2024 Project

Automatic Stock Prediction Program

Stock Way

Content

Team Position_프로젝트 팀원

한 석 희

프론트 엔드:
Login & Join Page Popup 개발

백 엔드:
금융데이터 수집 및 관리
금융데이터 예측 모델 개발 및 실행
전체 Data Base 설계 및 관리
로그인 및 회원가입 시스템 개발

테스트 :
GUI TEST 코드 작성
DB·금융데이터상호작용TEST코드작성
머신 러닝 예측 정확도 TEST 코드 작성

프로젝트 문서화 :
TEST REPORT 작성
프로젝트 PPT 작성

박 이 슬

프로젝트 관리 실무:
팀 업무 파트 분배

프론트 엔드:
전체 프로젝트 UI 설계 및 개발

백 엔드:
로그인 정규 표현식 적용

디자인 :
Login & Join page 개발
Main window 전체 디자인 개발

테스트 :
TEST CASE 시나리오 작성 및 실행
GUI·DB TEST 초안 코드 작성

프로젝트 문서화 :
요구사항 조사서 작성
WBS 작성
TEST REPORT 작성
프로젝트 PPT 작성

김 정 빈

프론트 엔드:
Main Window,
Sub Window 데이터 연결
데이터 시각화를 통한 그래프 구현

백 엔드:
GUI 삽입 금융데이터 크롤링

디자인 :
Main Window Qt Designer 적용
Sub Window Qt Designer 적용

테스트 :
DB연결및데이터관리TEST코드작성
금융데이터 스크래핑 TEST 코드 작성

프로젝트 문서화 :
프로젝트 PPT 작성

장 으 뜸

프론트 엔드:
Login & Join Page Popup 개발

디자인 :
Login & Join page 개발
Main window 전체 디자인 개발

테스트 :
DB 상호작용 금융데이터 TEST 진행

프로젝트 문서화 :
TEST REPORT 작성

Project Outline_프로젝트개요

프로젝트 개요
금융데이터 머신 러닝을 통한 주식 예측 프로그램 개발

※GUI 구현 화면

개발 배경

최근 빅데이터의 수집, 저장, 분석 기술의 발전으로 미래 예측 데이터 프로그램에 대한 수요가 증가함에 따라 웹사이트를 통해 쉽게 접근할 수 있는 금융 데이터를 활용하여, 사용자가 원하는 주식 정보를 한 눈에 쉽게 확인할 수 있는 주식 예측 프로그램을 개발하게 되었습니다.

