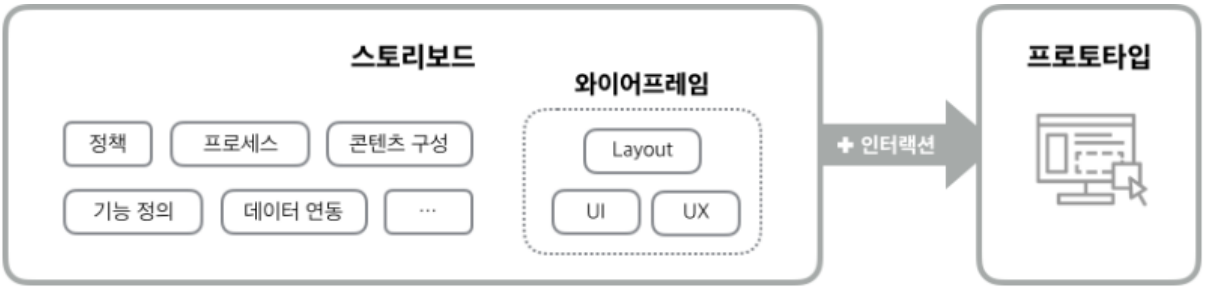
[09/27] 학습 커리큘럼, 스토리보드 및 와이어프레임, 압축, Git & Github, internet의 구조

- 커리큘럼 : 첫코딩(도서) > DB > React, Vue > AWS 구조화 > Javascript, Framework > Java, Spring

- 화면 설계서[1]



  - 스토리보드 : 기획, 정책, 프로세스, 컨텐츠구성, 기능정의, 데이터 연동
    - 기획 : 이후 요구사항정의서(ROC,Required Operational Capability)
    - 기능정의 : 지역별 -> 서울, 경기
    - 데이터 연동 : 자료가 화면에 표시
  - 와이어프레임 : 레이아웃, UI, UX
    - 각 세부페이지의 디자인 요소 결정
  - 프로토타입 : 인터랙션 이후 베타버전 출시, 과거 4:3 비율 CRT모니터 사용

- 이미지 손실, 무손실 압축에 관하여[2]
  - BMP(Bit Map) : 윈도우표준, 무손실압축
  - JPEG(Joint Photographic Coding Experts Group) : 국제표준, 손실압축, 적은용량(시간과 돈 절약, 빠름, 썸네일용)
  - GIF(Graphics Interchange Format) : 코닥, 손실압축
  - PNG(Portable Network Graphics) : 무손실압축, 배경투명처리 가능
  - AI : 일러스트레이트, 백터, 광고영상, 전용프로그램 필요(호환성x), 비쌈

- Git : 분산형 버전 관리 시스템
  - 협업 가능, 로컬과 분산 서버의 코드가 동일함
  - 순서 : 로컬머신 -> repository(저장소), add -> commit -> push
  - init : 동기화 시작
  - add : 깃에서 커밋하면 올라가능 영역 (스테이지 에어리어), readme(작업설명서)
  - commit : 깃헙에 추가(repository 로 올라감=동기화), -m[3] 설명추가 “text” 내용
  - branch : 하위,트리구조 -M
  - remote : 이전 내용은 내컴퓨터 저장, 리모트가 깃헙에 저장
  - push : 로컬 머신의 코드를 업로드하는 명령어, 최종 업데이트 완료
  - pull : 타인도 다운 가능
- Github : 형상관리버전틀, git 기반의 호스팅 사이트
  - Pull request : 관리자 이외의 유저가 코드 수정 요청
  - action : 코드 저장소에 어떤 이벤트가 일어날 때 특정 작업이 일어나게 하거나, 주기적으로 특정 작업을 반복 시켜 실행하는 경우에 사용

