

1024

① 작성 일시	@2023년 10월 24일 오전 9:36
☰ 태그	

프로젝트 수행 결과 : 작업한 화면의 기능들이 구현 되는 것을 영상으로 보여 주는 것

유스케이스 다이어그램 : 작업 처리 순서

요구사항 정의서(중요) : 기능 밑 구형

요구사항 정의서

요구사항 ID	요구사항명	기능 ID	기능명	상세 설명	필수 데이터	선택 데이터
MEM01	회원가입	MEM01_LOGIN01	마스터 회원가입	마스터 기능. 마스터 회원 가입	아이디,비밀번호,이름,성별,생년월일,이메일,회사명	
		MEM01_LOGIN02	학생 회원가입	마스터 기능. 학생 회원 가입	아이디,비밀번호,이름,생년월일,이메일,닉네임,학년,학교,학원 소속 번호	
		MEM01_LOGIN03	직원 회원가입	마스터 기능. 직원 회원 가입	아이디,비밀번호,이름,생년월일,이메일,부서명,직급	
		MEM01_LOGIN04	선생 회원가입	마스터 기능. 선생 회원 가입	아이디,비밀번호,이름,생년월일,이메일,학교,학원 소속 번호,연락처,담당 번호,경력	
MEM02	회원 정보 수정	MEM02_UPDATE01	회원 정보 수정	회원 정보를 변경	Access Token, 비밀번호	
		MEM02_UPDATE02	회원 정보 수정	회원 정보를 변경	Access Token, 편 번호	
MEM03	토큰 재발급	MEM03_LOGIN01	회원로그인	로그인 후 토큰 발급 및 refreshToken 리로드로 로그인 유지	아이디(비밀번호), 토큰 재발급	
		MEM03_TOKEN01	액세스 토큰 재발급	리로드로 토큰을 재발급	리로드로 토큰	
MEM04	아이디 찾기	MEM04_FINDID01	이름 및 생일 아이디 찾기	입력한 이름, 생일, 이메일과 일치하는 회원이 있을 경우 해당하는 회원의 아이디를 알려드립니다.	이름, 생일, 이메일	
		MEM04_FINDPWD01	아이디 찾기	입력한 이름, 아이디, 이메일과 일치하는 회원이 있을 경우 해당하는 회원의 비밀번호를 생성하고 해당 비밀번호를 메일로 수신해주시면 해당 메일로 비밀번호를 발송합니다.	아이디, 이름, 이메일	
MEM05	회원 탈퇴	MEM05_DELETE	회원 탈퇴	회원 탈퇴	Access Token, 회원 번호	
		MEM06_PRINT01	관리자로 제외된 모든 회원 정보 조회	관리자로 제외된 모든 회원 정보 조회	Access Token(관리자)	page, keyword
MEM06	모든 회원 조회	MEM06_PRINT02	모든 학생 조회	모든 학생 조회	Access Token(관리자)	page, keyword
		CLAA01	CLAA01_CLASS01	반 소속 학생 조회	반 소속 학생 조회	액세스 토큰,페이지
TE001	동일 학생 탐지	TE001_NOTICE01	반 고유 번호 대중	입력한 시험과 시험자 나온 학생들을 다른반으로 정렬하는 시험자 나온 학생들을 조회	액세스 토큰, 시험번호	
		FE001_FEEDBACK01	피드백 조회	로그인 한 사람의 피드백 목록 조회	Access Token	
FEE01	선생님 피드백	FE001_FEEDBACK02	피드백 등록 조회	로그인 한 선생님이 등록한 피드백 상세 조회	Access Token, 글 번호	
		FE001_FEEDBACK03	피드백 작성	로그인 한 선생님이 등록한 작성	Access Token, 학원 번호	
		FE001_FEEDBACK04	피드백 수정	로그인 한 선생님이 등록한 수정	Access Token, 글 번호	
		FE001_FEEDBACK05	피드백 삭제	로그인 한 선생님이 등록한 삭제	Access Token, 글 번호	
FE001	피드백 등록	FE001_FEEDBACK06	댓글 등록	개시글 작성 및 등록한 댓글 내용을 글에 게시글 등록 및 댓글내용 등록	Access Token, 글 번호, 댓글 내용	
		FE001_FEEDBACK07	댓글 등록	개시글 등록 및 댓글내용 등록	Access Token, 글 번호, 댓글 내용	
STA001	등록 및 조회	STA001_ST01	학생 전체 연령별 성적 조회	연령별(ex:2023-03) 기준 시, 해당 연령별에는 전년 학기의 시험의 성적과 과목별로 조사하는 기준	Access Token, 연령	
		STA001_OT01	반별 평균점 조회	연령(ex:2023-03) 기준 시, 해당 연령별에는 전년 학기의 시험의 성적과 과목별로 평균 점수를 조회하는 기준	Access Token, 연령	
		STA001_OT02	점수 높은 순 혹은 학생 조회	과목별 반별(연령(ex:2023-03) 기준 시, 해당 연령별에는 전년 학기의 시험의 성적과 과목별로 평균 점수를 조회하는 기준)으로 학생들을 정렬하는 기준	Access Token, 과목별로 평균 점수, 학생 번호	
GRA01	성적 입력	GRA001_INPUT01	성적 입력	교과목으로 학생별로 신생생별로 시험별 평점을 입력받아 성적을 정렬합니다.	Access Token, 과목별로, 학생별로, 신생생별로, 시험별로, 점수	
		GRA001_SHOW01	이번 달 성적 조회	이번 달 성적을 관리하고 조회	Access Token(학생)	
GRA01	등록 및 조회	GRA001_SHOW02	학생 별 등록 성적 조회	등록 및 성적 조회	Access Token(학생)	
		GRA001_SHOW03	학생 별 등록 성적 조회 /성적 등록 메세지/학과/과목 조회	등록 및 성적 조회	Access Token(학생)	
GRA01	학과별 학생 등록 성적 조회 (생생님, 지원 관원)	GRA001_SHOW04	해당 학생 이번달 성적 조회	선생님, 직원 권한으로 해당 담당 반별 학생 번호 및 학생별 성적 조회하기	Access Token(학원, 신생생님), 학생 번호와 함께 조회하기	
		GRA001_SHOW04	해당 학생 별 담당 학생 성적 조회 /성적 등록 메세지/학과/과목 조회	선생님, 직원 권한으로 해당 담당 반별 학생 번호 및 학생별 성적 조회,	Access Token(학원, 신생생님), 학생 번호와 함께 조회하기	

요구 사항 추적표 : 어디서 작동하고 어디와 연동되는지 표기

시험 계획서/시나리오 (중요) : 예외처리(버그) 버그 발생 시 어떻게 처리 할지 등.