목표

지도 학습을 통한 정확도 80%이상의 예측 데이터 생성

- ✓ 금융 데이터를 효율적으로 분석하여 사용자에게 유용한 예측 정보 제공
- ✓ 원하는 정보를 손쉽게 확인할 수 있는 직관적인 인터페이스 제공

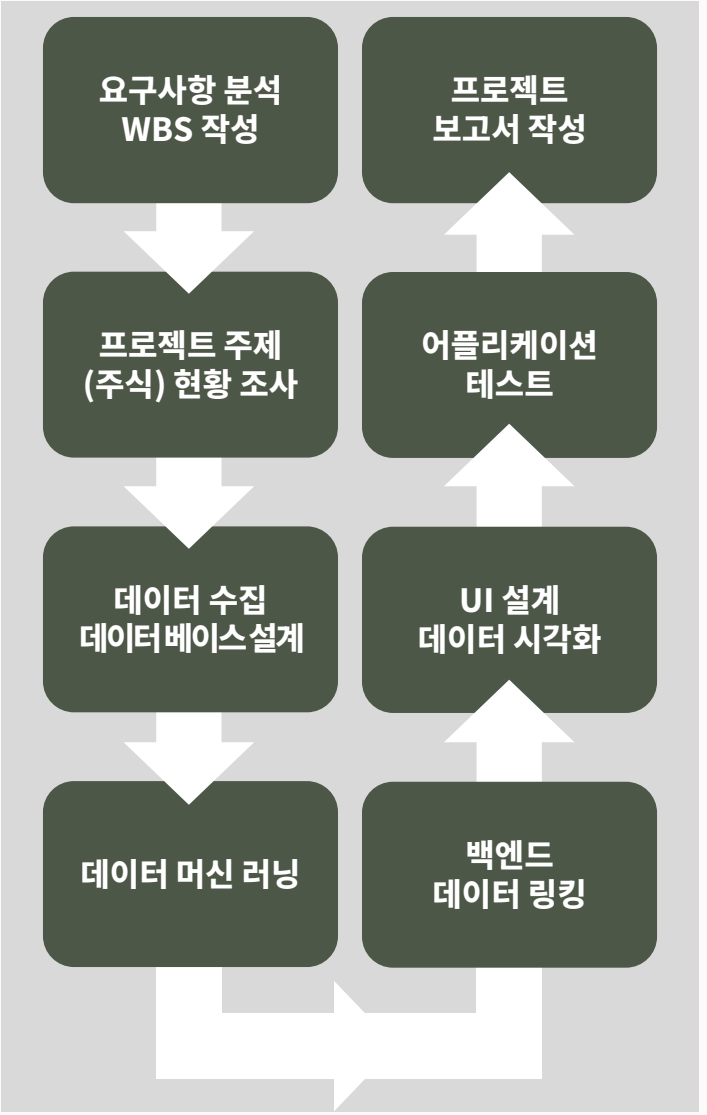
Project Order_wbs

Project	주식 데이터 예측 프로그램
Manager	
Write	박이슬
Version	
Issued date	2024-07-02

한석희
박이슬
김정빈
장으뜸

Work Break-down Structure								6월														
					schedule			1주							2주							
Procedures	Steps	Tasks	Details	담당자	시작일	종료일	산출물/비고	수	목	금	토	일	월	화	수	목	금	토	일	월	화	수
								19	20	21	22	23	24	25	26	27	28	29	30	1	2	3
1.0.0. 착수 및 프로젝트 관리																						
분석	1.1.0 프로젝트 관리																					
설계		1.1.1. 착수 OT(범위, 일정, 요구사항 등)		박이슬	6월 19일	6월 19일	WBS															
		1.1.2. WBS작성		박이슬	6월 20일	6월 20일	WBS															
	1.2.0 프로젝트 관리 실무																					
		1.2.1. 업무 파트 분배		박이슬	6월 20일	6월 20일	구글 스프레드 시트															
2.0.0. 분석/설계																						
	2.1.0 분석_조사																					
		2.1.1. PJ 데이터 조사	주식 관련 데이터 및 라이브러리 활용 사례 조사	ALL	6월 19일	6월 19일	구글 스프레드 시트															
		2.1.2 PJ 요구사항 조사	어플리케이션 요구사항 조사서 작성	박이슬	6월 19일	6월 19일	요구사항 조사서															
		2.1.2 레퍼런스 조사	GUI 구현 관련 레퍼런스 조사 및 ui 기획	ALL	6월 19일	6월 19일	ppt															
3.0.0. 디자인																						
	3.1.0 메인 디자인																					
		3.1.1. 메인디자인 시안 제안	메인 대시보드 레이아웃 설계	장으뜸	6월 20일	6월 20일	ppt															
		3.1.2 메인디자인 시안 작업	tkinter 메인 대시보드 GUI 1차 작업	김정빈	6월 20일	6월 24일	PjMain.py															
		3.1.3 메인디자인 시안 수정	메인 대시보드 GUI 디자인 추가 작업	박이슬	6월 23일	6월 24일	PjMain.py															
4.0.0. 프론트 개발																						
	4.1.0 프론트 개발_일반관리																					
		4.1.1. 로그인 페이지	tkinter 이용 1차 로그인페이지 구현	장으뜸	6월 20일	6월 21일																
		4.1.2 로그인 & 회원가입 페이지	kivy 활용 2차 로그인페이지 및 회원가입 페이지 구현	박이슬	6월 22일	6월 24일	PjKivyLogin.py															
		4.1.2. 비회원 페이지	비회원용 대시보드 구현	김정빈	6월 25일	6월 25일	pjNologin.py															
5.0.0. 백엔드 개발																						
	5.1.0 개발 설계부분																					
		DataBase 설계	회원데이터 기초 베이스 설계(Oracle)	장으뜸	6월 21일	6월 21일																
		DataBase 구축	회원데이터 베이스 변경 설계(Mysql) 및 연동	한석희	6월 24일	6월 25일	pjMyDb.py															
		DataBase 구축	주식데이터 백업용 테이블 설계 및 데이터 적재	한석희	6월 21일	6월 24일	pjMyDb.py															
		주식 데이터 예측 작업	주식데이터 수집 및 머신러닝	한석희	6월 20일	6월 24일	pjMachine.py															
		5.1.1. 상위 설계 및 산출물 제작	전체프로젝트 파일 취합 및 실행 테스트	한석희	6월 27일	7월 1일	테스트 리포트															
		5.1.2. 설계 및 산출물 컨펌	전체 프로젝트 검토 및 보완사항 정리	박이슬	7월 1일	7월 1일	구글 스프레드 시트															
6.0.0. 테스트 및 완료																						
	6.1.0 테스트																					
		6.1.1. 테스트 케이스 작성	테스트 케이스 및 시나리오 작성	박이슬	6월 24일	6월 26일	테스트리포트															
		6.1.2. 기능테스트	기능(GUI), 머신러닝,	한석희	6월 25일	6월 28일	테스트리포트, log data															
		6.1.3 성능테스트	어플리케이션 러닝타임 체크	한석희	6월 25일	6월 28일	테스트리포트, log data															
		6.1.4 외부연동테스트	데이터베이스 연결 테스트	김정빈	6월 25일	6월 27일	테스트리포트, log data															
		6.1.5 테스트 결과 보고서	테스트 결과 보고서 및 조치 사항 상세 작성	ALL	7월 1일	7월 1일	테스트리포트															
7.0.0. 완료																						
	7.1.0. 완료																					
		7.1.1. 내부 오픈	프로젝트 멤버별 어플리케이션 사용 및 AAR 진행	ALL	7월 1일	7월 1일																
	7.2.0 서류작업																					
		7.2.1 프로젝트 보고서 작성	프로젝트 보고서 상세 작성	ALL	7월 1일	7월 2일	ppt															
		7.2.2 포트폴리오	프로젝트 포트폴리오 작성	박이슬	7월 1일	7월 3일	ppt,pdf															

프로젝트 수행 순서



Project Order_요구사항 조사서

요구사항 정의서 (Requirment Definition)

