소프트웨어 공학 팀 프로젝트

팀원 : 고원준, 최문석, 송인용

## 개요

### 주제

* + 1. 데이터 분석 소프트웨어.

### 목적

* + 1. 기존의 코드를 이용하던 작업을 그래픽 환경을 통해 제공하여 간단한 조작을 통해 데이터 분석에 대한 결과를 얻고 파이프라인을 제작하여 데이터 분석에 대한 연계결과를 쉽게 얻을 수 있도록 한다.

### 기능

* + 1. 대용량 데이터를 무난히 처리할 수 있어야된다.
    2. 데이터에 대한 기본적인 통계 분석 정보 제공을 그래픽 환경으로 제공 되어야하며, 사용자가 분석을 원하는 데이터 항목을 선택할 수 있어야한다.
    3. 데이터에 대한 Pre-Processing 과정이 그래픽 환경으로 제공 되어야하며, 사용이 직관적이며, 엑셀과 같은 기능들이 제공 되어야한다.
    4. 그래픽 환경으로 마우스만 이용하여 쉽게 그래프(파이프라인)를 제작하여 코딩을 모르는 사람도 데이터를 분석하고 처리할 수 있어야한다.
    5. 소프트웨어는 웹 환경에서 이용이 가능해야한다.
    6. 로그인 기능이 제공되어 진행했던 작업을 저장하고 관리할 수 있어야한다.

### 비기능

* + 1. 장시간 이용 및 트래픽 폭주에도 서버가 안정적으로 유지 되어야한다.
    2. 데이터를 업로드하고 다른 작업을 할 수 있어야한다( 멀티 태스킹이 가능해야한다)
    3. 처리시작 버튼을 누른후 결과화면이 나오기 까지 3분이내로 걸려야한다.
    4. 한번에 10GB의 데이터를 업로드하고 분석할 수 있어야한다.
    5. 이미지 파일도 분석 할 수 있어야한다.
    6. 소프트웨어 이용방법이 직관적이어야한다.

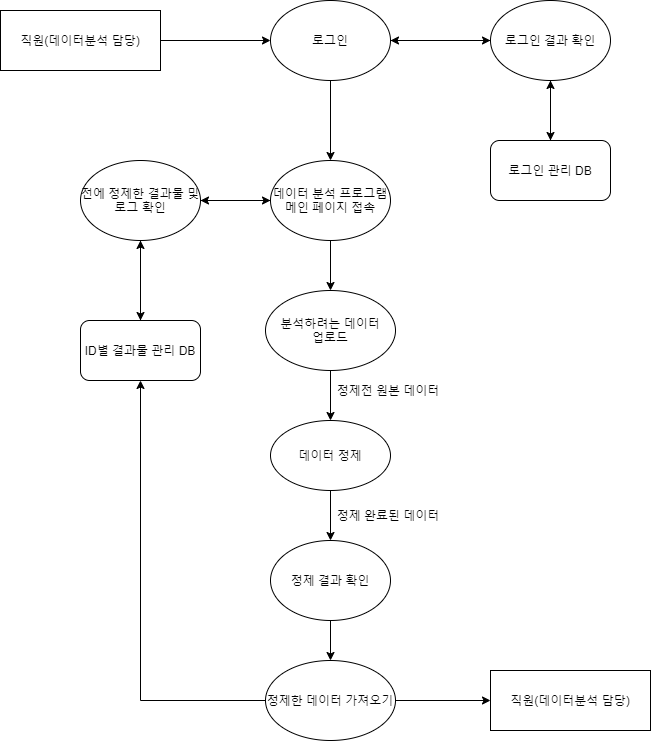
## 프로세스모델

### 애자일 프로세스모델

* + 1. 그래픽 사용환경과 특정유저에대한 이용조건을 고려해봤을때 클라이언트와 계속 소통을하면서 의견을 조율하고 반영할 필요가 있기때문에 적합하다고 생각하였다.
    2. 요구사항을 보면 프론트, 미들, 백에 모두 해당하는 요구사항이 있어, 해당 부분에 대한 분석및 설계를 모두 진행하여야하는데 다른 프로세스모델을 할 경우 상당히 오래 걸리기 때문에 애자일 프로세스 모델의 스크럼 개발론과 테스팅 주도 개발을 통해 빠른 개발을 할 필요가 있다.