WBS(*work breakdown structure*) : 작업 일정 계획서 (누가 했는지, 언제 했는지)

03. 프로젝트 절차 및 방법

	1 WEEK	2 WEEK	3 WEEK	4 WEEK	5 WEEK
프로젝트 기획	06 MAR - 08 MAR				
DB 구조화		08 MAR - 14 MAR			
프레임워크 설계		10 MAR - 16 MAR			
구현 기능 설계		14 MAR - 16 MAR			
기능 구현			15 MAR - 27 MAR		
중간점검				20 MAR	
기능 보완				22 MAR - 27 MAR	
더미 데이터 추가				23 MAR	
코드리뷰				27 FEB - 31 FEB	
오류 수정 및 보완				27 FEB - 31 FEB	
최종점검					31 MAR - 04 APR

TDD : 테스트 주도 개발 (솔로 프로젝트 or 대규모 인원 필요)



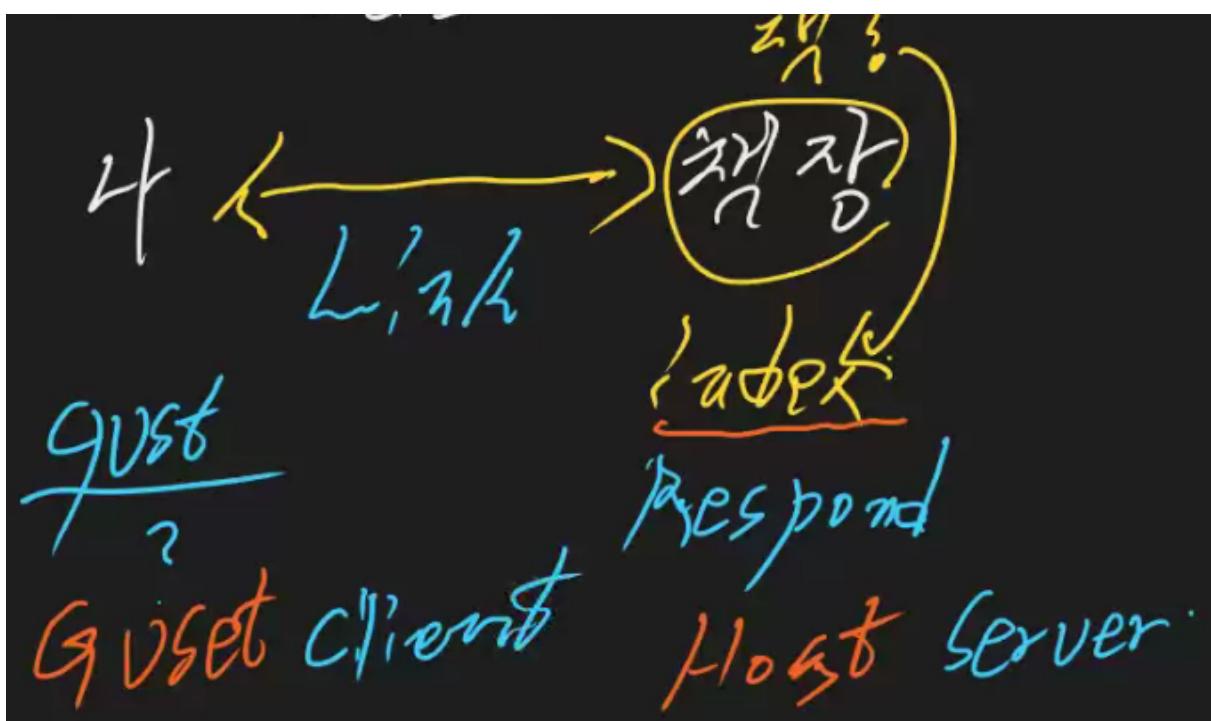
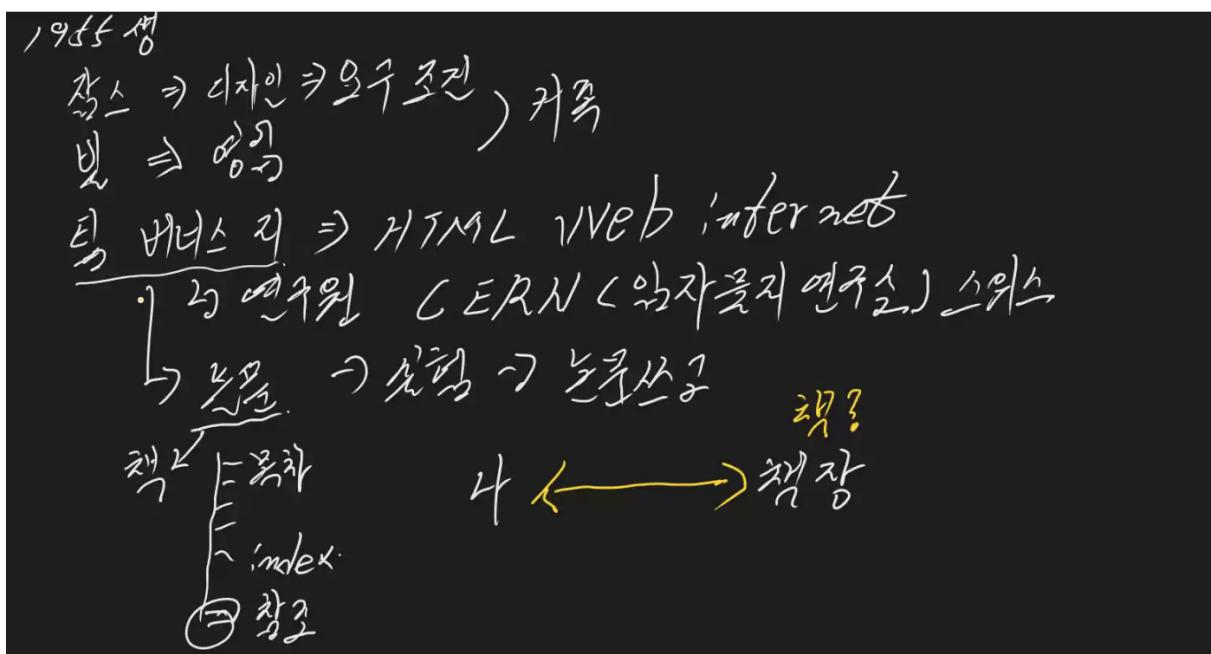
drow.oi 사용(백엔드)

