at java.base/java.net.Socket.<init>(Socket.java:231)
    at org.apache.catalina.startup.Catalina.stopServer(Catalina.java:667)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
    at java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.base/java.lang.reflect.Method.invoke(Method.java:566)
    at org.apache.catalina.startup.Bootstrap.stopServer(Bootstrap.java:391)
    at org.apache.catalina.startup.Bootstrap.main(Bootstrap.java:481)
```

Trouble Shooting

프로그램 수정 후, AWS에 재 배포 하는 과정에서 AWS서버에 오류가 났었습니다.

접속이 거절되도록 보안 조건이 걸려있는지 확인해 보고, 보안조건과 상관 없이, 일시적인 AWS서버 오류인 것으로 결론낸 다음,

AWS 서버 인스턴스 재부팅을 통해, 다시 정상화 시켰습니다.

부족한 점

경제적인 여유가 된다면, 더 좋은 서버 스펙을 이용하고 싶습니다.