시스템 (Application)	업무그룹	요구사항 ID	요구사항명	기능 요구사항	진행완료 내용 작성	프로세스 요구사항	담당자	화면 요구사항	보안 요구사항	성능 및 용량 요구사항	데이터 요구사항	기타 요구사항
주식 예측 자동화 프로그램	로그인	ST_LO01	**로그인화면 ID/Password 인증**	1. 로그인 시 대시보드 노출 2. 로그인 실패 시 팝업 생성 3. 비회원 버튼 클릭 시, 비회원용 팝업 생성	1. 로그인 성공 시 메인 대시보드 팝업 생성 2. 로그인 실패 시 팝업 생성 3. 비회원 버튼 클릭 시 비회원 노출용 대시보드 팝업	1. ID, PW 로그인 성공 실패 구현 2. 버튼 클릭 시 팝업 생성	장으뜸	로그인 팝업 레퍼런스 확인				
	회원가입	ST_JO01	**회원가입 구현**	1. 회원가입 정보 입력 시, 로그인 페이지에서 로그인 진행 2. 데이터베이스 내 고객정보 적재	GUI 화면 구현완료 회원가입 시 오라클 데이터베이스 내 적재 완료 고객데이터 암호화 작업 후 데이터 베이스 적재	중복체크를 통해 데이터베이스 적재 불가하도록 기능구현 Oracle -> Mysql로 전환 필요 정규표현식 적용	한석희,박이슬	회원가입 GUI 전체 색감 확인	데이터베이스 내 고객정보 암호화 필요			
	메인화면	ST_GU01	**프로젝트 네임**	1. 프로젝트 타이틀 노출	1. 프로젝트 타이틀 로고 작업 및 셋업	주식 관련 로고 생성	장으뜸	이미지 생성				
	메인화면	ST_GU02	**현재날짜**	1. 프로그램 실행 날짜 노출	1. GUI 화면 구현완료		김정빈	색감조정확인				
	메인화면	ST_GU03	**검색창 구현**	검색 데이터 입력 시 주식 데이터 자동 매핑 1. 향후 10일 날짜에 대한 예측 데이터 테이블삽입 2. 오늘날짜로부터 30일전까지의 데이터 테이블 삽입 3. 하단 그래프 구현(선형그래프,일봉그래프)	백엔드 코드 완료되었고, 리스트 정리 GUI 위젯 내 데이터 매핑작업 시각화 그래프 선택 후 작업 완료 페이지 전체적인 정돈 작업 진행		한석희,김정빈	데이터 테이블 색감 조정				
	메인화면	ST_GU04	**실시간 뉴스**	실시간 뉴스 리스트 테이블 삽입 -실시간 뉴스 클릭 시, 해당 사이트 뉴스페이지 이동 - 몇 개까지 리스트업 할건지 UI 보고 결정	뉴스 타이틀, url 크롤링 완료		김정빈					
	메인화면	ST_GU05	**코스피, 코스닥**	실시간 코스피, 코스닥 숫자데이터 노출 - 저가, 고가 색상변경 가능하면 적용 - 페이지 상태 봐서 그래프 등 캡쳐가능하면 추가	코스피, 코스닥 텍스트 데이터 크롤링		김정빈					
	메인화면	ST_GU06	**UI 디자인 작업**	kivy를 활용한 전체 GUI 색감 조정 Qtdesigner 를 통한 메인페이지 조정	로그인페이지 및 회원가입 페이지 GUI구현 완료 로고작업 후 삽입 완료		박이슬					
	메인화면	~~ST_GU07~~	~~**매수,매도 버튼 (추가로 진행**~~	~~매수, 매도 버튼 클릭 시 OpenApi를 이용한 작업 (키움 혹은 이베스트 증권 확인)~~	Open Api 연동 진행 miss로 인한 미구현		한석희					
	데이터베이스	ST_DB01	**데이터 백업**	데이터 베이스 내 주식 데이터 적재	회원가입 시 오라클 데이터베이스 내 적재 완료	Oracle -> Mysql로 전환 필요	한석희,장으뜸					
	어플리케이션 테스트	ST_TE01	**어플리케이션 테스트**	기능테스트 성능테스트 외부연동테스트	단위테스트 진행 완료 테스트 리포트 작성 완료	로그데이터	ALL					
	프로젝트 문서화	ST_DC01	**보고서**	프로젝트 문서화 작업(PPT)	프로젝트 문서 기초 초안 작업		ALL					

Report

주식 예측 자동화 프로그램

■ 기능적 요구사항

- 기존 주식 데이터를 활용한 예측 데이터 제공
- 회원 비회원별 페이지 분리
- 검색창을 통한 주식 데이터 종목 선택적 노출
- 주식 데이터 DB 적재를 통한 데이터 관리

■ 비기능적 요구사항

- 유저 데이터 보안을 위한 암호화 작업
- 전체 기능 테스트를 통한 결함 관리
- 주식 데이터 예측 정확성에 대한 테스트 진행
- 그래픽 사용자 인터페이스 환경(PC)