1995c ip

잡스 ⇒ 디자인 ⇒ 요구 조건

빌게이츠 ⇒ 영업

팀버너스 리 ⇒ HTML WEB internet

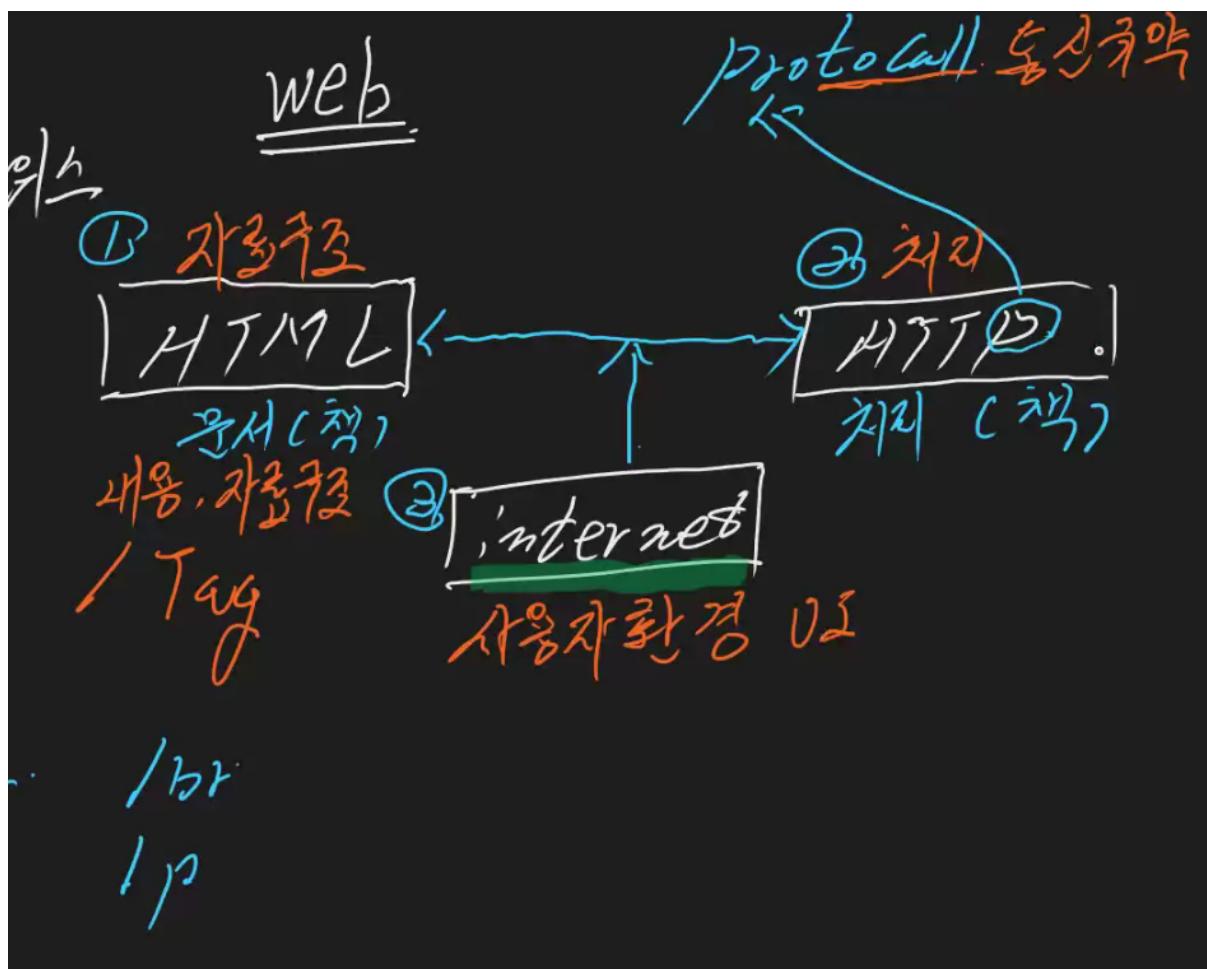


HTML : 내용, 자료구조 / Tag

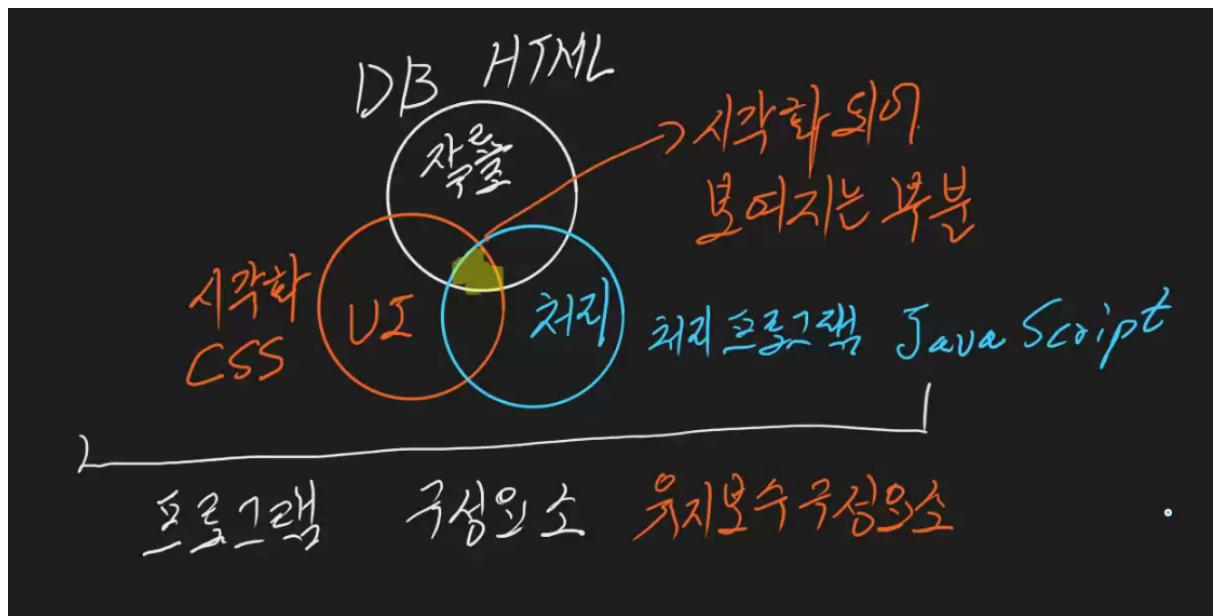
HTTP : P ⇒ Protocall (통신규약) : 처리

INTERNET : 사용자 환경 (UI) hello mr my yesterday

#프로그램 구성 요소



#위치에 따른 링크 구조



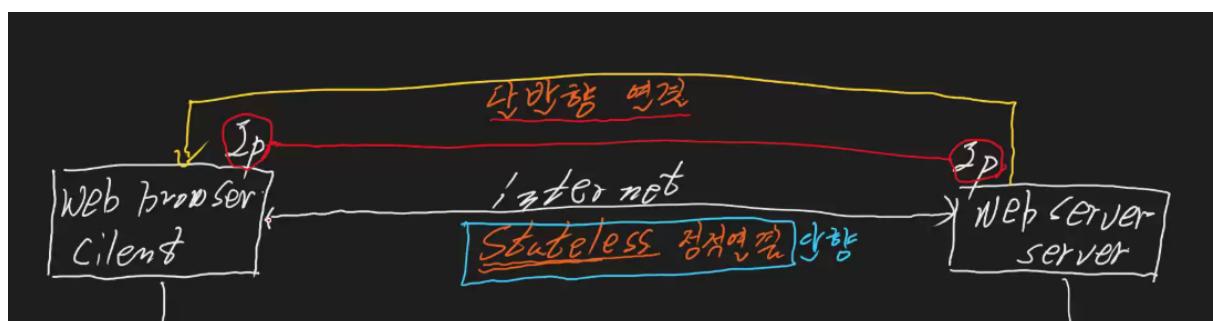
Web 이란

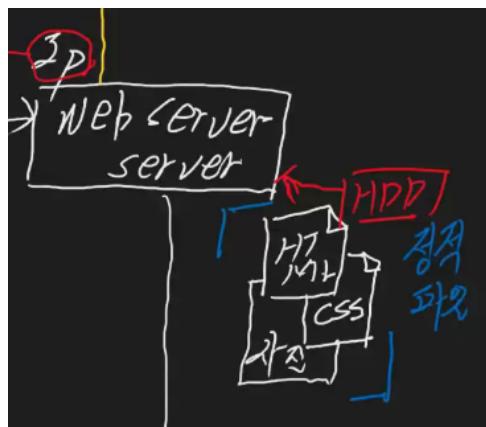
HTML + HTTP = 인터넷

HTTP

- 1.0 : 초기 인터넷 시대
- 1.1 : Web f/w
- 2.0 : IPv6 시대
- 3.0 : Meta

1.0 : 인터넷을 만들던 시대 (단방향)





대표적인 web server (port)

1. Apache 80
2. Tomcat 8080
3. NginX 8080
4. Node.Js 3600
5. front end

URL 검색 \leftrightarrow DNS

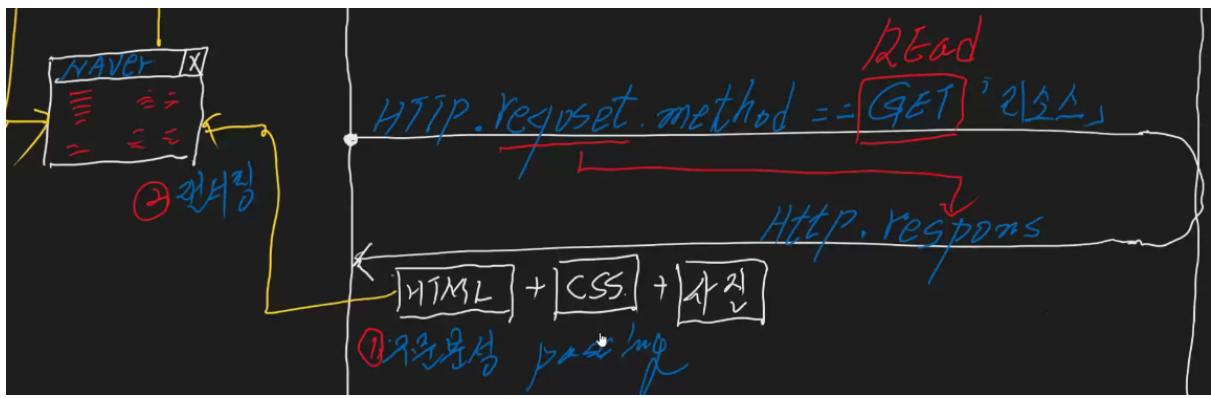
HTTP.request.method == get [리소스]