## 요구사항 분석

### DFD



### SRS

#### 소프트웨어 명세서

#### 1. 소개

1.1 문서의 목적

데이터 분석 소프트웨어 개발 완료 이후에 분쟁사항 방지 및 법적 책임의 한계와 관련 계약사항을 정확히 명시하기 위하여 작성됨

1.2 문서의 범위

소프트웨어의 전반적인 품질 및 제공해야하는 기능과 제약사항들을 총칭함

1.3 용어의 정의

개발자: 소프트웨어를 개발한 팀과 회사를 칭함

사용자: 개발한 소프트웨어를 이용해 데이터 분석을 진행하려는 사람을 총칭함

분석: 데이터의 Pre-Processing부터 통계적 분석, 피벗 테이블, 조인등 사용자가 업로드한 데이터를 명령을 통해 정제하는 과정을 총칭함

#### 2 전반적인 명세서

소프트웨어는 로그인 기능을 통해 각 사용자를 구분할 수 있어야함

소프트웨어는 웹에서 동작할 수 있어야 하며, 파일 업로드를 통해 데이터 정제와 분석이 이루어질 수 있어야함

소프트웨어의 분석 과정은 GUI와 그래프를 통해 이루어질 수 있어야함

사용자는 프로그래밍 언어에 대한 이해도가 없어도, 소프트웨어를 통해 데이터를 원하는 대로 정제해서 분석할 수 있어야함

사용자는 지난 작업들을 개인별로 할당된 스토리지에 저장하여, 소프트웨어를 통해 불러오거나 분석중인 작업을 개인 스토리지에 저장할 수 있어야함

#### 3, 소프트웨어의 모든 세부적인 요구사항

**외부 인터페이스 요구사항**

-사용자 인터페이스: 사용자 인터페이스는 사용자가 코딩 대신 GUI를 통해 데이터 분석을 진행 할 수 있는 UI를 지칭함.

웹 기술을 통해 사용자가 장소나 기기에 상관없이 인터넷 연결을 통해 로그인하여 사용자 자신임을 인증할 수 있다면, 지금까지의 분석 작업이나 진행중인 작업을 불러올 수 있어야 하며, 데이터를 업로드하고 정제와 분석을 진행할 수 있어야함.

데이터의 입력은 사용자 인터페이스 업로드를 통해 이루어지며 출력 역시 동일한 인터페이스에서 이루어져야 함.

사용자는 데이터의 분석 과정을 확인할 수 있도록, 각 분석 단계별로 시행 결과를 GUI 환경에서 확인할 수 있어야 하며, 사용자가 원하는 경우 각 단계별 분석 과정중인 데이터도 다운로드 받을 수 있어야함.

업로드하는 데이터의 형식은 txt, xlsx, csv이 될 수 있으며 소프트웨어는 이 파일 형식들 모두를 다룰 수 있어야 하며, 분석 진행중 데이터의 손실이나 오류가 발생해서는 안됨.

사용자는 원하는 분석 단계를 그래프 형식으로 연결하여 진행할 수 있어야 하며, 사용자별로 할당된 스토리지 용량 한도 한에선 중간 분석 과정이나 완료된 결과를 저장할 수 있어야함.

소프트웨어가 보장하는 시간에는 사용자의 인터넷 속도에 영향을 받는 파일 업로드 시간과 다운로드 시간, 분석된 샘플을 페이지에 출력해주는 시간은 제외됨.

분석에 걸리는 시간은 사용자가 데이터를 업로드하고 분석을 시작한다는 명령이 서버에 전달된 순간부터 사용자의 명령대로 분석 완료된 데이터를 다시 사용자 인터페이스에 출력하기 직전 까지를 말하며, 100MB당 0.5초 미만 속도가 보장되어야 함.

**기능 요구사항:**

소프트웨어는 입력된 데이터의 형식이 정확한지 인식할 수 있어야 하며, csv나 txt 형식의 경우 각 컬럼을 구분하는 구분자를 사용자가 확인할 수 있어야함.

각 컬럼의 데이터 타입을 확인할 수 있어야 하며, NAN값과 NULL값을 처리할 수 있어야함.

각 분석 단계는 그래프의 흐름에 따라 이루어져야 하며, 그래프로 연결되어 있다면, 전단계의 분석 결과가 달라지면, 이후에 이루어지는 단계들은 값이 초기화 되며, 사용자는 초기화 된 사실을 알 수 있어야함.

분석 작업에서 하위 단계의 변화는 상위 단계에 영향을 끼쳐서는 안됨.