Project Order_프로젝트 수행절차 및 방법

프로젝트 사용 기술

Main & DB

· Python
Main 언어
외부라이브러리를 이용한
프로그램 개발

· MySQL
Main 데이터베이스
사용자데이터및주식정보저장

TEST

· Pytest
Main 테스트 라이브러리
GUI에 대한 테스트 수행 및
유연한 Assert 매서드의 사용

· Unitest
Sub 테스트 라이브러리
주식데이터에 대한 테스트를
진행할때다양한Assert매서드의
사용과 풍부한 기능을 이용

GUI

· Kivy
Main GUI
사용자들을 위한 로그인 및
회원가입 페이지 구현

· Qt5
Main GUI
로그인 사용자와 비로그인
사용자들을 위한 MainPage,
SubPage 구현

· tkinter
Sub GUI
메세지 박스를 활용한 경고
문구 전달 작업

Library

<Main library>

· Scikit learn
미래 주식 데이터를 예측하기
위한 지도 학습 및 선형 회귀
방식을 사용

· Selenium
주식회사의이름및주식종목코드를
사용자가 검색한 데이터에 맞추어
웹페이지 크롤링 작업 수행

<Sub library>

· FinanceDataReader
주식 데이터 및 각 나라 환율을 가져오기
위한 외부 라이브러리 사용

· Matplotlib
주식 데이터 캔들 스틱차트화를 위한
외부 라이브러리 사용

· Pandas
주식 데이터 중 필요한 데이터를 학습
시키기 위해 데이터 프레임 작업 수행

Project Order_어플리케이션 수행절차

프로세스 흐름

App Start

pjKivyLogin & pjMydb

pjQtui

Loop Start

pjMain or pjNologin

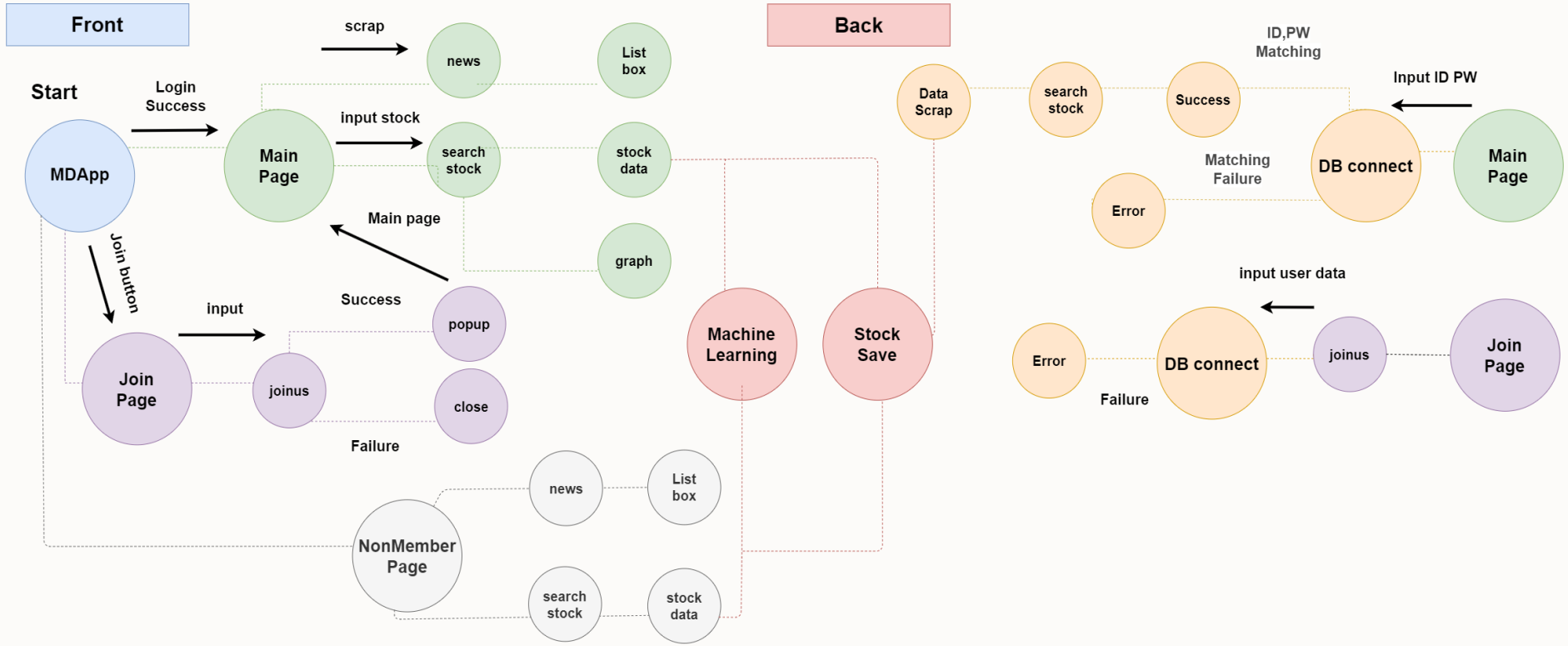
pjScrap

pjDownStock

pjMydb

pjDownStock

Repeat Loop



Project Code

```
# Use Selenium
news_link = []
news_title = []

news_title.append(driver.find_element(by=By.CLASS_NAME, value='tit').text)
news_link.append(driver.find_element(by=By.CLASS_NAME, value='tit').get_attribute('href'))


for i in range(1,3):
    for j in range(1,4):
        news_title.append(driver.find_element(by=By.XPATH,
        value='//*[@id="boxApp"]/div[1]/div[2]/ul['+str(i)+']/li['+str(j)+']/a').text)
        news_link.append(driver.find_element(by=By.XPATH,
        value='//*[@id="boxApp"]/div[1]/div[2]/ul['+str(i)+']/li['+str(j)+']/a')
        .get_attribute('href'))



news = pd.DataFrame(data=news_link,index=news_title)

kospi = driver2.find_element(by=By.XPATH,
        value='//*[@id="boxIndexes"]/div[1]/span[1]/strong').text
        + " " + \ driver2.find_element(by=By.XPATH,
        value='//*[@id="boxIndexes"]/div[1]/span[1]/p[1]').text
        + " " + \ driver2.find_element(by=By.XPATH,
        value='//*[@id="boxIndexes"]/div[1]/span[1]/p[2]').text

kosdaq = driver2.find_element(by=By.XPATH,
         value='//*[@id="boxIndexes"]/div[2]/span[1]/strong').text
         + " " + \ driver2.find_element(by=By.XPATH,
         value='//*[@id="boxIndexes"]/div[2]/span[1]/p[1]').text
         + " " + \ driver2.find_element(by=By.XPATH,
         value='//*[@id="boxIndexes"]/div[2]/span[1]/p[2]').text

# Class Ui_pjMain
def stock_market_color(self):

        if '+' in kospi: self.kospi.setStyleSheet('color: red;')
        else: self.kospi.setStyleSheet('color: blue;')

        if '+' in kosdaq: self.kosdaq.setStyleSheet('color: red;')
        else: self.kosdaq.setStyleSheet('color: blue;')
```