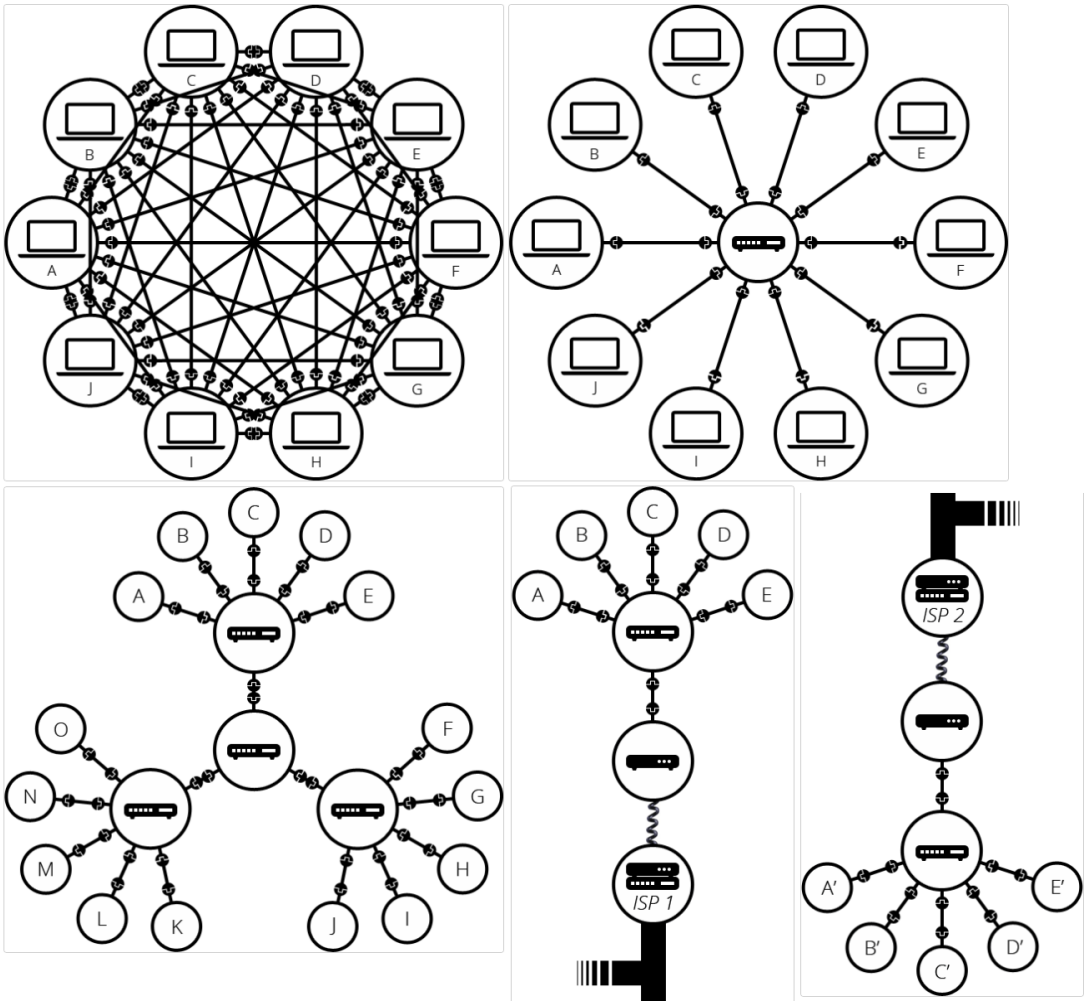
---

[1] [웹/앱 설계의 기본]화면설계서 작성방법

[2] Comparison of graphics file formats

[3] -m : massage / M : Merge

- internet[4] : 전 세계의 컴퓨터들이 정보를 주고 받는 네트워크들의 집합체



  - 1:N 대의 컴퓨터가 서로 연결되는 것 보다 Router를 통해 송수신
  - Router to Router 를 통해 네트워크가 확장
  - Moderm 을 통해 ISP(Internet Service Provider)로 인해 외부의 네트워크와 연결
  - 컴퓨터 찾기 : 모든 컴퓨터는 IP(인터넷 프로토콜) 라는 고유 주소가 있음,
    개인의 컴퓨터가 아닌 보다 큰 단위의 컴퓨터의 경우 도메인 주소(google.com) 사용