get=read

request \rightarrow http.response

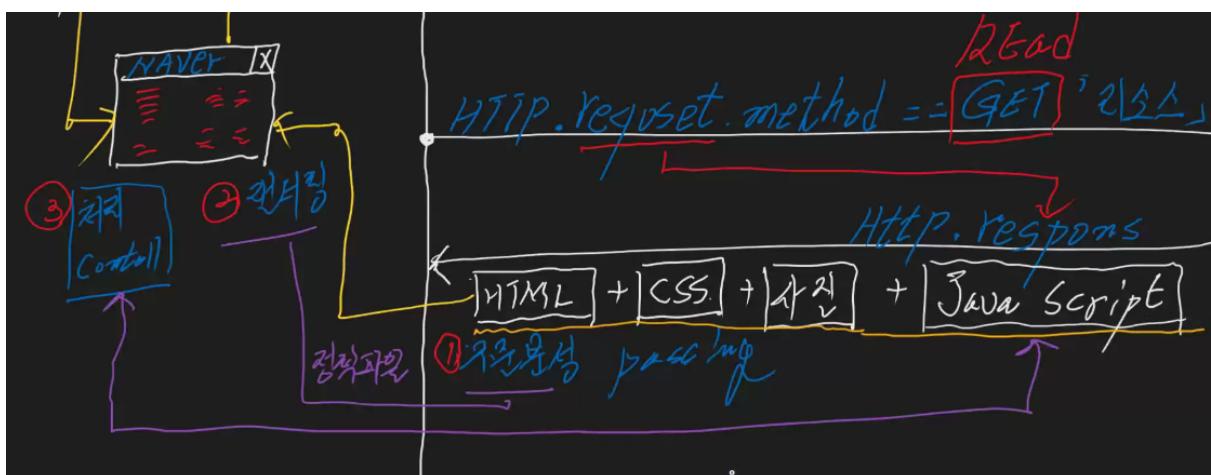
정적 파일

1. 구문분석 (parsing)
2. 랜더링

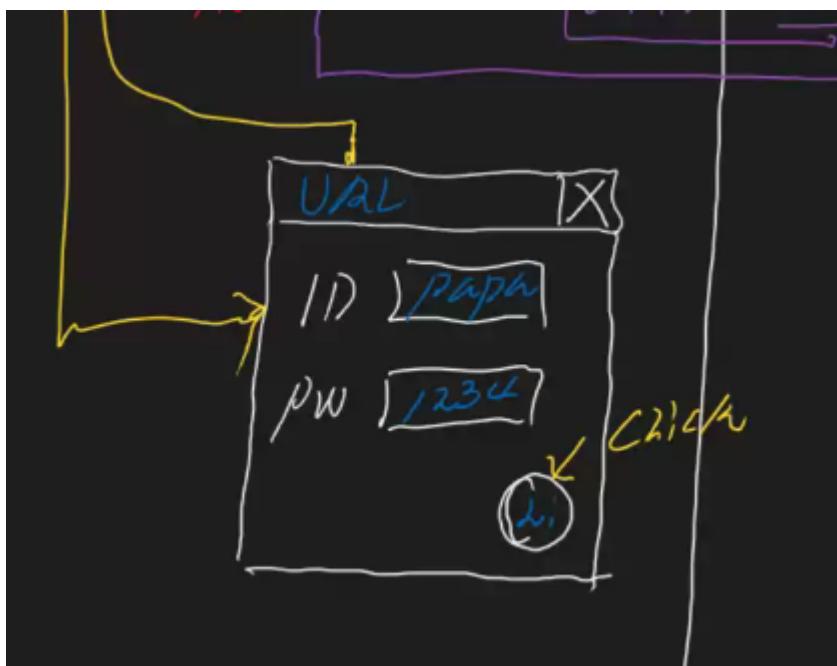
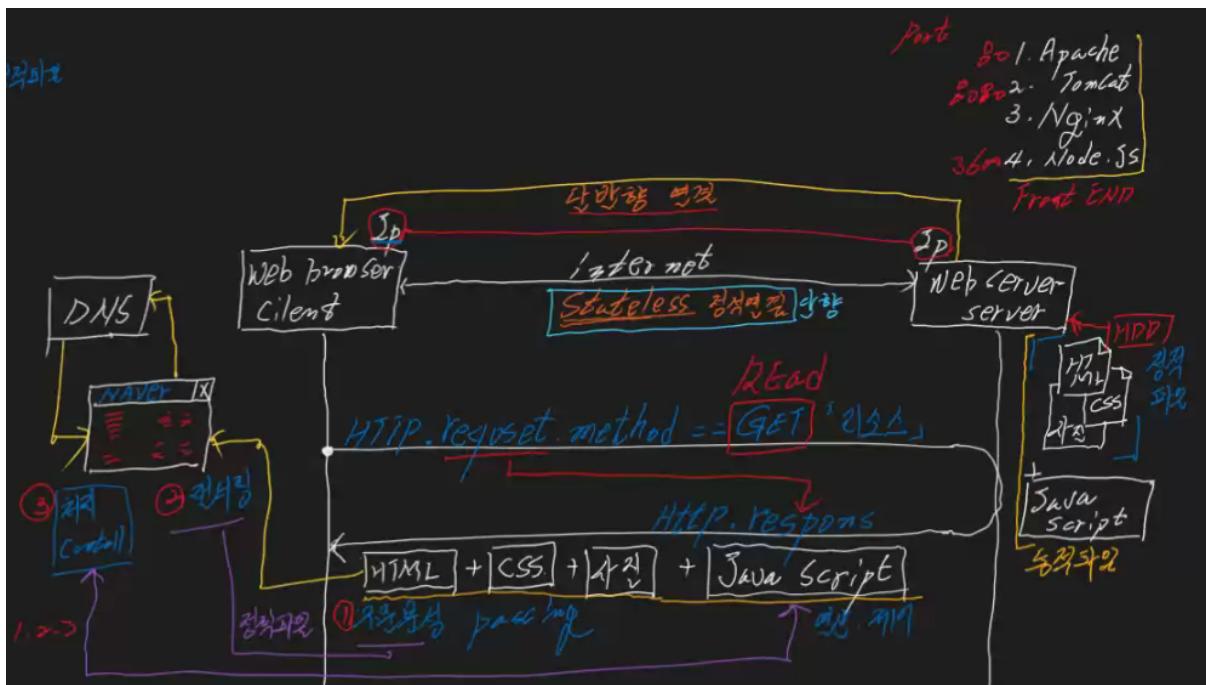


동적 파일 → 쌍방향 연결 / JSON 문법, 구문 등장

JS (연산, 제어) 추가 ↔ control



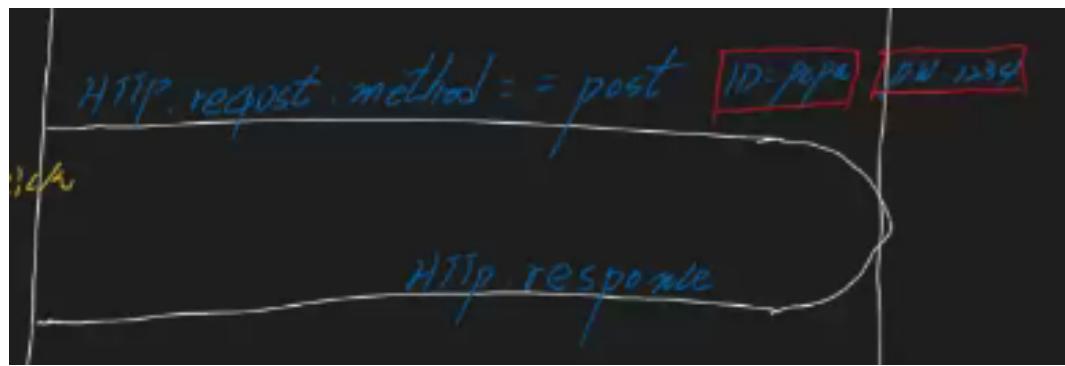
1.0의 웹 서버



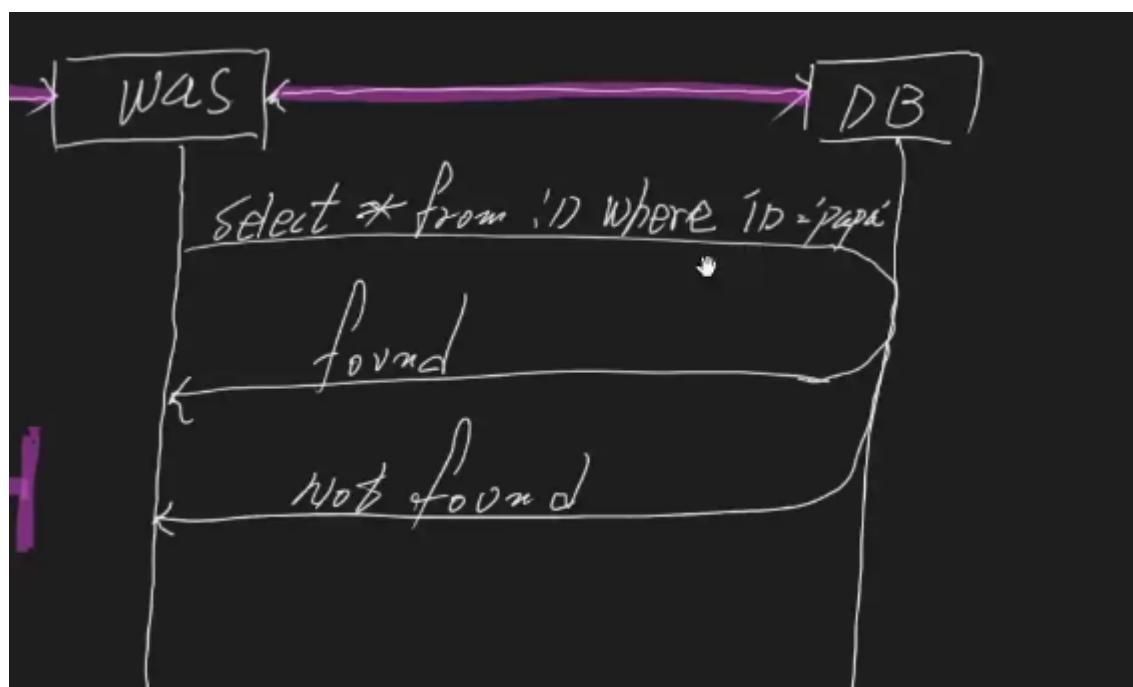
event 발생 (작동 시키는 무언가가 구동)

http.request.method==post ID=papa PW=1234

http.response



was : DB한테 데이터를 받아 계산 처리 응용



sever → was → select * from id,pw where id='papa', pw='1234' → DB

→ found or not found → was → server → request

was 대표 언어 : Jsp, pHp, Asp p-process

Jsp : os independent → JAVA 기준에서 작동 (JDBC가 연동 되어야 사용가능)