PjMain.py

Frontend - Main window

- 금융 데이터 웹 스크랩핑
- 금융 데이터 pyQt 활용 GUI 구현

Selenium

- XPATH를 활용하여 , 웹페이지 내 금융 뉴스 헤드라인, 뉴스 링크, 코스피, 코스닥 스크랩핑 실행
- 웹 스크랩핑 데이터 변수 저장
- 코스피·코스닥 등락률(+,-) 기준 컬러 적용

Project Code

```python
class User :
    def __init__(self, username, pw, uname, email):
        self.username = username
        self.password = pw
        self.uname = uname
        self.email = email

class UserBuilder :
    def __init__(self):
        self.reset()

    def reset(self):
        self.username = None
        self.password = None
        self.uname = None
        self.email = None

    def setUsername(self, username):
        self.username = username
        return self

    def setPassword(self, password):
        self.password = password
        return self



    def setUname(self, uname):
        self.uname = uname
        return self

    def setEmail(self, email):
        self.email = email
        return self

    def build(self):
        return User(self.username, self.password, self.uname, self.email)
```

PjMydb.py

Backend- Join user Data

- 회원가입 유저 데이터 빌더 패턴 구현

Builder Pattern

- 빌더 패턴 생성으로 코드 가독성 증대
- UserBuilder 호출 시 Reset 후 객체 데이터 저장
- User Class init가 각 객체에 맞는 데이터 저장

Project Code

```python
# password hashing
def hashPass(password, salt):
    return hashlib.sha256(password.encode() + salt).hexdigest()

# Save regist user
def regiUser(userInfo):
    try :
        conn = connectToDb()
        cur = conn.cursor()
        salt = os.urandom(16)
        hashed_password = hashPass(userInfo.password, salt)
        sql = "INSERT INTO usertb(id, pw, salt, uname, addr) VALUES (%s, %s, %s, %s, %s)"
        cur.execute(sql,
        (userInfo.username, hashed_password, salt.hex(), userInfo.uname, userInfo.email))
        conn.commit()
        cur.close()
        conn.close()

    except db.DatabaseError :
        pass

# Login user match
def login_user(loginId, loginPass) :
    try :
        conn = connectToDb()
        cur = conn.cursor()
        sql = "SELECT pw, salt FROM usertb WHERE id = %s"
        cur.execute(sql, (loginId,))
        result = cur.fetchone()

        if result :
            dbPass, salt = result
            salt = bytes.fromhex(salt)
            hashed_password = hashPass(loginPass, salt)
            if hashed_password == dbPass :
                return True
            return False

    except db.DatabaseError as e :
        print(e)
```

PjMydb.py

Backend- Join user Save & Login Match

- 유저 데이터 암호화 후 DB 저장
- DB 내 회원가입 유저 로그인 시 매칭 작업

Builder Pattern

- 유저 가입 시 Builder pattern 데이터 암호화 작업
- 암호화 된 PW · salt 포함 한 전체 유저 데이터 저장

pymysql

- DB 내 저장 된 ID 기준 PW · salt 매칭

  →True, False 반환하여 로그인 작업 구현

hashlib & os

- os 무작위 난수와 sha256을 이용한 PW 암호화

Project Code

```
# 사용자가 검색한 주식데이터의 종목코드를 반환
def selectstock(self):
  Cop_Id = pjScrap.findCopId(self.select_stock.toPlainText())

#pjScrap file
def findCopId(findData) :
  chrome_options = Options()
  chrome_options.add_experimental_option("detach", True)
  chrome_options.add_argument("--enable-javascript")
  chrome_options.add_argument("--headless")
  driver = webdriver.Chrome(options=chrome_options)
  wait = WebDriverWait(driver, 5)

  driver.get("https://finance.naver.com/")
  search = wait.until(EC.presence_of_element_located((By.XPATH,'//*[@id="stock_items"]')))
  search.click()
  search.send_keys(findData) # 사용자가 검색한 데이터 입력
  search.send_keys(Keys.RETURN)
  time.sleep(1)

  if driver.current_url.startswith("https://finance.naver.com/search/"):
        count = driver.find_element(By.CLASS_NAME, 'result_area')
        count = count.text[-3]
        if int(count) == 0 :
            driver.close()
            retur False
        else :
            clickList = wait.until(EC.presence_of_element_located((By.XPATH,
            '//*[@id="content"]/div[4]/table/tbody/tr[1]/td[1]/a')))
            clickList.click()

    copId = driver.current_url.split("=")[-1]
    copName = driver.find_element(By.XPATH, '//*[@id="middle"]/div[1]/div[1]/h2/a').text
    saveCop(copId, copName)

    driver.close()
    return copId
```

PjScrap.py

Backend- Find Stock ID & Corporation

· 사용자 검색 기반 종목 코드 및 회사 이름 웹 스크랩핑

Selenium

· Chrome 웹 드라이버 이용 네이버 증권 정보 접속

· 사용자 검색 기반 종목 코드 및 회사 이름 스크랩

스크랩

Time

· 웹페이지 간 로딩 시간 고려한 고정 대기시간 1초 부여

(EC[expected conditions] 명시적 대기 복합 사용)

