SABU
SELL AND BUY

오픈소스SW기초 6분반

분산 투자 전략 지원, SABU

#주식 #분산 투자

사고8조

정보통계학과 32200472 김동혁
컴퓨터공학과 32211228 김태형
소프트웨어학과 32232597 엄세훈

목차

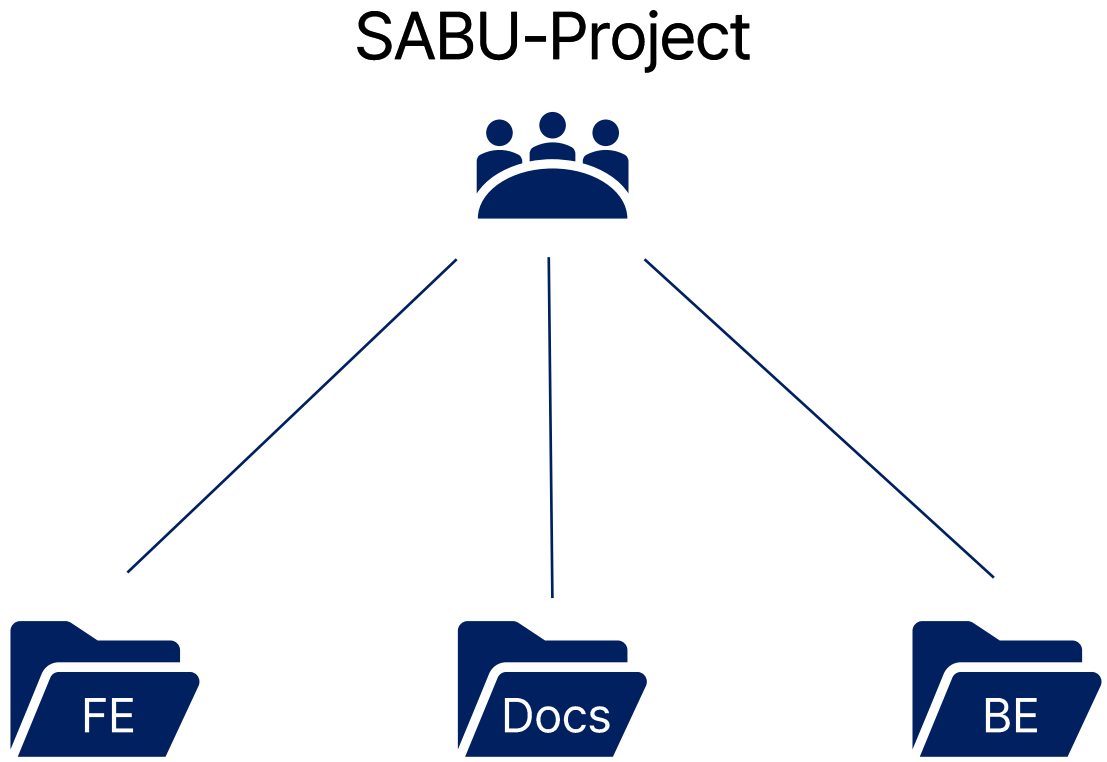
SABU
SELL AND BUY

1. GitHub 관리

GitHub - Project

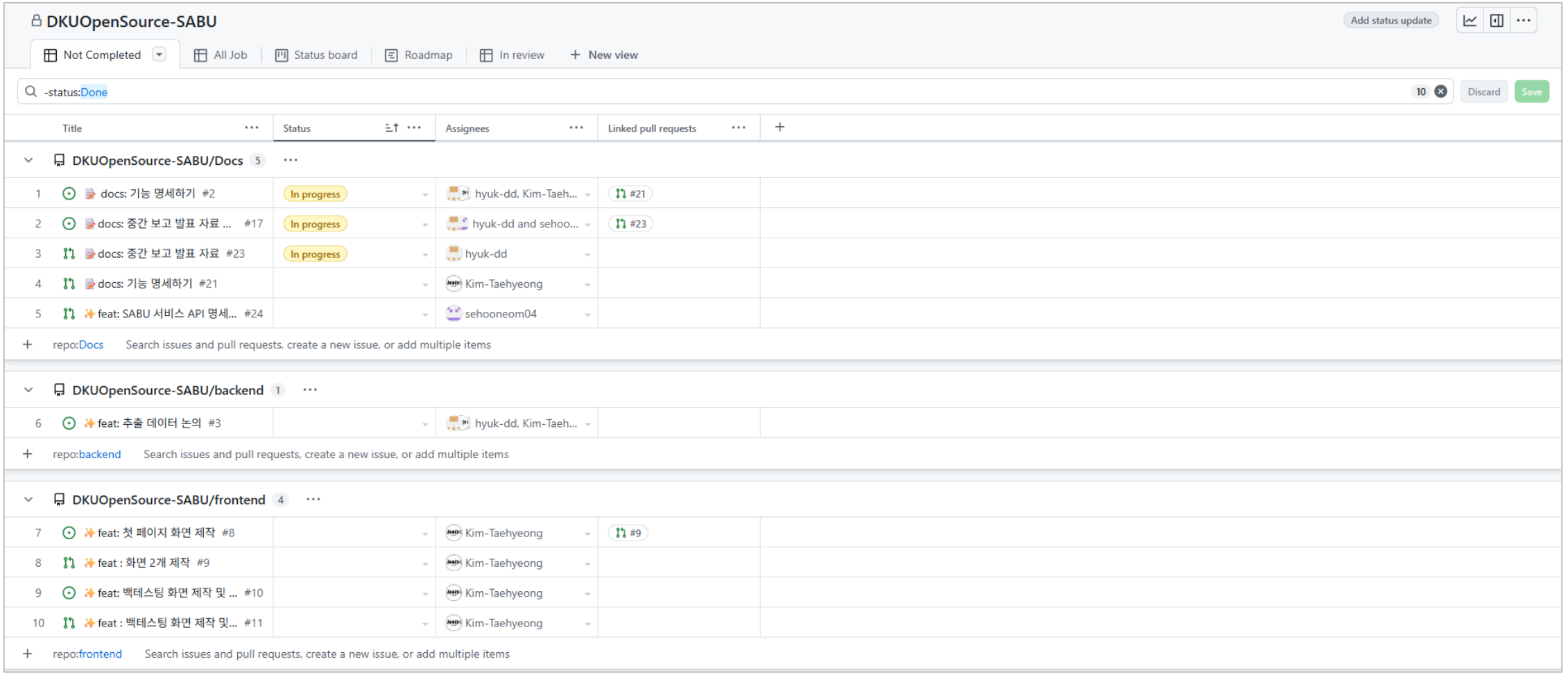
Project를 생성하여 작업 리스트 관리

- 전반적인 작업의 진행 상황, 완료 상태 등 한눈에 볼 수 있음
- 어떤 일을 하거나 작업이 필요한 것은 SABU-Project → 이슈 생성
- 하위에 각각 Frontend, Backend, Docs 레포지토리 존재
  - Frontend : Front와 관련한 논의, 코드 작성 등 작업 시행
  - Backend : Back과 관련한 논의, 코드 작성 등 작업 시행
  - Docs : 모든 프로젝트 관리, 문서 작업 시행