사용자와 개발자 모두 확인할 수 있도록 오류 로그가 기록되어야 하며, 만일 소프트웨어의 오류로 인해 데이터의 손실이 발생할 경우 개발자가 책임 져야함.

소프트웨어가 보장하는 데이터의 안정성은 데이터가 업로드 된 순간부터 분석된 데이터를 다운로드하기 직전 까지만 한정되며, 업로드 과정이나 웹에서 사용자의 로컬 환경으로 다운로드중 사용자의 인터넷 환경 오류로 인해 발생한 데이터의 손실에 대해선 보장하지 않음.

소프트웨어는 자동적인 백업 기능을 제공해주며, 사용자는 자동 백업의 기능을 사용할지 선택할 수있음.

백업 기능은 개인 스토리지의 용량이 가득 찰 경우 지원되지 않으며, 스토리지 용량부족으로 인해 손실된 데이터에 대해선 개발자가 책임지지 않음.

사용자는 스토리지에 저장된 데이터를 다운로드하거나 삭제할 수 있어야함.

출력은 오직 업로드 된 데이터와 관련된 분석 과정만 한해서 이루어져야 하며, 동일한 사용자의 다른 작업이나 다른 사용자의 업로드 된 데이터의 영향을 받아서는 안됨.

**성능 요구사항:**

데이터는 최대 10GB까지 업로드 가능해야함.

서버와 트래픽 폭주로 데이터의 손실이 발생해선 안됨.

여러 사용자가 동시에 분석 작업을 진행할 수 있어야함.

데이터의 분석 작업 속도는 항상 100MB당 0.5초 미만을 보장해야함.

각 사용자별로 30GB 이상의 스토리지를 제공해줘야하며, 사용자의 요청에 따라 스토리지의 용량을 늘릴 수 있어야함.

**논리 데이터베이스 요구사항:**

로그인 기능을 담당하는 데이터베이스가 유지되어야함.

각 사용자별로 분석 과정을 저장할 수 있는 스토리지가 구현되어야함.

각 사용자별 스토리지는 다른 사용자의 스토리지에 영향을 끼칠 수 없도록

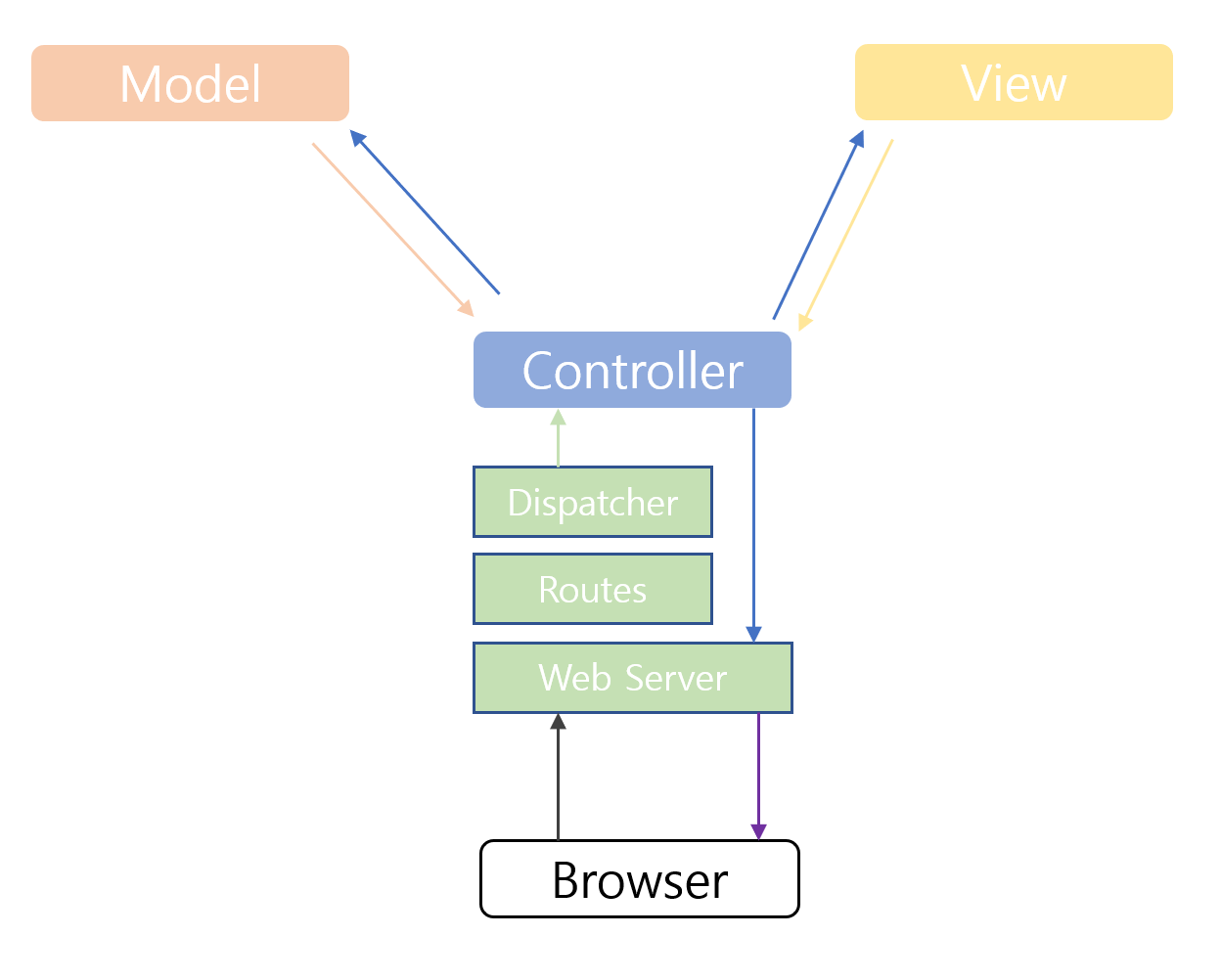
설계되어야함.