Project Code

```python
def learnData(stockTarget) :
    target = str(stockTarget)
    startDate = (datetime.today() - timedelta(days=5*365)).strftime('%Y-%m-%d')
    startDate30 = (datetime.today() - timedelta(days=50)).strftime('%Y-%m-%d')
    endDate = datetime.today().strftime('%Y-%m-%d')
    stockData, stockData30 = pjDownStock.downloadStock(target, startDate,
                                                       startDate30, endDate)
    if stockData.empty :
        return None

    closeStock, realStock = pjDownStock.findCloseStock(stockData)
    closeStock = pd.DataFrame(closeStock)

# pjDownStock file
def downloadStock(tracker, startDate, startDate30, endDate) :
    stockData = fdr.DataReader(tracker, startDate, endDate)
    if stockData.empty :
        return None
    else :
        stockData.sort_index(inplace=True)
        stockDB = pjMydb.saveStock(tracker, stockData)
        stockData30 = fdr.DataReader(tracker, startDate30, endDate)
        return stockDB, stockData30

def findCloseStock(stockData) :
    closeStock = stockData[['SK_CLOSE']].copy()
    for i in closeStock[['SK_CLOSE']].tail().iloc :
        realStock = i.values[0]
    closeStock['NextClose'] = closeStock[['SK_CLOSE']].shift(-1)
    closeStock = closeStock.iloc[:-1]
    return closeStock, realStock

# pjMain file(Exchange rate using fdr)
today = datetime.utcnow().strftime('%Y-%m-%d')

exchange_JPY = fdr.DataReader('JPY/KRW', start=today, end=today)
exchange_USD = fdr.DataReader('USD/KRW', start=today, end=today)
exchange_EUR = fdr.DataReader('EUR/KRW', start=today, end=today)

JPY_To_KRW = 100 * exchange_JPY['Close'][0]
USD_To_KRW = exchange_USD['Close'][0]
EUR_To_KRW = exchange_EUR['Close'][0]
```

PjMachine.py

Backend-Download Stock & Find close Stock

- 5년치 주식 데이터 다운 및 종가 데이터 DB적재
- 스크랩 된 주식코드를 이용 한 주식 예측 지도 학습

timedelta

- 5년 전 날짜와 50일 전 날짜 데이터 산출

FinanceDateReader

- 금융 데이터 추출
- 미국, 일본, 유럽에 대한 환율정보 변수에 적재

Project Code

```
def saveStock(stockTarget, stockDf) :
    try :
        stockCode = "ST"+stockTarget
        conn = connectToDb()
        cur = conn.cursor()
        cur.execute("SELECT table_name FROM INFORMATION_SCHEMA.TABLES WHERE
                    table_schema = 'stock' AND table_name LIKE 'ST%'")
        table_list = [row[0] for row in cur.fetchall()]

        if len(table_list) >= 10 :
            table_list.sort()
            table_to_drop = table_list[0]
            copName = table_to_drop.upper()
            sql = 'delete from copname where stockid = %s'
            cur.execute(sql, (copName))
            cur.execute(f"DROP TABLE {table_to_drop}")
            print("delete copname & drop table")
            conn.commit()

        if stockCode.lower() not in table_list :
            cur.execute("create table %s(SK_DATE DATE, SK_OPEN FLOAT,
                        SK_HIGH FLOAT, SK_LOW FLOAT, SK_CLOSE FLOAT,
                        SK_VOLUME FLOAT)" % stockCode)
            for i, j in stockDf.iterrows() :
                sql = f"INSERT INTO {stockCode} VALUES(%s, %s, %s, %s, %s, %s)"
                cur.execute(sql, (i.strftime('%Y-%m-%d'), float(j['Open']),
                                  float(j['High']), float(j['Low']),
                                  float(j['Close']), float(j['Volume'])))
            conn.commit()

        cur.execute("select * from %s" % stockCode)
        columns  [des[0] for desc in cur.description]
        stockData = pd.DataFrame(cur.fetchall(), columns=columns)
        stockData = stockData.sort_values(by=['SK_DATE'])

        cur.close()
        conn.close()
        return stockData

    except db.DatabaseError as e :
        print(e)
```

PjMydb.py

Backend- Save Stock

- 유저 검색 기반 주식 데이터 DB 적재

pymydb

- 검색 된 주식 데이터 최대 10개 테이블까지 자동 저장

  (ST+종목코드 형태로 생성 DB내 종목테이블 보유 시 테이블 데이터 반환)

- 회사명 저장 테이블의 경우 최대 10개의 행까지 저장

pandas

- 머신 러닝을 위한 DB 테이블 dataframe 형태로 변환

Project Code

```
#def learnData
X = closeStock[['SK_CLOSE']]  # 종가 실제데이터
y = closeStock['NextClose'] # 종가 학습데이터
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.8, shuffle=False)
linearModel = LinearRegression()
linearModel.fit(X_train, y_train)
y_pred = linearModel.predict(X_test)
todayClose = pd.DataFrame(closeStock.iloc[-1][['SK_CLOSE']].values.reshape(-1,1),
             columns = ['SK_CLOSE'])
tomorrow_pred = linearModel.predict(todayClose)
r2 = r2_score(y_test, y_pred)

future_pred30 = [] # 30일의 예측데이터를 저장하기 위한 리스트
future_pred10 = [] # 10일의 예측데이터를 저장하기 위한 리스트
currentPrice = todayClose.values.reshape(-1,1) # 오늘날의 종가 데이터 2차원 배열 형태로 변형

for _ in range(30) : # 30일 이후의 종가데이터를 예측하기 위한 반복문
  nextPrice = linearModel.predict(currentPrice)
  future_pred30.append(nextPrice[0])
  currentPrice = nextPrice.reshape(-1,1)

futureDate30 = pd.date_range(start=datetime.today().strftime('%Y-%m-%d'),
               periods=30, freq='B')
futureFrame30= pd.DataFrame(future_pred30, index=futureDate30, columns=['PredClose'])

#futureFrame10은 상기 코드와 동일 과정 진행
futureFrame10['Multi'] = futureFrame10['PredClose'].diff()
futureFrame10 = futureFrame10.dropna()

#pjMain file(Machine learnig data return to def selectStock)
# 10일 뒤의 예측데이터를 treewidget 정렬
for date, value in zip(futureFrame10.index, futureFrame10.values) :
  __sortingEnabled = self.predicted_value.isSortingEnabled()
  item = QtWidgets.QTreeWidgetItem(self.predicted_value)
  item.setText(0, _translate("pjMain", date.strftime('%Y-%m-%d')))
  item.setText(1, _translate("pjMain", str(round(value[0], 2))))
  item.setText(2, _translate("pjMain", str(round(value[1], 2))))
  stockData30 = stockData30[::-1]

# 실제 30일 주식 데이터 treewidget 정렬
for date, value in zip(stockData30.index, stockData30.values) :
  __sortingEnabled = self.treeWidget.isSortingEnabled()
  self.treeWidget.setSortingEnabled(False)
  item = QtWidgets.QTreeWidgetItem(self.treeWidget)
  item.setText(0, _translate("pjMain", date.strftime('%Y-%m-%d')))
  item.setText(1, _translate("pjMain", str(round(value[0], 2))))
  item.setText(2, _translate("pjMain", str(round(value[1], 2))))
  item.setText(3, _translate("pjMain", str(round(value[2], 2))))
  item.setText(4, _translate("pjMain", str(round(value[3], 2))))
  self.treeWidget.setSortingEnabled(__sortingEnabled)
  stockData30 = stockData30[::-1]
```