JDBC : JAVA 관리

[JDBC → spring → springboot] Back end Frame work

pHp(리눅스) : 소규모 웹사이트 제작에 좋음 (진입 장벽: 上, 대용량 처리에 부적합)

Asp : c, c++, c# (.NET) (애니메이션 사용)

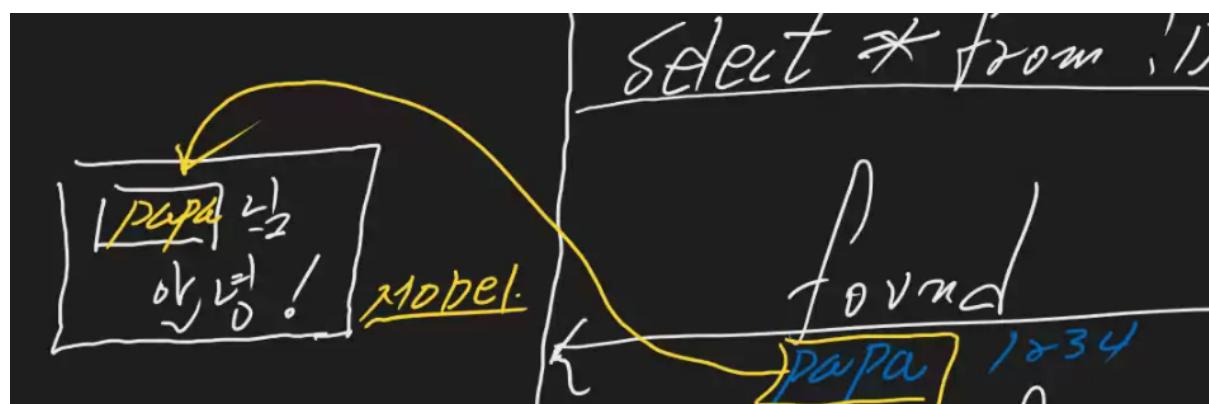
프론트 : web server 이전

백 : web server 이후

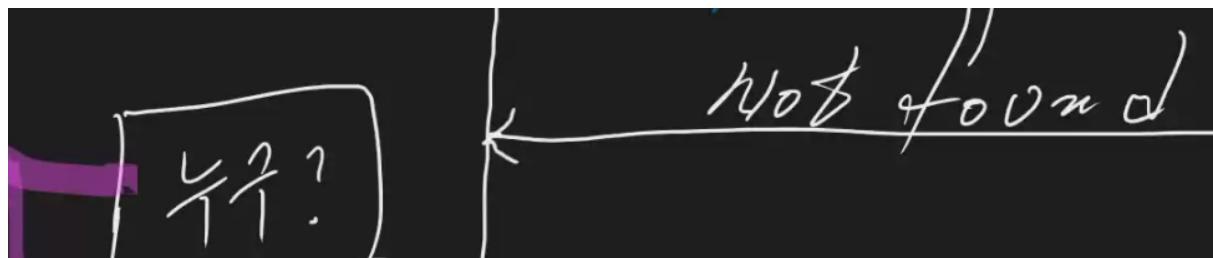
제일 많이 사용 : tomcat / spring / springboot



found

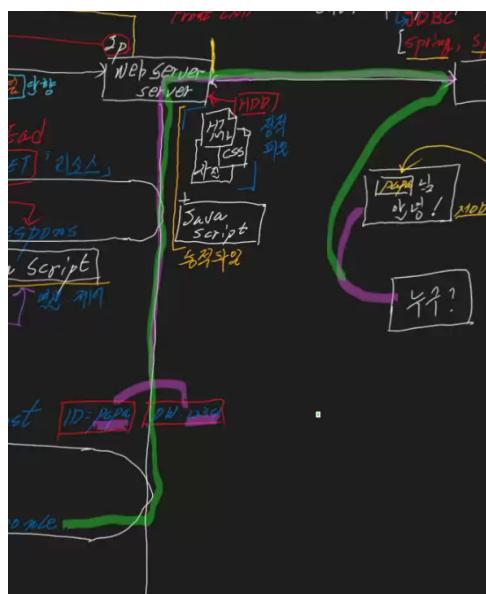


not found

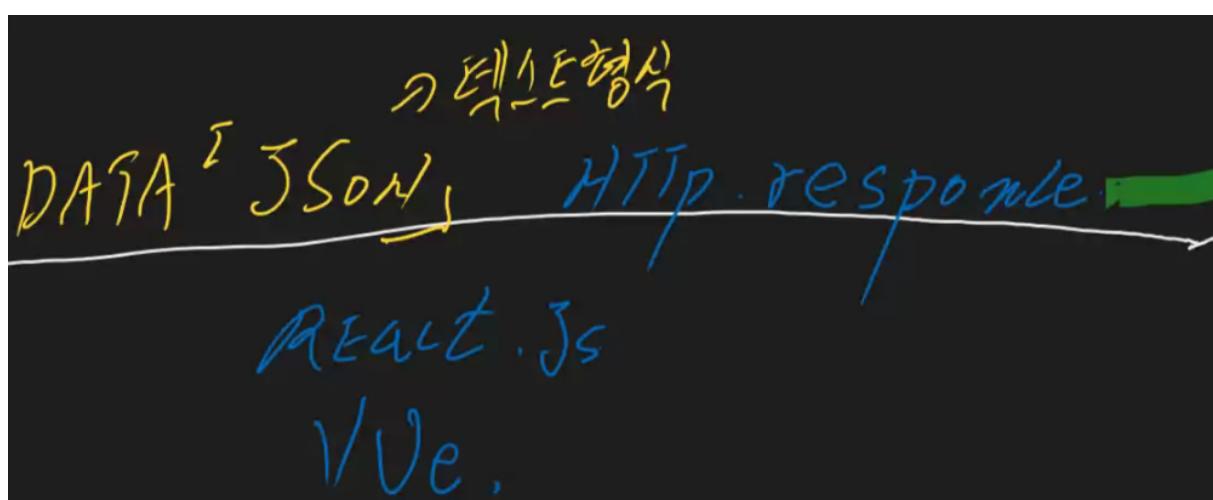


found의 경우

→ 정보 전달(mvc pattern.1)