SABU
SELL AND BUY

GitHub - Project



DKUOpenSource-SABU

Add status update

Not Completed | All Job | Status board | Roadmap | In review | + New view

-status:Done

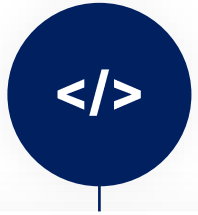
10 | Discard | Save

	Title	Status	Assignees	Linked pull requests
	DKUOpenSource-SABU/Docs 5			
1	docs: 기능 명세하기 #2	In progress	hyuk-dd, Kim-Taeh...	#21
2	docs: 중간 보고 발표 자료 ... #17	In progress	hyuk-dd and sehoo...	#23
3	docs: 중간 보고 발표 자료 #23	In progress	hyuk-dd	
4	docs: 기능 명세하기 #21		Kim-Taehyeong	
5	feat: SABU 서비스 API 명세... #24		sehooneom04	
+	repo:Docs Search issues and pull requests, create a new issue, or add multiple items			
	DKUOpenSource-SABU/backend 1			
6	feat: 추출 데이터 논의 #3		hyuk-dd, Kim-Taeh...	
+	repo:backend Search issues and pull requests, create a new issue, or add multiple items			
	DKUOpenSource-SABU/frontend 4			
7	feat: 첫 페이지 화면 제작 #8		Kim-Taehyeong	#9
8	feat : 화면 2개 제작 #9		Kim-Taehyeong	
9	feat: 백테스팅 화면 제작 및 ... #10		Kim-Taehyeong	
10	feat : 백테스팅 화면 제작 및... #11		Kim-Taehyeong	
+	repo:frontend Search issues and pull requests, create a new issue, or add multiple items			

SABU
SELL AND BUY

컨벤션

컨벤션을 이용하여
협업을 진행하면서 생기는
불필요한 커뮤니케이션을 줄이고
원활한 작업 진행



코드 컨벤션

커밋 컨벤션

이슈 컨벤션

SABU
SELL AND BUY

코드 컨벤션

Frontend

코드 컨벤션

사용 언어

- JavaScript

기본 규칙

- 들여쓰기(Indentation): 2칸 (스페이스 사용)
- 함수/변수명: snake_case
- 컴포넌트명: PascalCase
- 파일명 : PascalCase.jsx 또는 kebab-case.js
- 세미콜론: 붙이기
- 따옴표: "더블 쿼터"
- 상수: UPPER_SNAKE_CASE

컨벤션 예시

```
// UserCard.jsx
import React from "react";

const TYPE = 1;

function UserCard({ name }) {
  return <div className="user-card">{name}</div>;
}

export default UserCard;
```

- 사용 언어: JavaScript
- 들여쓰기, 함수/변수명, 컴포넌트명, 파일명 등 기본 규칙 지정
- Frontend 코드 컨벤션 예시

SABU
SELL AND BUY

코드 컨벤션

Backend

코드 컨벤션

사용 언어

- Python

기본 규칙

- 들여쓰기(Indentation): 4칸 (스페이스 사용)
- 최대 줄 길이: 79자
- 함수/변수명: snake_case
- 클래스명: PascalCase
- 상수: UPPER_SNAKE_CASE
- 불필요한 공백 금지:
  - a = 1 (O) / a=1 (X)

정렬, 줄바꿈

- 불필요한 줄바꿈 피하기
- 함수 간에는 2줄 띄우기
- import는 표준 -> 서드파티 -> 로컬 순서로 정리하고 각각은 빈 줄로 구분

컨벤션 예시

```python
import os
import sys

import requests

from my_module import something

class TestClass():
    print("Something")


def test_function():
    print("Something")
```

- 사용 언어: Python
- 들여쓰기, 최대 줄 길이, 함수/변수명, 클래스명 등 기본 규칙 지정
- 정렬 및 줄바꿈 규칙 지정
- Backend 코드 컨벤션 예시

SABU
SELL AND BUY

커밋 컨벤션

Commit Message Convention

커밋 메시지를 일관성 있게 작성하기 위한 가이드입니다.

커밋 타입 정리

타입	설명	이모지	예시 설명
feat	새로운 기능 추가	✨	로그인 기능을 새로 개발했다
fix	버그 수정	🐛	회원가입할 때 이메일 중복 체크가 제대로 안 되는 문제를 고쳤다
docs	문서 수정 (README 등)	📝	README 파일에 설치 방법을 추가했다
style	포맷팅, 세미콜론 누락 등 기능 변화 없는 수정	🎨	코드 스타일을 맞추기 위해 포맷을 정리하고 세미콜론을 추가했다
refactor	리팩토링 (동작에는 변화 없음)	♻️	중복된 API 호출 코드를 깔끔하게 리팩토링했다
test	테스트 코드 추가/수정	✅	로그인 기능을 검증하는 테스트 코드를 작성했다
chore	빌드 설정, 패키지 추가 등 기타 변경	🔧	프로젝트에 eslint 설정 파일을 추가했다
perf	성능 향상 관련 커밋	⚡	데이터 조회 속도를 빠르게 개선했다
ci	CI 설정 변경	👷	GitHub Actions 워크플로를 수정해서 자동 배포가 되게 했다

커밋 메시지 작성 규칙

1. 제목(Subject)

- 한 줄로 요약 (50자 이내)
- 첫 글자는 대문자로
- 마지막에 마침표(.) 사용하지 않기
- 동사 원형 사용 ("Fixes" 대신 "Fix" 사용)

2. 본문(Body)

- 제목과 본문은 한 줄 띄워 분리
- 본문은 "무엇을", "왜" 했는지를 자세히 설명
- 72자마다 줄 바꿈 권장

3. 꼬리말(Footer)

- 이슈 트래킹 시 사용
- 예시: `Closes #123`, `Related to #456`

커밋 메시지 예시

```
✨ feat: 사용자 로그인 기능 추가

사용자가 이메일과 비밀번호로 로그인할 수 있도록 인증 API를 추가했다.
이 기능을 통해 회원 전용 페이지 접근이 가능하다.

Closes #10
```

```
🐛 fix: 회원가입 시 이메일 중복 검사 오류 수정

이메일 중복 검사 API가 항상 true를 반환하던 문제를 수정했다.
이제 실제 데이터베이스를 조회하여 정확한 결과를 반환한다.

Related to #12
```