PjMachine.py

Backend– Machine learning

- 머신 러닝을 이용한 주식 정보 예측 데이터 생성
- 주식 종목 검색 시 주식 데이터 GUI 구현

Sklearn

- tarin_test_split 모듈을 사용하여, 학습데이터와 테스트 데이터 분할로 Overfitting 방지
- LinearRegression모듈 사용으로 간추려진 데이터를 선형 회귀를 통해 시간에 따른 명확한 예측 데이터 생성

pyQt

- 사용자 검색 데이터(주식 종목) 기반 GUI 구현
- 검색 종목 10일 예측 체결가 ·30일간 일별 시세 TreeWidget 데이터 삽입

Project Test Report

Executed	*473*	PASSED(pytest 통과수)	354failed, 473 passed, 114 warnings, 2 errors in 5034.70s(1:23:55)
	354	FAILED(pytest 실패수)	
	943	*(total) Tests executed (실행된 전체 테스트 수)*	
	826	*(Passed+failed) (성공한테스트+실패테스트 합계)*	

class_name	Functions	Description	TCs Executed	TCs Passed	TCs pending	Priority	comment	Remarks
pjDownStock	downloadStock	fdr 주식데이터 서치 및 MysqlDB 저장	100%	100%	0	High	200개의 5년 주식 데이터를 finance-datareader 통해 다운로드 후 DB 저장 확인	
	findCloseStock	DB에 저장 된 주식데이터의 종가 추출	100%	96%	4%		200개의 데이터를 close(종가)만 추출하여 데이터 유무 확인	
pjKivyLogin (GUI test)	MDButton	클릭 및 새페이지 이동	100%	100%	0		버튼 클릭 및 페이지 이동 후 input 입력 진행 pjMain 환율데이터 warnigs 발생(시간설정 miss로 코드 수정)	
	MDTextField	데이터 입력 및 MySQL DB 데이터 적재	100%	100%	0		로그인 텍스트 삽입 정상작동여부 확인 완료	
pjMachine	learnData	머신러닝,예측 정확도	100%	89%	11%	High	5년 주식 데이터 200개를 지도학습 후 80%이상일 경우 Passed	
pjMain (GUI test)	webdriver	데이터 크롤링, 러닝타임	100%	100%	0	High		
	pyQt	GUI 화면 데이터 삽입 및 버튼실행	100%	100%	0			
pjQtUi (GUI test)	MainWindow	PyQt GUI 윈도우 표시	100%	100%	0		메인대시보드(Qt)실행 시, mathplotlib 에서 custom 폰트를 찾지 못함. ▶전체파일 Font 명 오류로 인해 DEBUG (pjNologin, pjKivyLogin , PjMain 폰트명 변경)	폰트명 변경 완료
pjMydb	Oracledb	데이터베이스 연결	100%	100%	0	High	데이터베이스 연결 테스트 및 테스트 데이터 ,실제데이터 매칭	
	saveStock	테이블 생성 및 검색	100%	100%	0		테스트 주식 데이터 생성	
pjMydb	saveStock	테이블 10개이상 적재 시 오래된 테이블 삭제	100%	100%	0		ramdom함수로 무작위 테이블 생성, 10개 이상시 제일 처음에 적재된 테이블 삭제	
	saveUser	회원가입 유저 데이터 적재,암호화	100%	100%	0		hashlib, os라이브러리로 사용자 pw암호화	테스트 진행 후 데이터 삭제 확인
	matchUser	가입 된 유저데이터 매칭	100%	100%	0		테스트 데이터를 가져와 salt를 바이트로 변환 동일한 salt로 비밀번호를 해시하여 해시된 비밀번호와 유저명이 일치하는지 확인	테스트 진행 후 데이터 삭제 확인
pjScrap	findCopId	주식종목 입력 시 종목코드 찾기	100%	100%	0		상장된 회사 400개 pjScrap.py findCopId함수로 주식이름 검색시 종목코드 찾음	

Test Report

※Pytest 진행으로 전체 log data 보관

GUI Test

- Button Test 진행으로 정상 클릭
- Login 시 input text 정상 삽입
- Mainwindow 정상 실행

Backend – Test

데이터 접근 및 적재 테스트

- 주식 데이터 정상 접근 및 DB 적재
- 금융 데이터 스크랩핑 시 종목 코드 매칭 확인
- DB 내 저장된 주식 데이터 정상 추출
- DB 연결 테스트
- DB 내부 주식 데이터 메모리 관리