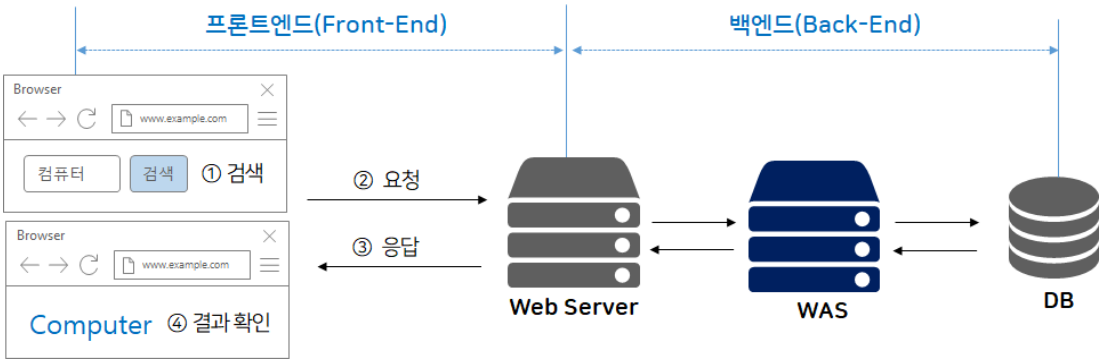
- HTTP(HyperText Transfer Protocol) : World Wide Web의 토대이며 하이퍼텍스트 링크를 사용하여 웹 페이지를 로드하는 데 사용, HTTP를 통한 일반적인 흐름에는 클라이언트 시스템에서 서버에 요청한 다음 서버에서 응답 메시지를 보내는 작업 포함
- DNS(Domain Name Server/System)[5] : 사람이 읽을 수 있는 도메인 이름(예: www.amazon.com)을 머신이 읽을 수 있는 IP 주소(예: 192.0.2.44)로 변환
- WS(Web socket) : 초기에는 클라이언트가 서버에 요청하는 핸드쉐이크 방식에서 이후 웹소켓(실시간 채팅, 주식등)으로 HTTP를 통한 양방향 방식으로 전환됨

---

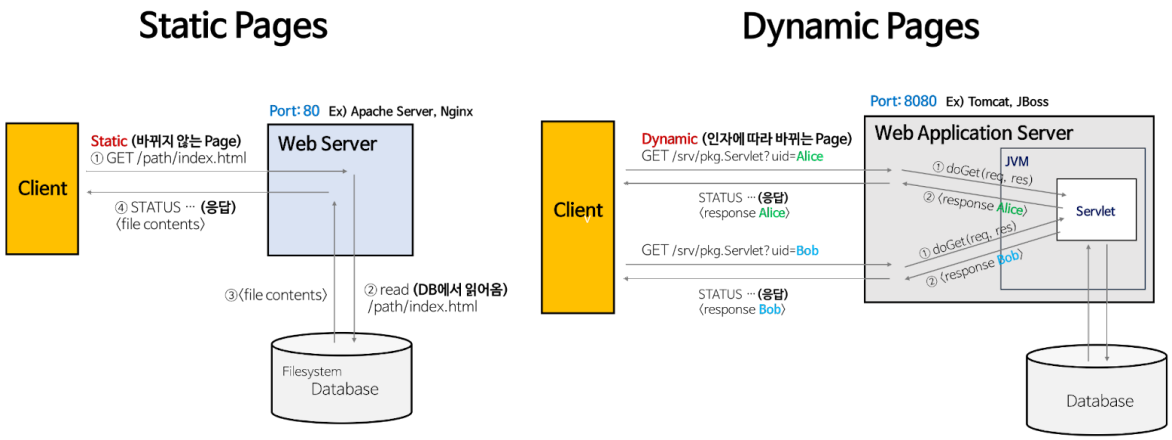
[4] 인터넷은 어떻게 동작하나

[5] DNS란 무엇입니까? # cloudflare 학습센터

- Client - Nic(Network Interface card, LAN[6] Card) - Router - DNS - Web Server - WAS - DB
    - Client <-> Web Server : React, Vue, HTML, CSS, Javascript
    - Web Server <-> WAS : Node.js, Spring, Spring Boot, Php
    - WAS <-> DB : MySQL, Maria, Oracle, MongGoDB



- Static Pages와 Dynamic Pages[7][8][9][10]



    - Static Pages
        - Web Server는 파일 경로 이름을 받아 경로와 일치하는 file contents를 반환한다.
        - 항상 동일한 페이지를 반환한다.
        - Ex) image, html, css, javascript 파일과 같이 컴퓨터에 저장되어 있는 파일들

    - Dynamic Pages
        - 인자의 내용에 맞게 동적인 contents를 반환한다.
        - 즉, 웹 서버에 의해서 실행되는 프로그램을 통해서 만들어진 결과물 * Servlet: WAS 위에서 돌아가는 Java Program
        - 개발자는 Servlet에 doGet()을 구현한다.