SABU
SELL AND BUY

커밋 컨벤션


```
################
# <타입> : <제목> 형식으로 제목을 아래 공백 줄에 작성하세요.
# 제목은 50자 이내로 작성합니다.
# 변경사항이 "무엇"인지 명확하게 작성합니다.
# 제목 끝에 마침표를 사용하지 않습니다.
# 첫 글자는 소문자 타입 후, 제목은 동사 원형으로 시작합니다.
# 예시: feat: 사용자 로그인 기능 추가



################
# 본문(Body)을 아래에 작성합니다.
# 제목과 본문 사이에는 반드시 한 줄 공백을 둡니다.
# 본문은 "무엇을", "왜" 변경했는지 상세히 설명합니다.
# 여러 줄 작성 시 "-"로 구분하며, 한 줄은 72자 이내로 유지합니다.



################
# 꼬릿말(Footer)을 아래에 작성합니다.
# 관련 이슈 번호를 명시할 때 사용합니다.
# 예시: Closes #10, Related to #12



################
# 타입 가이드
# ✨ feat : 새로운 기능 추가
# 🐛 fix : 버그 수정
# 📝 docs : 문서 수정 (README 등)
# 🎨 style : 포맷팅, 세미콜론 누락 등 기능 변화 없는 수정
# ♻️ refactor : 리팩토링 (동작에는 변화 없음)
# ✅ test : 테스트 코드 추가/수정
# 🔧 chore : 빌드 설정, 패키지 추가 등 기타 변경
# ⚡ perf : 성능 향상 관련 커밋
# 👷 ci : CI 설정 변경
################
```

gitmessage.txt

- gitmessage.txt 파일에 커밋 메시지 규칙에 맞는 템플릿을 제작하여 커맨드에서 'git commit -m ~~~' 할 필요 없이 커밋의 구조 통일화 및 작성 용이
- 커밋의 타입과 함께 제목 설정
- 본문에 '무엇을', '왜' 변경하였는지 상세히 설명
- 관련 이슈 번호 명시가 필요할 시 꼬릿말 작성

SABU
SELL AND BUY

이슈 컨벤션

이슈 작성 규칙

이슈 제목(Title)

- [타입] 간결하고 명확한 설명
- 타입은 커밋 타입과 통일: `feat`, `fix`, `docs`, `style`, `refactor`, `test`, `chore`, `perf`, `ci`
- 예시: `[feat] 사용자 로그인 기능 추가`

이슈 본문(Body)

설명

- 어떤 작업을 할 것인지 구체적으로 작성

작업 항목 (Checklist)

- [ ] 해야 할 작업을 체크리스트 형식으로 작성

기타 참고사항

- 관련 링크, 스크린샷 등 추가

name	about	title	labels	assignees
Default Issue Template	Default Issue Template	[feat, fix, docs, style, refactor, test, chore, perf, ci] 간결하고 명확한 설명		

설명

사용자가 이메일과 비밀번호를 통해 로그인할 수 있는 기능을 추가한다.

작업 항목

- [ ] 로그인 API 생성
- [ ] JWT 토큰 발급 기능 추가
- [ ] 로그인 실패 시 에러 메시지 반환

참고사항

- 디자인 시안: [링크]
- 관련 문서: [링크]

- 이슈 작성 규칙 지정
- 이슈 템플릿을 지정하여 이슈 작성 시 구조 통일화 및 작성 용이
- 커밋 타입과 동일하게 이슈 타입과 함께 제목 작성
- 본문에 논의할 내용 및 어떤 작업을 할 것인지 설명, 작업 항목 나열

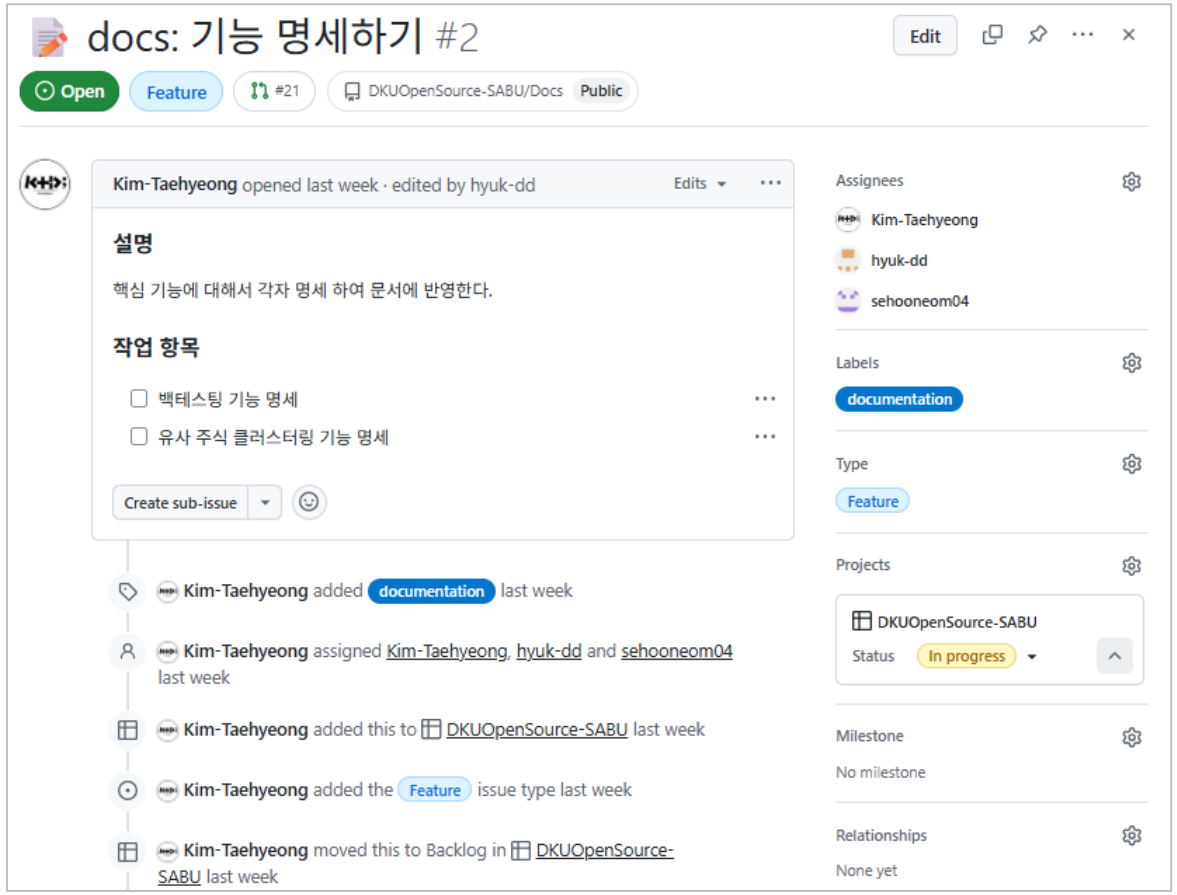
SABU
SELL AND BUY

GitHub 작업 순서

1. 해야 할 작업, 논의할 내용에 대해 Issue 생성
2. Issue 번호와 연결되는 Branch 생성
3. Issue와 관련한 작업 실행 후 생성한 Issue Branch에 Commit
4. Push 후 Pull Request 생성
5. PR에 대해 다른 두 팀원의 리뷰 및 승인, 마지막 검토자가 Merge 시행

