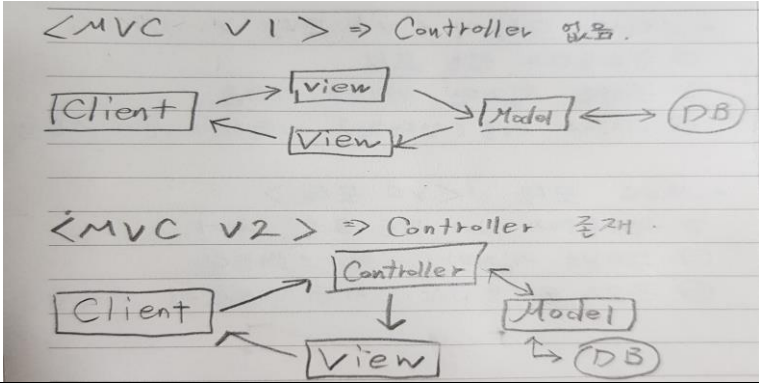
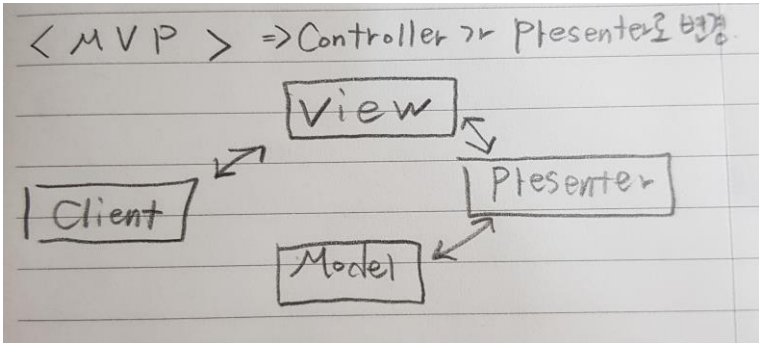


MVC,MVP,MVVM 아키텍처 패턴에 대해서 조사한 후 아래 순서에 따라 각각 정리하여 제출

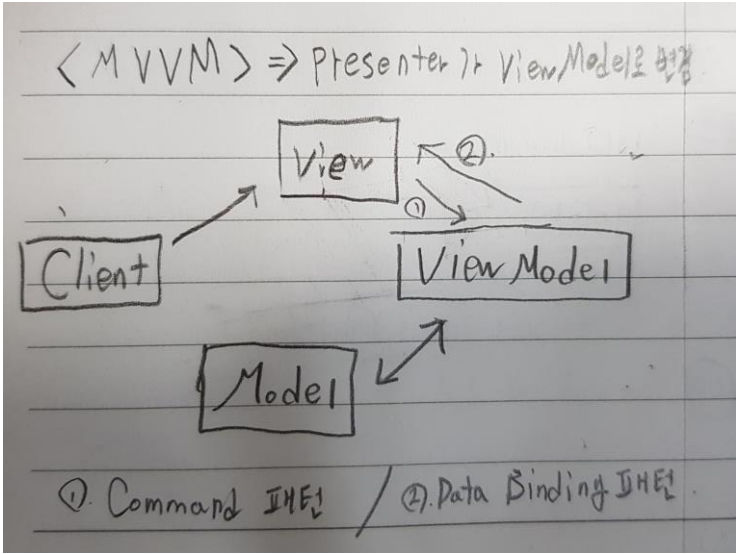
1. 정의
2. 패턴의 구조
3. 패턴의 구조에 기반한 동작 설명
4. 특징
5. 장점 & 단점