---

[6] LAN(local area network) : 근거리 통신망

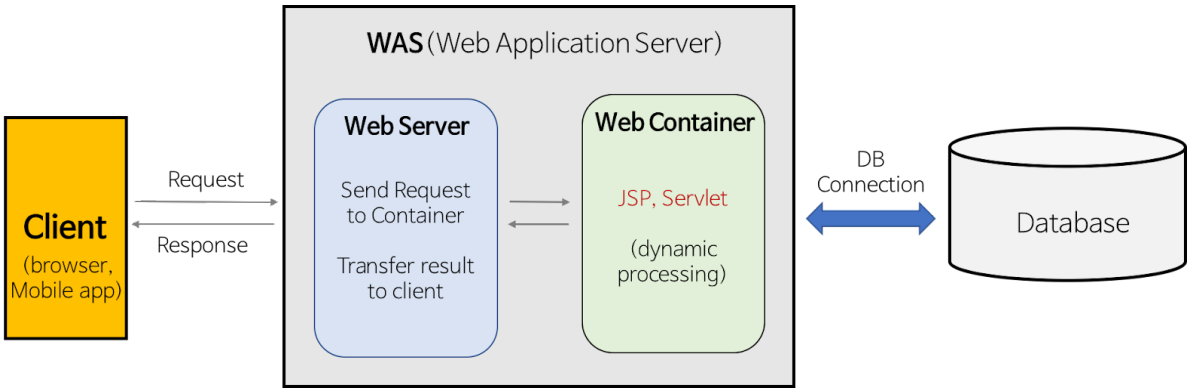
[7] Web Server와 WAS의 차이와 웹 서비스 구조

[8] kt cloud 보안의 이해 (심화) 01편 # KT cloud 온라인교육 Advanced 과정

[9] [AWS] Web Application 3Tier-Architecture # 교보정보통신 기술 블로그

[10] 3-Tier Architecture 정의 및 구성방식

- Web Server 와 WAS 차이

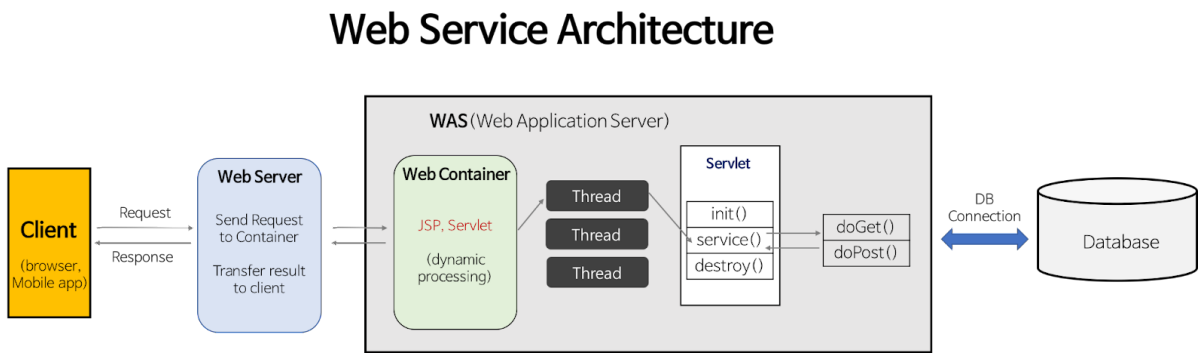


- Web Server
  - 개념 : 하드웨어와 소프트웨어로 구분
    - 하드웨어 : Web 서버가 설치되어 있는 컴퓨터
    - 소프트웨어 : 웹 브라우저 클라이언트로부터 HTTP 요청을 받아 정적인 컨텐츠(.html .jpeg .css 등)를 제공하는 컴퓨터 프로그램
  - 기능
    - HTTP 프로토콜을 기반으로 하여 클라이언트(웹 브라우저 또는 웹 크롤러)의 요청을 서비스 하는 기능을 담당한다.
    - 요청에 따라 아래의 두 가지 기능 중 적절하게 선택하여 수행한다.
      - 정적인 컨텐츠 제공, WAS를 거치지 않고 바로 자원을 제공한다.
      - 동적인 컨텐츠 제공을 위한 요청 전달, 클라이언트의 요청(Request)을 WAS에 보내고, WAS가 처리한 결과를 클라이언트에게 전달(응답, Response)한다.
  - 예시 : Apache Server, Nginx, IIS(Windows 전용) Web 서버 등