데이터 처리 및 로직 테스트

- 회원가입 유저 암호화 수행 여부
- 머신 러닝 데이터 예측 정확도 확인

Project Evaluation_평가의견

한 석 희

목표 : 얻고자 한 것

· 팀원과의 협업 및 프로젝트 구축에 필요한 언어 숙지
· 주식 예측 데이터의 시각화 구현
· 전체적인 코드 구성의 최적화 작업

결과 : 실제로 얻은 것

· 업무 분배 및 팀원 간의 의견 조율 능력 향상
· Python 언어 사용 능력 향상
· 기대 이상의 어플리케이션 구현

원인 분석 :차이와 원인

· 주식 데이터 예측 시 종가(Close)만 사용하여 캔들 스틱 차트 활용 제한 발생
· 새로운 외부 라이브러리에 대한 숙지 부족으로 불완전한 GUI 구현
· 코드 파일 작성 후 즉시 테스트 진행이 되지 않아 최적화가 미흡함
· 로그인 및 회원가입 보안 관련 코드 이해도 부족으로 인한 불완전한 구축

피드백 종합 : 교훈

팀원 간 원활한 의견 소통의 중요성을 깨달았고, 사전 계획 수립과 작업 시간 스케줄링이 구축 순서의 중요도가 높다는 것을 인지하였습니다.
또 ,프로젝트 진행을 통해 적재적소에 필요한 테스트 진행이 되어야 완전한 어플리케이션을 만들 수 있다는 교훈을 얻게 되었습니다.

박 이 슬

목표 : 얻고자 한 것

· 협업을 통해 개인의 기술적 측면의 부족한 점 배우고 성장하는 것
· 프로젝트의 우선순위에 맞는 계획서 작성 및 계획 맞는 업무 실행하는 것
· python 언어에 대한 이해도를 향상시키고, 새로운 기술을 사용하는 것

결과 : 실제로 얻은 것

· 각자 맡은 영역의 부족한 부분은 팀원의 보완으로 작업 완성도를 높이며 업무를 마무리 할 수 있었던 점
· 새로운 외부라이브러리에 대해 분석하고, 사용법에 대한 기술을 습득한 점
· WBS 사용으로 업무 순서와 흐름 파악

원인 분석 :차이와 원인

· 어플리케이션 개발 순서에 대한 정확한 이해도 부족으로 인해 업무시간 지연
· TEST 케이스 실행 타이밍 Miss로 개발 완성도 미흡
· 새로운 라이브러리 정보가 많지 않아, 완벽히 숙지하지 못한 상태에서 기술을 사용함으로써 업무의 속도가 느려짐

피드백 종합 : 교훈

프로젝트를 시작 전 전체적인 업무 흐름과 개발 시 진행되어야 하는 전체 업무리스트를 나열하지 못한 채 WBS를 작성하게 되어 프로젝트가 끝나갈 즈음에 TEST를 진행하게 되어 불완전한 개발이 이루어진 것 같아 아쉬움이 있습니다. 그러나 각자의 맡은 영역에 대해 최선을 다하고 , 서로 보완하며 작업하는 모습으로 협업이 무엇인지 깨닫게 되는 시간이었습니다.

Project Evaluation_평가의견

김정빈

목표 : 얻고자 한 것

· 개인 프로젝트 진행 시 보다 나은 업무순서 스케줄링을 통한 프로젝트 진행
· 협업 프로젝트를 통해 팀원과의 커뮤니케이션 스킬 향상

결과 : 실제로 얻은 것

· 선 진행 된 업무 우선순위 작업을 통해 프로젝트 순서에 대한 이해도 향상
· 새로운 라이브러리사용을 통한 개인의 언어 능력 향상

원인 분석 :차이와 원인

· 새로운 라이브러리를 사용하기전 라이브러리 사용법 미숙지
· 프로젝트 진행 시 희망 포지션에 대한 적극적 의견 제시 미흡

피드백 종합 : 교훈

협업을 통해 팀원들에게 도움을 줄 수 있었던 점, 프로젝트를 잘 마무리 한 것에 대한 만족감을 느꼈습니다. 추후 , SMTP 프로토콜을 이용해 현재 만든 어플리케이션에 기능을 더해 주식 데이터를 사용자에게 보내주는 기능을 구현해 보고 싶습니다. 프로젝트를 통해 '개발자들은 항상 소통이 중요하다' 라는 말의 중요성을 느끼게 되었습니다

장으뜸

목표 : 얻고자 한 것

· Python GUI 라이브러리별 디자인 및 기능 구현
· 새로운 기능 Qtdesigner 기본 기능 숙련도 향상
· GUI 여러 화면 전환 기능 구현

결과 : 실제로 얻은 것

· Python Tkinter GUI 구현 기술 습득
· Qt designer의 기본적인 사용 방법 습득
· 일러스트레이터를 사용한 로고 디자인

원인 분석 :차이와 원인

· 새로운 라이브러리인 Qtdesigner에 대한 이해도에 대한 시간 소요
· 적극적인 의사 소통의 미흡
· 프로젝트 메인언어 및 라이브러리에 대한 숙지도 미흡

피드백 종합 : 교훈

팀 프로젝트인만큼 각자의 역할에 대한 중요성과 서로 간의 소통이 중요하다는 것을 깨닫게 되었습니다. 또한, 프로젝트 진행 중 새롭게 사용하게 된 기능들은 자기개발 시간을 통해 보완,발전 하는 시간을 계획하고 있습니다.