GitHub 작업 순서

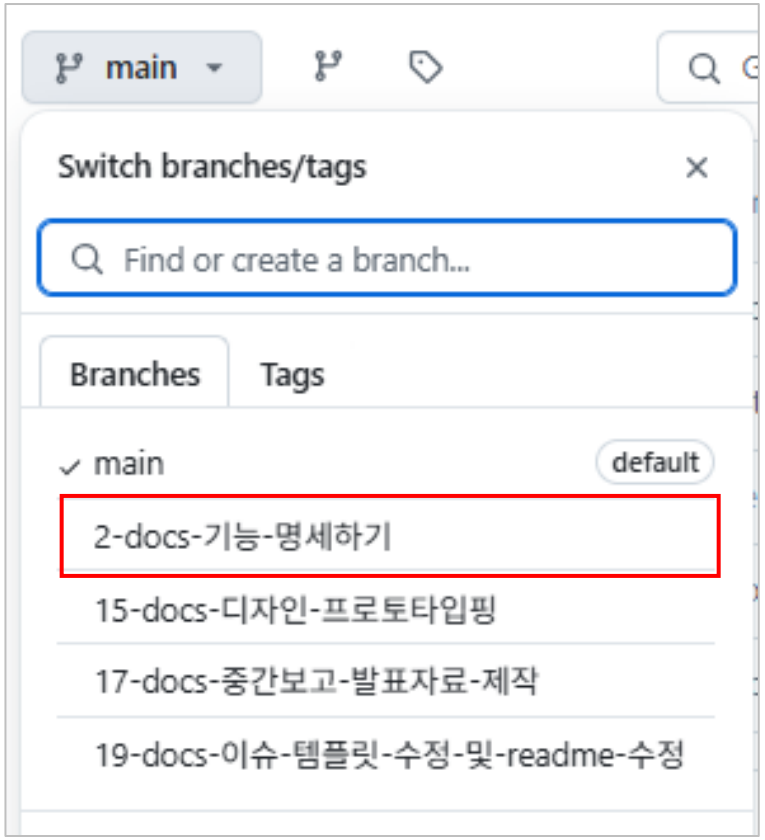
1. 해야 할 작업, 논의할 내용에 대해 Issue 생성



SABU
SELL AND BUY

GitHub 작업 순서

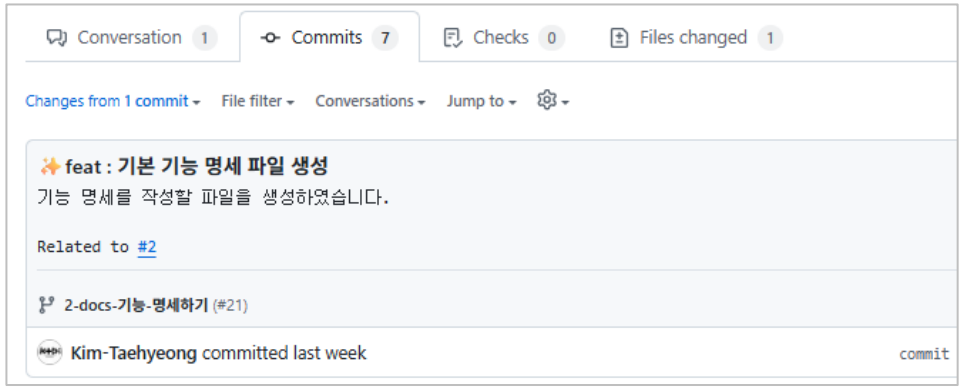
2. Issue 번호와 연결되는 Branch 생성



SABU
SELL AND BUY

GitHub 작업 순서

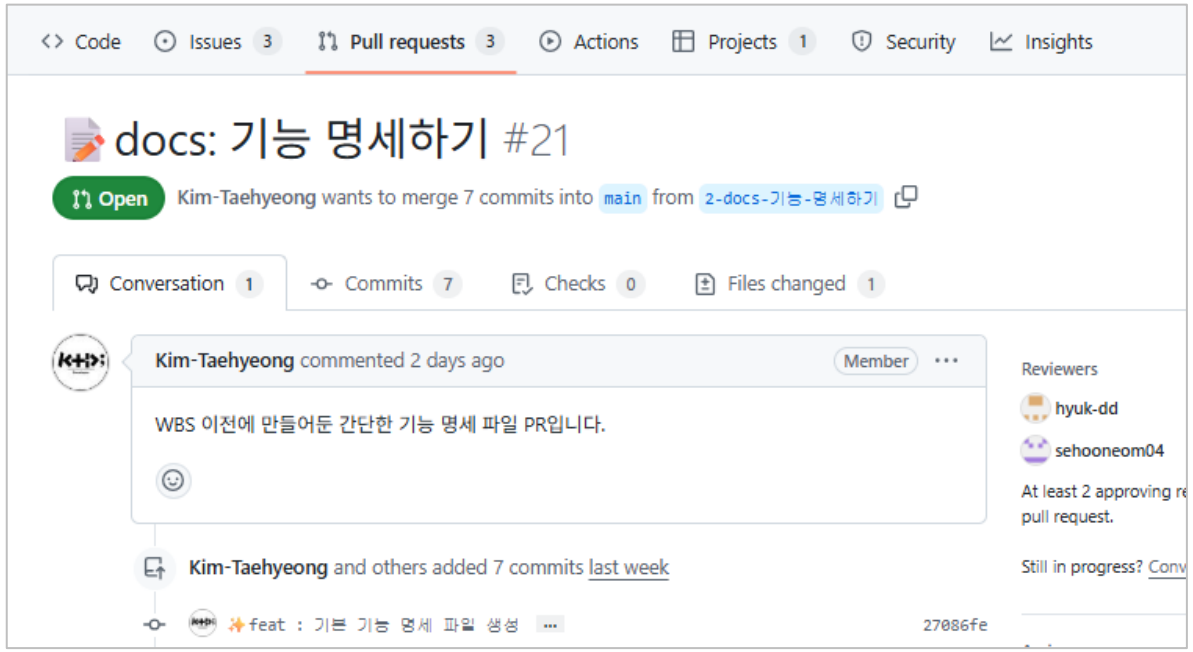
3. Issue와 관련한 작업 실행 후
생성한 Issue Branch에 Commit