사용자에게 할당된 스토리지 크기는 필요에 따라 가변적으로 변할 수 있게

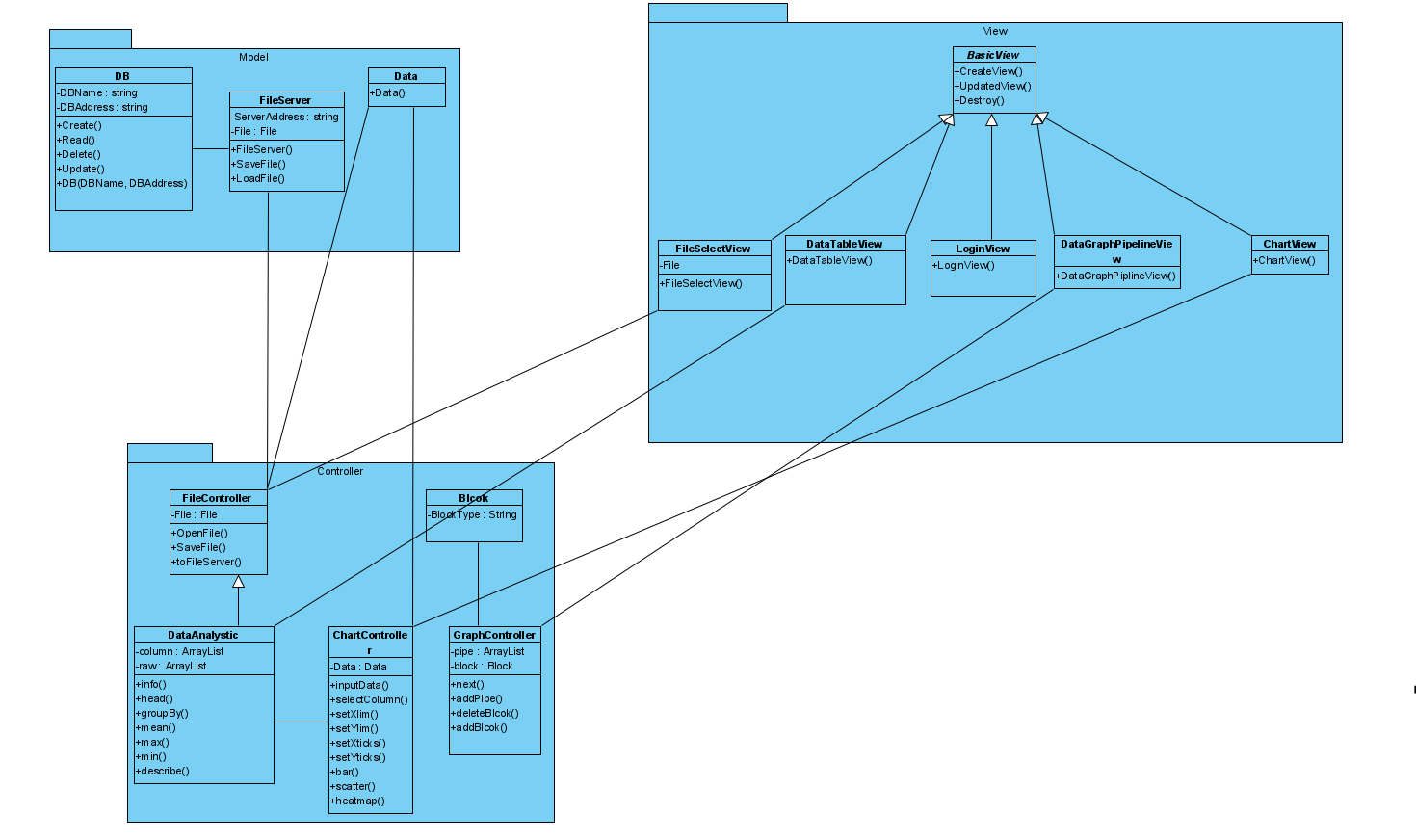
설계되어야함.