DATA'Json'(텍스트 형식)로 변환 하여 전달

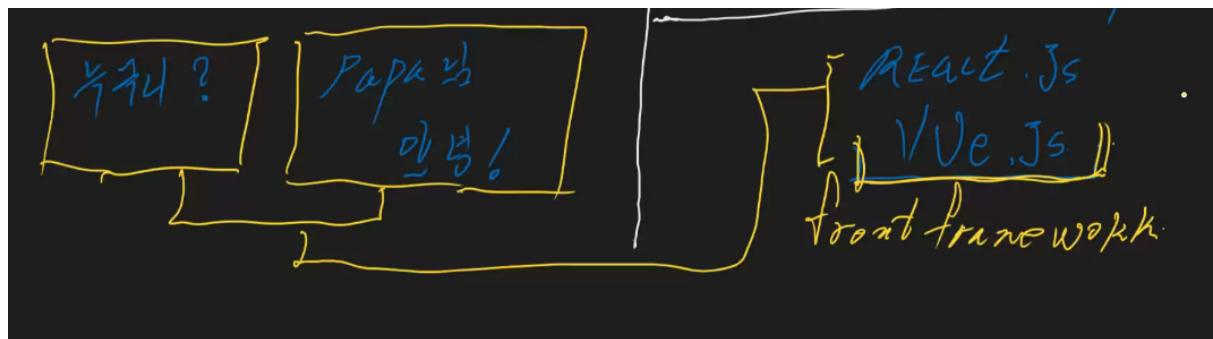


front frame work : react.JS / Vue.JS (data'Json'을 변환해서 사용)

Not found 의 경우

-예외 처리

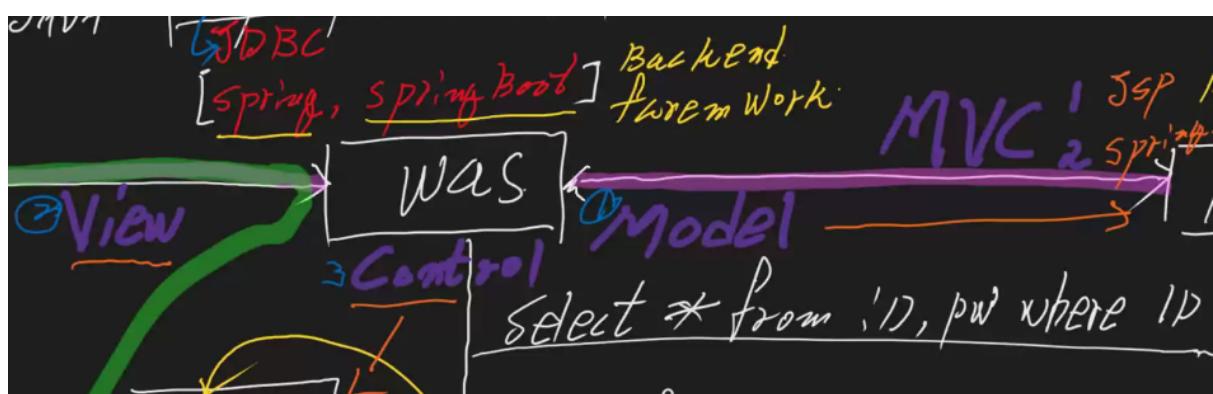
#결과값 출력 : react.js / Vue.js 가 랜더링



백

1. model : DB
2. view : 출력
3. control : 어떻게 나오는지

mvc1 = jsp mvc2 = spring



프론트

M → Json 파일, / rest api, restfull → 자료 요청, 처리 방법

V → Browser 구동 화면

C → Json으로 구현하는 과정

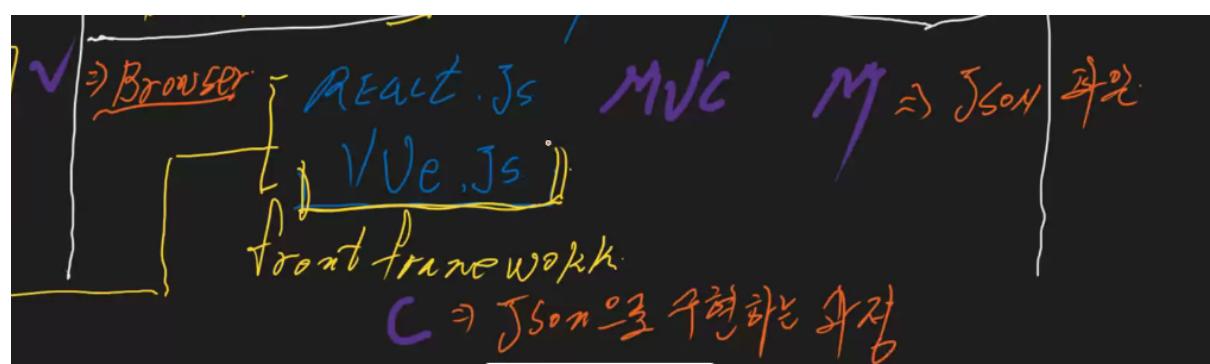
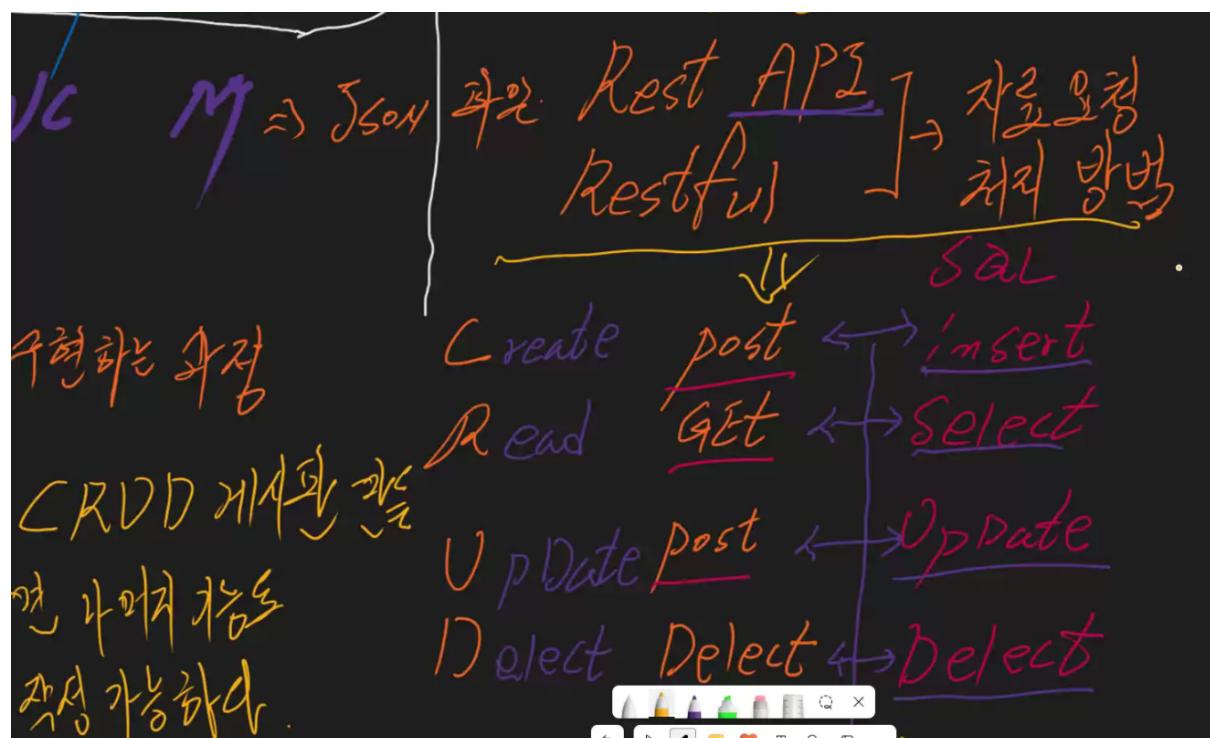
C : create \leftrightarrow post \leftrightarrow insert(SQL)