SABU
SELL AND BUY

GitHub 작업 순서

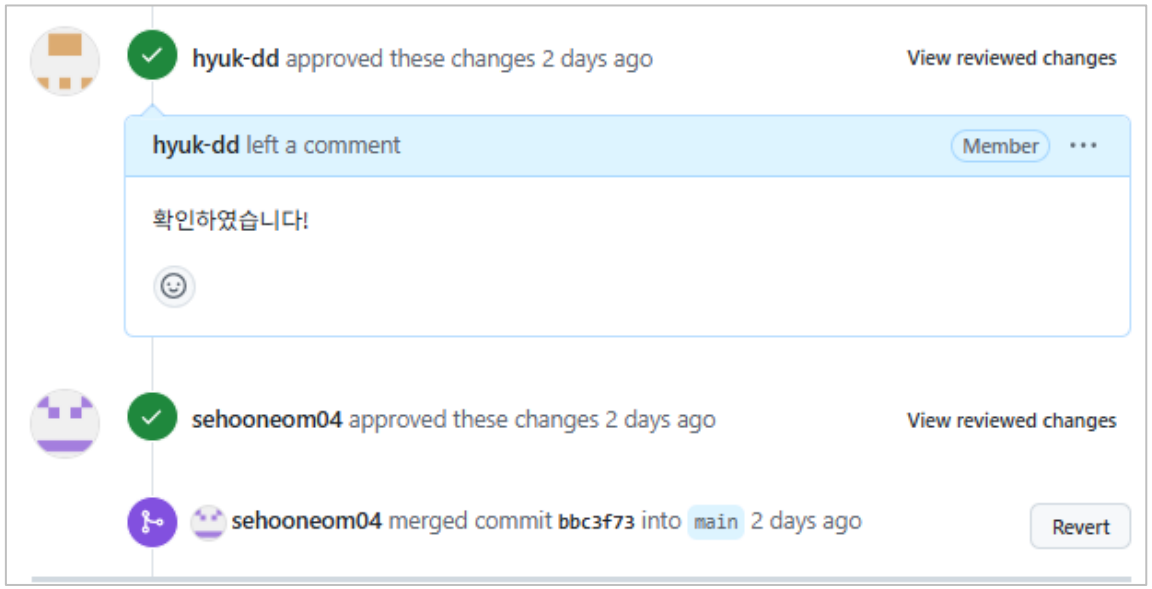
4. Push 후 Pull Request 생성



SABU
SELL AND BUY

GitHub 작업 순서

5. PR에 대해 다른 두 팀원의 리뷰 및 승인,
마지막 검토자가 Merge 시행



SABU
SELL AND BUY

2. 데이터 수집 & 모델링 계획

데이터 수집

- 우선적으로 클러스터링을 진행할 수 있도록 가격 지표부터 추출
  → 그 후 추출 가능한 데이터를 논의 후 넣어보며 모델을 최적화 하는 것으로 계획

- Alpha Vantage API → Tiingo API 변경
  + Nasdaq에서 데이터 수집

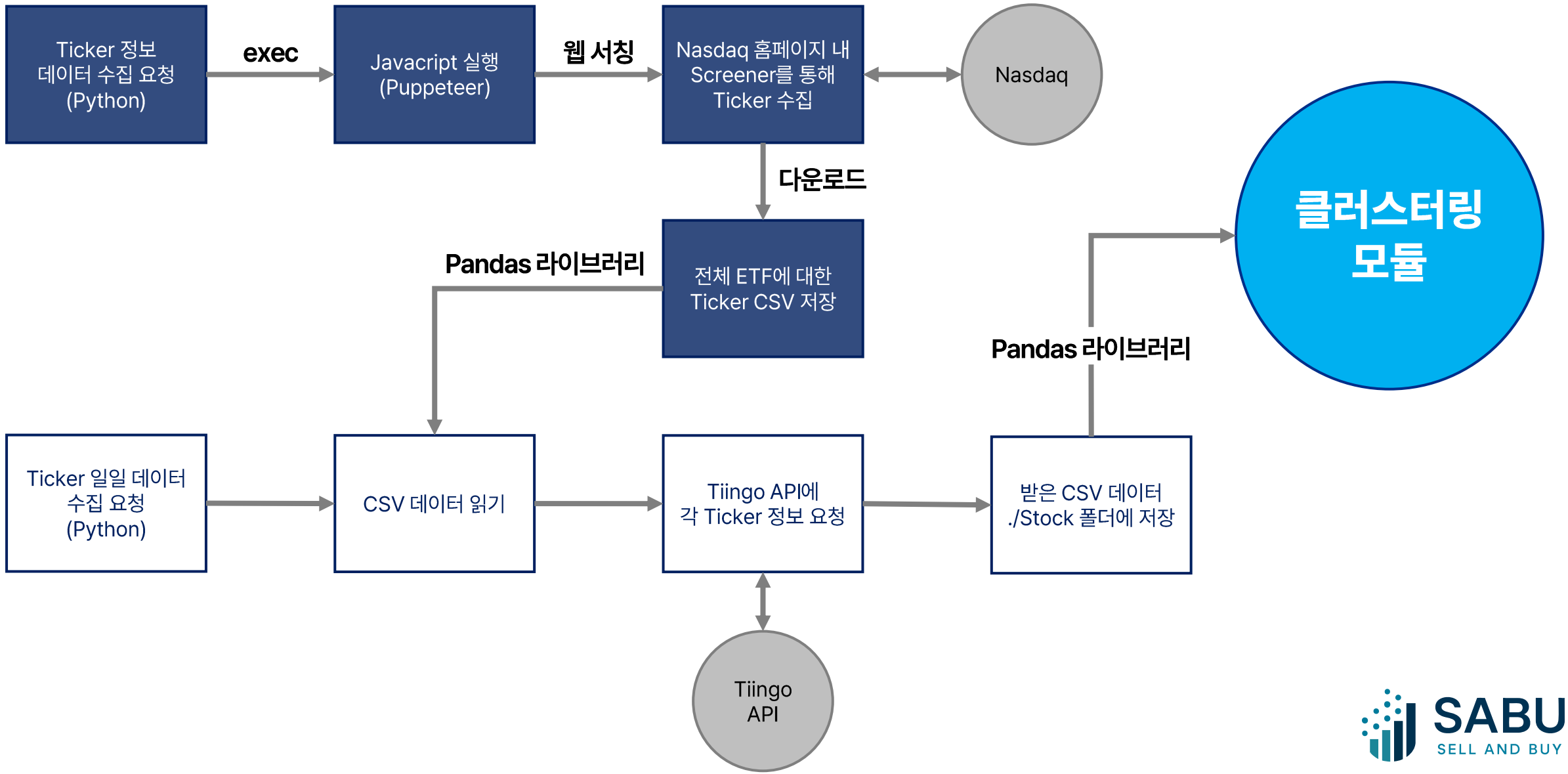
- 총 3,500개 데이터 다운 필요 → 약 3~4일 소요 예상

- Tiingo에서 각 Ticker 별 현재 수집되고 있는 데이터:
  date, open, close, high, low, volume, adjClose, adjHigh, adjLow, adjOpen, adjVolume, divCash, splitFactor
  날짜, 시가, 종가, 최고가, 최저가, 수정 종가, 수정 최고가, 수정 최저가, 수정 거래량, 현금 배당금, 주식 분할 비율