## 아키텍처 모델

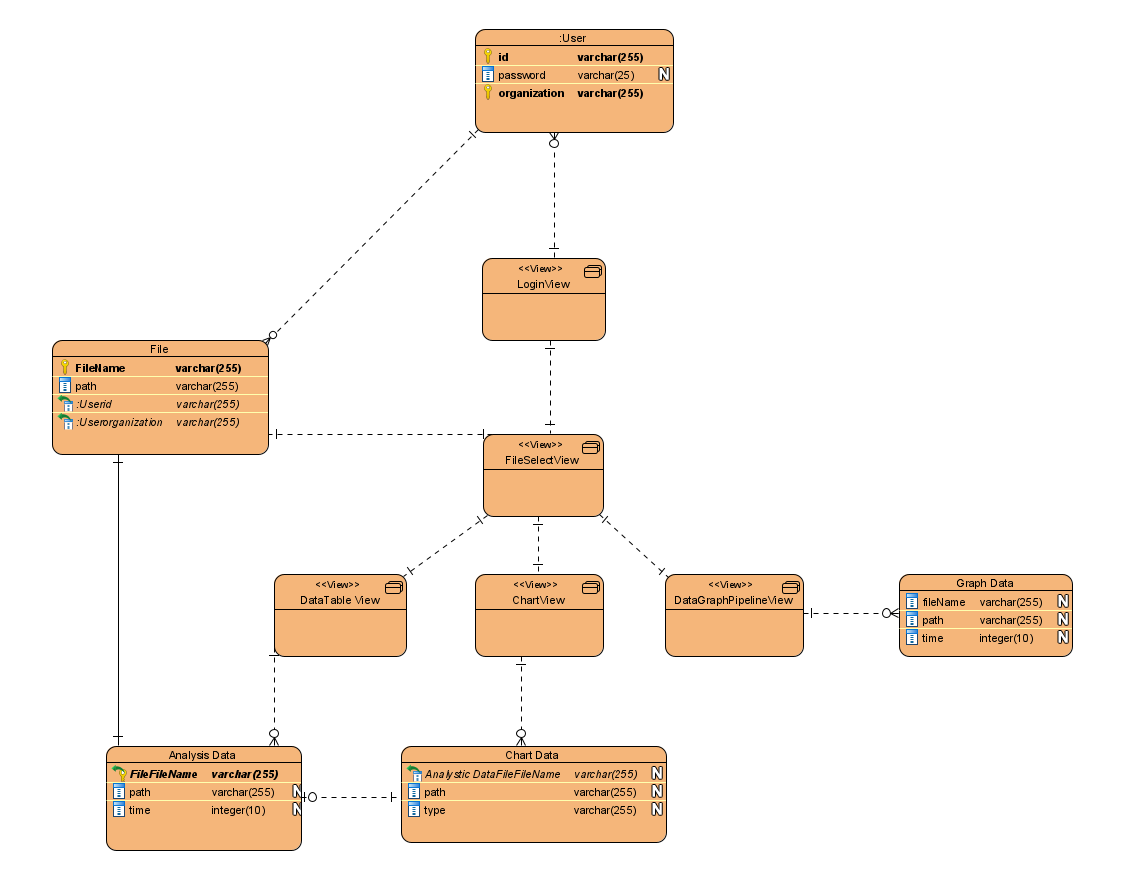
### MVC모델

* + 1. 도식화  
       

### 클래스 다이어그램

* + 1. 

### ERD

* + 1. 