R : read \leftrightarrow get \leftrightarrow select(SQL)

U : update \leftrightarrow post \leftrightarrow update(SQL)

D : delete \leftrightarrow delete \leftrightarrow delete(SQL)

CRUD 게시판을 만들 줄 알면 나머지 기능도 작성이 가능하다



FRONTMVC - WEB - BACK MVC

MVC

MVC

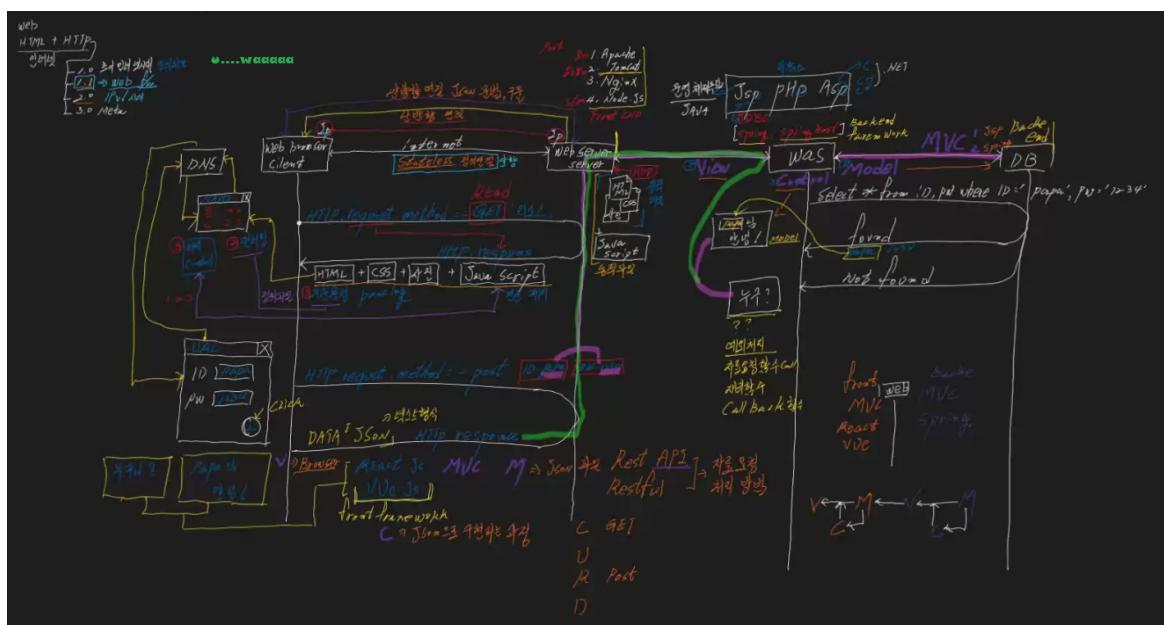
REACT

SPRING

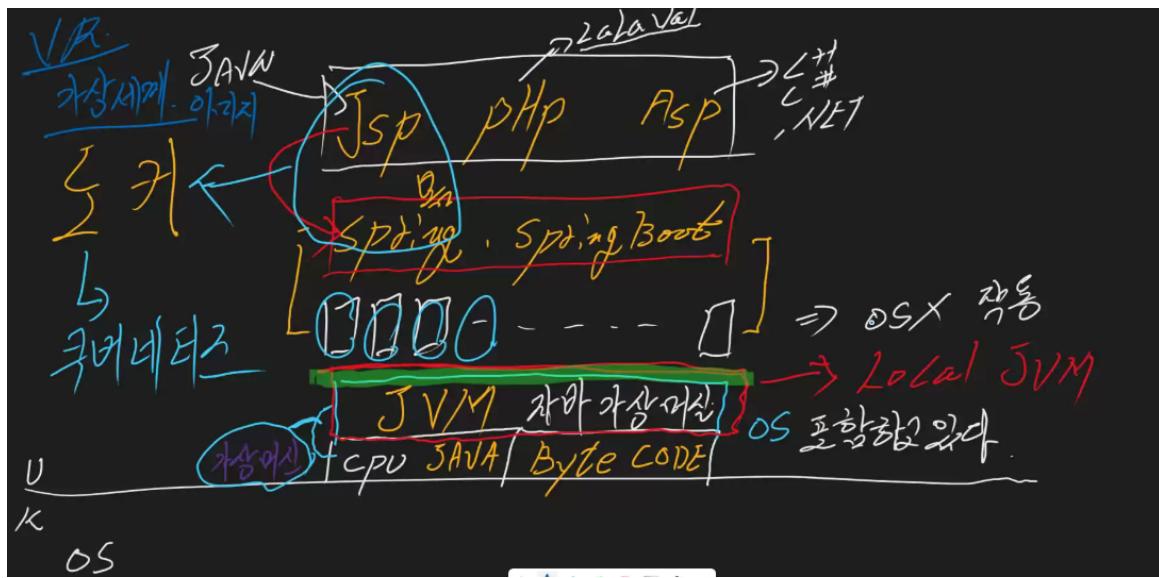
VUE



-성능 가속화할 방법 강구



가상머신 → JVM/cpu java/byte code (os를 포함)



- 어떤 환경이든 JVM 위에서 구동 할 것
- JVM 위에 구동되는 프로그램은 OS와 무관하게 작동한다
- 각각 작동하는 프로그램을 묶음 처리하기 위해 spring, spring boot가 만들어짐
- JSP PHP ASP → 단위 LALAVAL
- 프로그래밍 언어의 뿌리는 c 언어

도커

VR에 이미지로서 존재

web server

Apache → 정적 80

Tomcat → 동적 8080 (was server)

└ pHp

└ ASP

└ JSP