- Nasdaq에서 각 Ticker 별 현재 수집되고 있는 데이터:
  SYMBOL,NAME,LAST PRICE,NET CHANGE,% CHANGE,DELTA,1 yr % CHANGE
  티커 심볼, 종목(회사) 명칭, 종가, 전일 종가 대비 절대적 변화폭, 전일 종가 대비 변동률, 가격 변화량, 1년 전 대비 변동률

SABU
SELL AND BUY

데이터 수집



SABU
SELL AND BUY

클러스터링 구현 계획

- K-Means 알고리즘 사용
- 수집한 데이터 전체를 K-Means 클러스터링을 적용한 후, 사용자가 선택한 종목이 어떠한 클러스터에 포함되는지 시각화하여 보여줄 계획
- 차트 분석에 활용되는 지표인 이동 평균선, 수익률, 변동성, RSI, MDD, 거래량 등을 우선적으로 이용하여 클러스터링
- 수집된 데이터 이상치 및 결측치 처리
- 각 지표 정규화 진행 (필요 시 PCA를 이용하여 주성분만 이용)
- K-Means 알고리즘 시행, 사용자가 지정하는 클러스터링 개수에 따라 클러스터링 조정, 이에 따른 시각화 제공
- 추후 Hierarchical, DBSCAN의 방법을 사용한다면 코사인 유사도 행렬을 이용할 계획

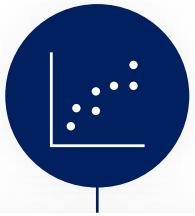
3. 서비스 설계

WBS

* WBS: Work Breakdown Structure

- 개발 전 서비스 설계를 위하여 MVP(Minimal Viable Product) 기준으로 최소한으로 구현해야 할 WBS 작성
- 사용자 기준, 각 기능별로 사용자가 어떤 기능을 사용할 수 있는지 구분
- 크게 클러스터링 기능, 백테스트 기능, 주식 데이터 수집
- 추상적인 기능들에 대해 CRUD 기준 세부적인 Use Case 추출
- 추출한 Use Case에 고유 프로세스 ID를 붙여 개발, Branch, Issue 관리 용이

WBS



클러스터링 기능

주식 간 지표를 기반으로
클러스터링 하여 시각화 제공
및 분산 투자 포트폴리오
제안

백테스팅 기능

사용자가 선택한 주식 조건에
맞추어서 과거 수익률
시뮬레이션을 해주는 기능

주식 데이터 수집

API를 이용하여
주식 과거 데이터 및
최대 기간 데이터 수집

SABU
SELL AND BUY

WBS

클러스터링 기능

SABU 시스템 WBS

No.	업무 영역 (Level 1)	Level 2	Level 3	프로세스 설명	프로세스 ID	비고
1	클러스터링 기능			ETF 간 지표를 기반으로 클러스터링하여 시각화 제공 및 분산투자 포트폴리오 제안	CLU	
2	클러스터링 기능	ETF 종목 코드 검색		사용자가 조회할 ETF의 티커(Ticker) 입력	CLU-001	
3	클러스터링 기능	선택 ETF 종목 목록 조회		선택한 종목의 목록과 각 종목별 기간 내 수익률 조회	CLU-002	
4	클러스터링 기능	선택 ETF 종목 삭제		선택한 종목 중 조회하고 싶지 않은 종목 삭제	CLU-003	
5	클러스터링 기능	클러스터 필터		사용자가 직접 클러스터에 대한 조건 설정		
6	클러스터링 기능	클러스터 필터	군집 수(K) 선택	K-Means 알고리즘 기반 사용자가 조회하고 싶은 군집의 개수 설정	CLU-004	
7	클러스터링 기능	클러스터링 결과 조회		선택한 종목, 클러스터 필터에 따른 클러스터링 시각화 결과 조회	CLU-005	
8	클러스터링 기능	클러스터 추천		사용자에게 분산 투자 포트폴리오에 제안할 수 있는 특정 클러스터 추천	CLU-006	

* 진행을 하며 기존 WBS 작성 내용 일부 수정됨

- 주식 티커 검색
- 선택한 주식 조회
- 선택한 주식 삭제
- 클러스터 군집 수 선택
- 클러스터링 결과 조회
- 클러스터 추천

SABU
SELL AND BUY

WBS

* 진행을 하며 기존 WBS 작성 내용 일부 수정됨

백테스팅 기능

9	백테스팅 기능			사용자가 선택한 ETF 조건에 맞추어서 과거 수익률 시뮬레이션을 해주는 기능	BTE	
10	백테스팅 기능	기간 설정		수익률 시뮬레이션을 확인하기 위한 기간 설정	BTE-001	
11	백테스팅 기능	백테스트 결과 조회		설정한 기간과 종목의 지표(누적 수익률, MDD, 변동성 등) 조회	BTE-002	종목은 CLU-002에서 설정한 종목에 따름
12	백테스팅 기능	자산 변동 추이 조회		시뮬레이션 된 자산 변동 그래프 조회	BTE-003	종목은 CLU-002에서 설정한 종목에 따름

- 백테스트 기간 설정
- 백테스트 결과 조회
- 자산 변동 추이 그래프 조회

주식 데이터 수집

13	ETF 데이터 수집			Alpha Vantage를 통해서 ETF 과거 데이터 및 최대 기간 데이터 수집	COL	
14	ETF 데이터 수집	ETF 정보 조회		티커에 기반한 ETF의 지표 및 정보 검색	COL-001	사용자 X
15	ETF 데이터 수집	데이터 저장		가격 지표, 섹터 정보 관련 데이터 저장	COL-002	사용자 X
16	ETF 데이터 수집	특정 기간 데이터 조회		특정 기간의 데이터 조회	COL-003	사용자 X