MVC	
1.정의	Model-View-Controller 의 약자로, 개발시 3가지의 형태로 역할을 나누어 개발하는 소프트웨어 개발 방법론으로 컨트롤러가 있는 v1과 컨트롤러가 없는 v2로 나뉜다. 비즈니스 로직과 표현(데이터)을 분리해서 개발함으로써 서로 영향없이 개발하기 수월하다는 장점이 있다.
2.패턴의 구조	<p>구조는 크게 3가지로 나뉜다. Model, View, Controller</p>  <p><MVC v1> ⇒ Controller 없음.</p> <p><MVC v2> ⇒ Controller 존재.</p> <ol style="list-style-type: none"> 1. v1 : 컨트롤러가 없는 모델로, 구현은 쉽지만 유지보수가 어렵다.따라서 작은(간단한) 어플리케이션에서 사용된다. 2. V2 : 컨트롤러가 있어 모든 요소를 제어하며, 유지보수가 쉽지만 구현이 어렵고 오래걸린다. 따라서 규모가 큰(복잡한) 어플리케이션에 사용된다.
3.패턴의 구조에 기반한 동작 설명	<ol style="list-style-type: none"> 1. Model(무엇을) : 자료의 비즈니스 로직(멤버 메소드,함수,제어문등)에 관한 처리를 한다. 또한 데이터의 상태변화가 생기면 컨트롤러와 뷰에 통보하며 데이터베이스와 연결된다. 데이터 호출하여 전송 및 저장기능을 담당 -> "자바빈즈" 로 구현 2. View(화면표현) : 사용자가 보는 화면으로(사용자 인터페이스 요소), 표현부분(자바스크립트,css3등 표현부분)을 나타내며 모델로부터 정보를 얻어온다. -> "JSP,HTML" 로 구현한다. 3. Controller(어떻게) : Model(데이터)과 View(화면요소)를 연결하는 역할을 하며 모델에 명령을 보내 모델의 상태를 변경한다. -> "자바서블릿,JSP" 로 구현
4.특징	<ol style="list-style-type: none"> 1.Model 에서의 특징 : <ul style="list-style-type: none"> -사용자가 편집하길 원하는 모든 데이터를 가지고 있어야 한다. -뷰나 컨트롤러에 대해서 어떤 정보도 알지 말아야 한다. -변경이 일어나면, 변경 통지에 대한 처리 방법을 구현해야만 한다. 2.View 에서의 특징 : <ul style="list-style-type: none"> -모델이 가지고 있는 정보를 따로 저장해서는 안된다. -모델이나 컨트롤러와 같이 다른 구성요소들을 몰라야 된다. -변경이 일어나면 변경통지에 대한 처리방법을 구현해야만 한다.

	<p>3.Controller 에서의 특징 :</p> <ul style="list-style-type: none"> -모델이나 뷰에 대해서 알고있어야 한다. -모델이나 뷰의 변경을 모니터링 해야 한다. <p>따라서 각각 맡은바에만 집중을 할 수 있게 된다. 디자이너는 디자인에, 프로그래머는 비즈니스 로직에 집중할 수 있게 된다.</p>
5.장점 & 단점	<p>1. 장점 :</p> <ul style="list-style-type: none"> -개발의 효율성 증가 -웹 응용프로그램 수정, 확장, 유지보수가 쉽다. <p>2. 단점 :</p> <ul style="list-style-type: none"> -모델과 뷰 사이의 의존성이 높아 커질수록 복잡해지고 유지보수가 어렵다. -설계시간이 오래걸리고 숙련된 개발자가 필요 -모델과 뷰의 완벽한 분리가 어렵다.

MVP	
1.정의	Model-View-Presenter 의 약자로, MVC에서 컨트롤을 Presenter로 바꾼것이다. Presenter는 뷰에서 요청한 정보를 Model로부터 가공해서 View로 전달한다. 따라서 뷰와 모델이 Presenter를 통해서만 동작할수 있으므로, 뷰와 모델의 의존성을 제거한 패턴이다.
2.패턴의 구조	<p>구조는 크게 3가지로 나뉜다. Model, View, Presenter</p> 
3.패턴의 구조에 기반한 동작 설명	<p>1. Model(무엇을) : 자료의 비즈니스 로직(멤버 메소드,함수,제어문등)에 관한 처리를 한다. Presenter과 연결되어 동작하며 뷰와는 연결되지 않는다.</p> <p>2. View(화면표현) : 사용자가 보는 화면으로(사용자 인터페이스 요소), 표현부분(자바스크립트,css3등 표현부분)을 나타낸다.</p> <p>3. presenter : Model(데이터)과 View(화면요소)를 연결하는 역할을 하며 뷰와 의존성이 더 높다.</p>
4.특징	<p>1.Model 에서의 특징 :</p> <ul style="list-style-type: none"> - 비즈니스 로직을 독립적으로 테스트 할 수 있다. <p>2.View 에서의 특징 :</p> <ul style="list-style-type: none"> -presenter 에서 호출할 수 있도록 인터페이스를 가지고 있고, 인터페이스로 presenter와 통