## 디자인 패턴

### Observer 패턴

* + 1. 사용자에 대한 입력에 반응을 하기 위해 사용하는 패턴으로 주로 UI와 서버에 연동되는 비동기 처리를 및 백그라운드 작업을 위해 사용될 예정이다.

### Mediator 패턴

* + 1. 로그인 기능에 사용되는 패턴이며, 로그인 로직을 처리하기 위해 사용한다.

### Template Method패턴

* + 1. 주로 View를 구현하기 위해 사용되며 기본 뷰를 상속받아 하위 클래스에서 구체적인 명세를 하여 각 뷰에대한 의존도를 줄이기 위해서 사용한다.

### Memento 패턴

* + 1. 진행중인 작업의 상태를 저장하여 사용자가 종료후 다시 로그인 했을때 이전에 작업하던 작업을 이어서 할 수 있도록 제작하기 위해 사용한다.

### Front Controller 패턴

* + 1. 각 요청에 대한 사전 처리를 위한 컨트롤러를 만들어 사용하기 위해 적용한다.

### Composite View 패턴

* + 1. 상위 View로 부터 파생한 작은 컴포넌트들로 구분하여 코드 재사용성을 증진 시키기 위한 목적으로 사용한다.

### MVC 패턴

* + 1. 그래픽적인 작업 요구 환경이 많고, 서버와 데이터베이스에 대한 컨트롤이 많이 요구 되기 때문에 각 부분을 독립하여 개발할 필요가 있었다,
    2. 사용자에 대한 ReActive한 프로그래밍이 요구되었기 때문에 Obsever패턴이 적용가능한 MVC아키텍처를 선택하게 되었다.

## System Testing과 User Testing 시 주 점검 항목과 선택 이유.

### System Testing :

* + 1. 기능 1. 대용량 데이터를 무난히 처리할 수 있어야 한다.
    2. 비기능 1. 장시간 이용 및 트래픽 폭주에도 서버가 안정적으로 유지 되어야한다.
    3. 비기능 2. 데이터를 업로드하고 다른 작업을 할 수 있어야한다(멀티 태스킹이 가능해야 한다)
    4. 이유 : 실제 빅데이터 분석에 관한 사이트 등을 이용하였을 때 대용량 데이터를 처리하지 못하여 여러 파일로 분할하여 진행하였으며, 서버가 불안정하여 팅김현상이 여러 번 발생하였으고 그에 대한 불만이 많았으며, 본 소프트웨어 또한 같은 오류가 발생할 가능성이 높기 때문에 우선적으로 확인하여야 한다.

### User Testing :

* + 1. 기능 4. 그래픽환경으로 마우스만 이용하여 쉽게 그래프(파이프라인)를 제작하여 코딩을 모르는 사람도 데이터를 분석하고 처리할 수 있어야한다.
    2. 비기능 6. 소프트웨어 이용방법이 직관적이어야한다.
    3. 이유 : 개발자(숙련자)의 입장에서는 쉽게 이해할 수 있고, 사용에 지장이 없으나 특정 사용자(입문자)의 입장에서는 다르게 느껴질 수 있는 부분이 있기 때문에 실제 사용자의 피드백을 받아 요구사항대로 진행되었는가를 확인하여야 한다.

## 요구사항 분석-설계-구현-테스트-운영 및 유지보수 단계에서의 필수 문서와 선택 이유.

### 요구사항 추적표.

### 이유 :

* + 1. 식별된 각 요구사항을 기준으로 하여 이후 개발될 산출물들과의 추적 관계 정의 및 고객 요구사항이 주요 단계별 작업산출물에 적절히 반영되었는 가를 확인하기에 유용하다.
    2. 요구사항이 모두 반영되었는지의 여부를 양방향으로 검증할 수 있다.
    3. 고객 요구사항 > 소프트웨어 요구사항 > 구조 설계 > 상세 설계 > 구현 > 테스트 순으로 요구사항의 추적이 이루어지기 때문에 각 단계별로 요구사항에 적합한 산출물인가를 파악할 수 있다.
    4. 해당 산출물에서 변경이 발생한 경우 추적 관계 갱신이 편하다.
    5. > MVC 모델을 사용하기 때문에 Model, View, Controller에 맞춰 요구사항이 모두 적용되었는가를 파악할 수 있으며, 웹 환경에서 구동되어야 하기에 어려운 유지보수에 적합하며, Controller가 클라이언트 겸 서버의 역할을 하기 때문에 양방향 검증이 필요하다.