- 주식 데이터 수집
- 주식 데이터 저장
- 특정 기간 데이터 조회

SELL AND BUY

추후 추가 기능 논의

주가분석

- 시장 상황을 반영한 감정 분석 및 다양한 추가 분석 기능

그외추가기능

- 리더보드
- 실시간 커뮤니티 기능
- QR 코드로 만들어 저장하는 기능
- 분산 투자 전략 공유 및 순위 매기기 기능

클러스터링

- 사용자가 선택한 종목끼리만 클러스터링 되도록 하는 기능
- K-Means 알고리즘 사용시 엘보우 기법을 이용한 초기 Default 군집 개수 지정 기능
- K-Means 알고리즘 이외에 다른 클러스터링 방식 추가하여 사용자가 선택하는 기능
- 사용자가 궁금해하는 지표를 직접 선택하여 선택한 지표에 따른 클러스터링 결과 제공 기능

4. API 명세

API 명세 내용

기능 분류	기능 설명	HTTP Method	Endpoint	pid	Request	Response
[클러스터링]	ETF 종목 코드 검색	GET	/etfs/search?q={ticker}	CLU-001	ticker	ticker, name
	클러스터 실행	POST	/clustering/run	CLU-003	ets, n_clusters	message, cluster_ids
	클러스터링 결과 조회	GET	/clustering/results	CLU-004		clusters
	클러스터 추천	GET	/clustering/recommendation	CLU-005		recommended_etfs
[백테스팅]	백테스트 실행	POST	/backtest/run	BTE-001	etfs, start_date, end_date, initial_capital, fee_rate	message, backtest_id
	백테스트 결과 조회	GET	/backtest/results?backtest_id={id}	BTE-002		
	자산 변동 추이 조회	GET	/backtest/variation?backtest_id={id}	BTE-003		dates, estimated_assets

SABU
SELL AND BUY

API 명세 세부 설명

- **기능 분류**: 기능이 속한 업무 영역(ETF 검색, 클러스터링, 백테스트 등)
- **기능 설명**: 기능의 목적 및 주요 동작
- **HTTP Method**: API 요청에 사용되는 HTTP 메서드 (GET, POST 등)
- **EndPoint**: 기능 호출을 위한 경로
- **Request**: 요청 시 전달해야 할 파라미터나 JSON 형식
- **Response**: 응답으로 반환되는 데이터 구조 및 주요 필드
- **pid** :각 기능을 구별하기 위해 부여한 식별 번호
- request와 response는 실제로는 JSON 형식, 우선 초안이기에 어떤 필드인지만 간단하게 정리

5. Open Source

백테스팅 Open Source

mementum/
backtrader

Python Backtesting library for trading strategies

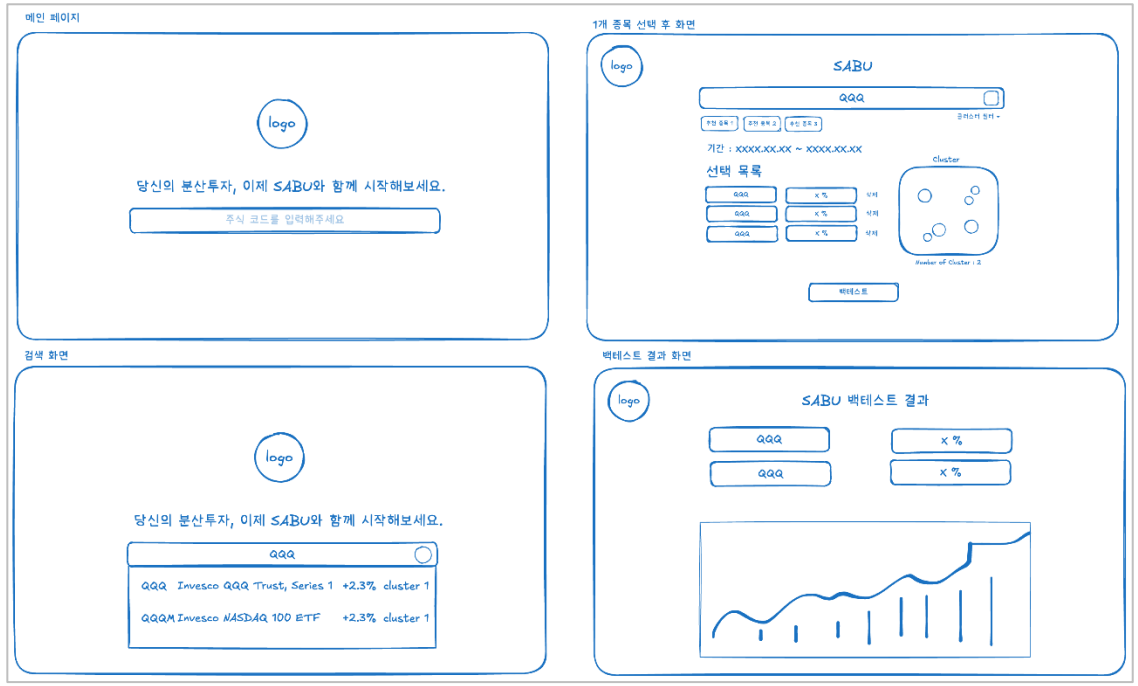
52 Contributors | 3k Used by | 17k Stars | 4k Forks

- **BackTrader** 사용
- 사용 언어가 모두 Python → Python 환경에서 바로 사용 가능하여 적합
- cereno.plot()으로 시각화가 간편
- 튜토리얼이 많아 백테스팅 구현을 할 때 참고 자료 풍부