	<p>신한다.</p> <ul style="list-style-type: none"> -presenter과 1:1 관계로 통신한다. <p>3.Presenter 에서의 특징 :</p> <ul style="list-style-type: none"> - Controller와 같지만 뷰와 연결되지 않고 인터페이스에 연결되었다는 점이 다르다. -뷰와 통신하기 위해 인터페이스를 가지고 있고 모델과 직접 연결된다. <p>따라서 뷰와 모델간의 의존성을 제거했다. MVC가 가진 테스트 가능성 문제, 모듈화, 유연성 문제를 해결했다.</p>
5.장점 & 단점	<p>1. 장점 :</p> <ul style="list-style-type: none"> -뷰와 모델간 의존성이 없다. - <p>2. 단점 :</p> <ul style="list-style-type: none"> -뷰와 presenter간의 의존성이 높다.

MVVM	
1.정의	<p>Model-View-ViewModel 의 약자로 MVP에서 Presenter 를 ViewModel로 바꾼것이다. 마이크로소프트의 WPF와 실버라이트 개발에서 고안된 패턴이다. 바인딩 추상층 객체가 추가되어 뷰를 표현하기 위해 만들어진 뷰를 위한 모델이다. 두가지 디자인 패턴 command과 Data Binding을 사용한다. 이 패턴으로 인해 뷰와 뷰모델은 의존성이 완전히 사라진다.</p>
2.패턴의 구조	<p>구조는 크게 3가지로 나뉜다. Model, View, viewmodel</p> 
3.패턴의 구조에 기반한 동작 설명	<p>1. Model(무엇을) : 자료의 비즈니스 로직(멤버 메소드,함수,제어문등)에 관한 처리를 한다.</p> <p>2. View(화면표현) : 입력이 들어오면 Command패턴으로 뷰모델에 명령을 한다. 또한 뷰모델과 Data Binding으로 인해 자동으로 값이 갱신된다.</p> <p>3. ViewModel : 필요한 데이터를 모델에 요청하고 응답받은 데이터를 가공해서 저장한다. 뷰</p>

	를 추상화한 객체로 모델에서 어떤 데이터를 처리하고 뷰에서 어떠한 방식으로 보여줄지를 결정한다.
4.특징	<p>Command 패턴과 Data Binding 패턴을 이용하여 구현되었다. 추상층객체로 인해 뷰모델의 값이 뷰에 자동으로 갱신되어 둘 사이의 의존성을 없앴다. 이 패턴으로 인해 뷰와 뷰모델 사이의 의존성을 없애 뷰와 뷰의 관계가 n:1의 관계가 되었다.</p> <p>따라서 뷰의 속성과 뷰모델의 속성을 바인딩 시켜줌으로써 뷰모델의 속성이 변경될 때마다 뷰가 자동업데이트 된다.</p>
5.장점 & 단점	<p>1. 장점 :</p> <ul style="list-style-type: none"> -뷰와 모델 사이의 의존성이 없다. - Command 패턴과 Data Binding 패턴을 사용해 뷰와 뷰모델 사이의 의존성을 없앴다. -모듈화가 가능하다. -관리에 용이해져 재사용성이 좋다. <p>2. 단점 :</p> <ul style="list-style-type: none"> -뷰모델의 설계가 어렵다. -플랫폼 제한적인 요소가 존재. -뷰에대한 처리가 복잡해질수록 뷰모델이 거대해져 오버스펙이 될 수있다.