SABU
SELL AND BUY

6. 와이어프레임

UI 스케치

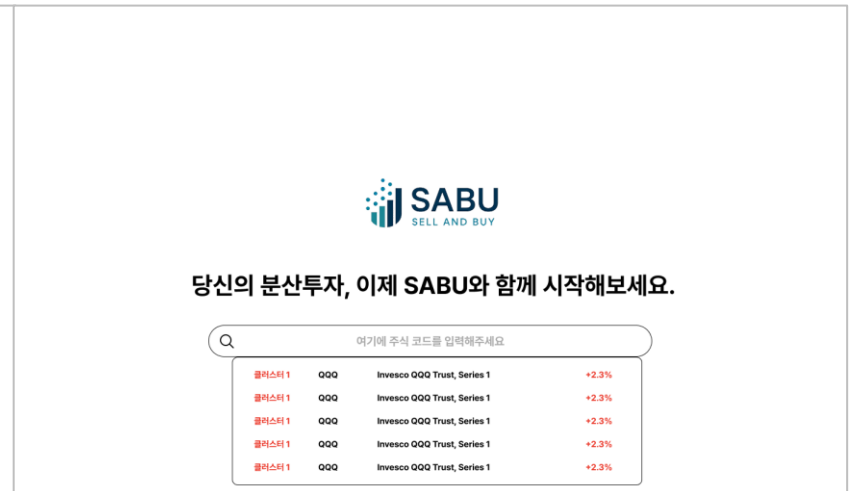


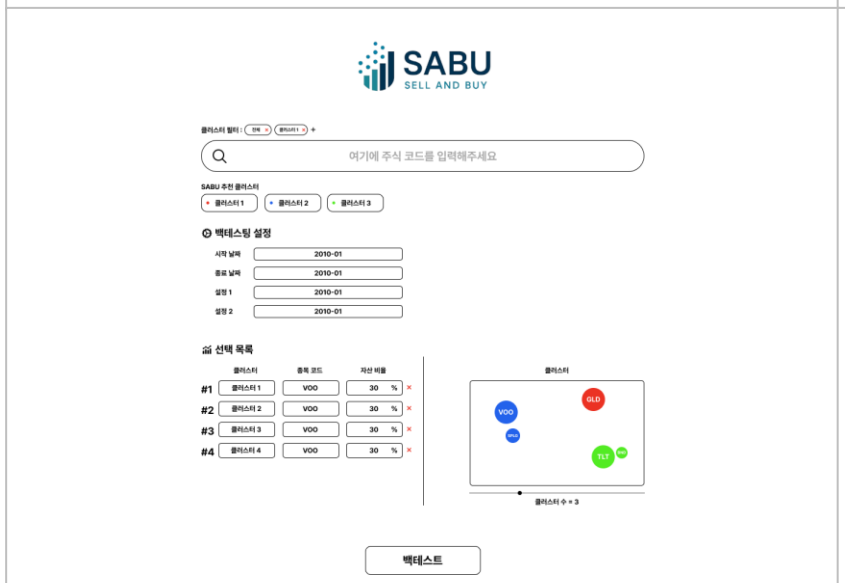
초기 UI 스케치

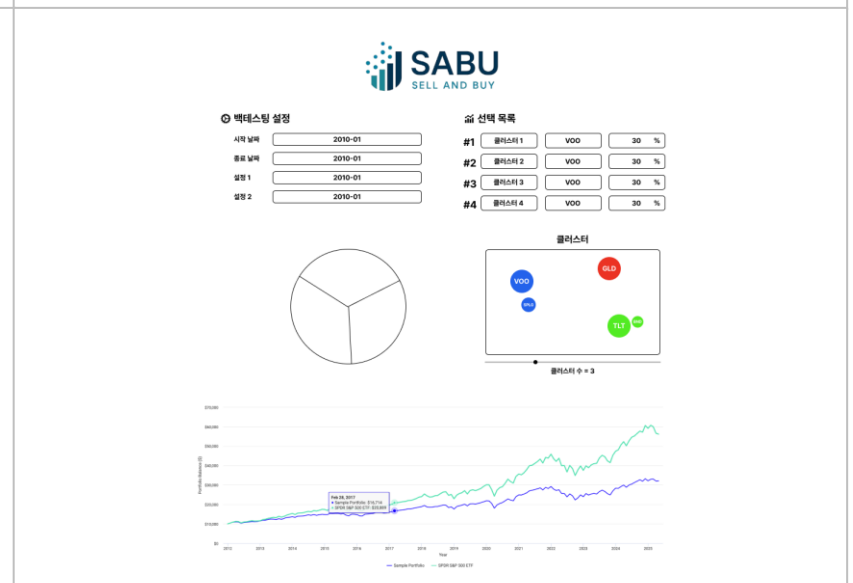
- 초기 UI 스케치에 따라 와이어프레임 구성
- 메인 화면, 검색 중 화면, 종목 1개 선택 후 화면(클러스터링 결과, 백테스트 조건 설정), 백테스트 결과 화면

SABU
SELL AND BUY

프로토타입







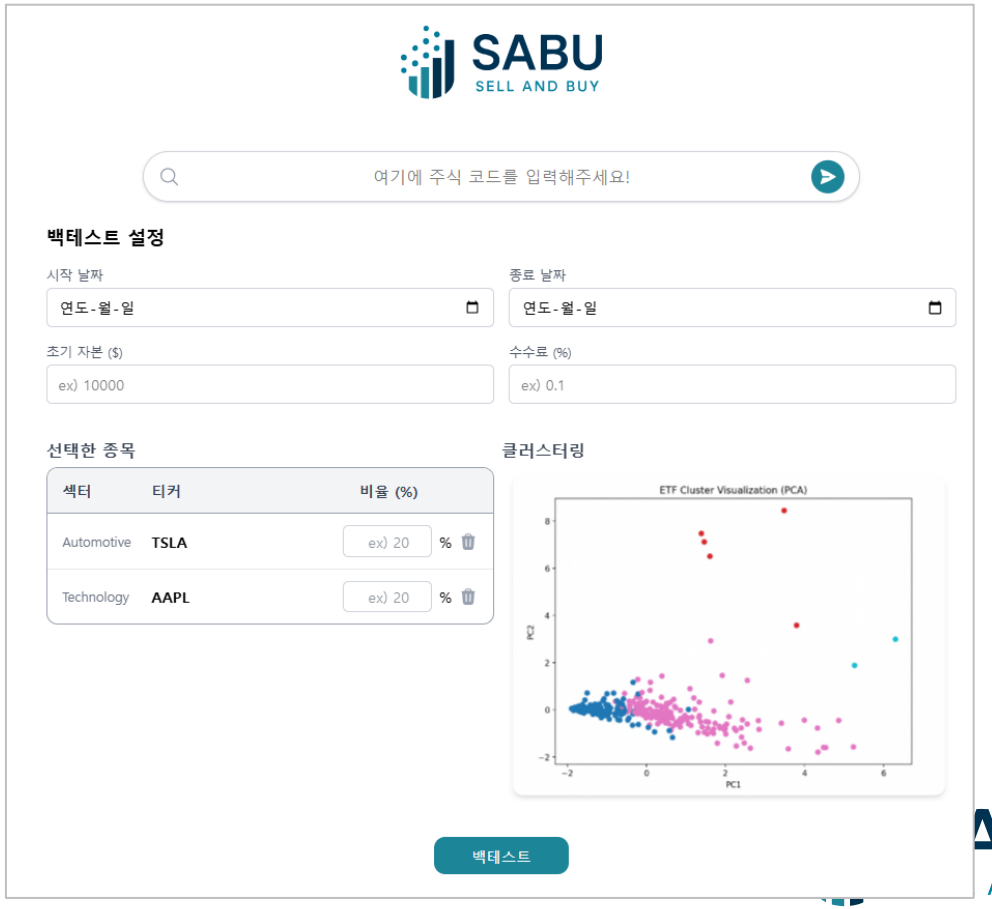
화면 명세, 도표, 그래프의 경우
아직 정확한 Feature가 나오지 않아
개략적으로 표현

SABU
SELL AND BUY

시연

https://dkuopensource-sabu.github.io/frontend/

* 실제 수치 %, 들어가는 동작들에 대해서는 아직 제대로 되어 있지 않음
  이는 프로토타입이며, API 연동이 필요한 부분에 대해서는 목업 데이터로 채워두었으므로 실제와는 무관



Q&A