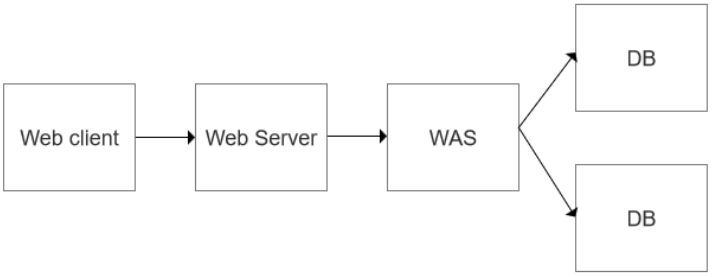
- WAS(Web Application Server)
  - 개념
    - DB 조회나 다양한 로직 처리를 요구하는 동적인 컨텐츠를 제공하기 위해 만들어진 Application Server
    - HTTP를 통해 컴퓨터나 장치에 애플리케이션을 수행해주는 미들웨어(소프트웨어 엔진)이다.
    - 웹 컨테이너(Web Container) 혹은 서블릿 컨테이너(Servlet Container)
      - Container란 JSP, Servlet을 실행시킬 수 있는 소프트웨어즉, WAS는 JSP, Servlet 구동 환경을 제공한다.
  - 역할
    - WAS = Web Server + Web Container
    - Web Server 기능들을 구조적으로 분리하여 처리하고자하는 목적
      - 분산 트랜잭션, 보안, 메시징, 쓰레드 처리 등의 기능을 처리하는 분산 환경에서 사용된다.
    - 주로 DB 서버와 같이 수행된다.
    - 현재는 WAS가 가지고 있는 Web Server도 정적인 컨텐츠를 처리하는 데 있어서 성능상 큰 차이가 없다.
  - 주요 기능
    - 프로그램 실행 환경과 DB 접속 기능 제공
    - 여러 개의 트랜잭션(논리적인 작업 단위) 관리 기능
    - 업무를 처리하는 비즈니스 로직 수행
  - 예시 : Tomcat, JBoss, Jeus, Web Sphere 등

- Web Server 와 WAS 의 구분
    - Web Server 필요성 : 정적 컨텐츠만 필요시 WS에서 처리하여 서버 부담 감소
        - 클라이언트(웹 브라우저)에 이미지 파일(정적 컨텐츠)을 보내는 과정을 생각해보자.
            - 이미지 파일과 같은 정적인 파일들은 웹 문서(HTML 문서)가 클라이언트로 보내질 때 함께 가는 것이 아니다.
            - 클라이언트는 HTML 문서를 먼저 받고 그에 맞게 필요한 이미지 파일들을 다시 서버로 요청하면 그때서야 이미지 파일을 받아온다.
            - Web Server를 통해 정적인 파일들을 Application Server까지 가지 않고 앞단에서 빠르게 보내줄 수 있다.
        - 따라서 Web Server에서는 정적 컨텐츠만 처리하도록 기능을 분배하여 서버의 부담을 줄일 수 있다.
    - WAS 필요성 : 정적,동적 컨텐츠를 구분하여 제공하여 자원의 효율적 사용
        - 웹 페이지는 정적 컨텐츠와 동적 컨텐츠가 모두 존재한다.
            - 사용자의 요청에 맞게 적절한 동적 컨텐츠를 만들어서 제공해야 한다.
            - 이때, Web Server만을 이용한다면 사용자가 원하는 요청에 대한 결과값을 모두 미리 만들어 놓고 서비스를 해야 한다.
            - 하지만 이렇게 수행하기에는 자원이 절대적으로 부족하다.
        - 따라서 WAS를 통해 요청에 맞는 데이터를 DB에서 가져와서 비즈니스 로직에 맞게 그때 그때 결과를 만들어서 제공함으로써 자원을 효율적으로 사용할 수 있다.
    - 구분시의 장점 : 자원 이용의 효율성 및 장애 극복, 배포 및 유지보수의 편의성 을 위해 Web Server와 WAS를 분리, Web Server를 WAS 앞에 두고 필요한 WAS들을 Web Server에 플러그인 형태로 설정하면 더욱 효율적인 분산 처리가 가능
        - 기능을 분리하여 서버 부하 방지
            - WAS는 DB 조회나 다양한 로직을 처리하느라 바쁘기 때문에 단순한 정적 컨텐츠는 Web Server에서 빠른 제공하는 것이 효율적
            - WAS는 기본적으로 동적 컨텐츠를 제공하기 위해 존재하는 서버
            - 정적은 Web Server 가 동적은 WAS가 처리시 빠른 수행가능, 페이지 노출 시간 또한 감소
        - 물리적으로 분리하여 보안 강화
            - SSL에 대한 암복호화 처리에 Web Server를 사용
        - 여러 대의 WAS를 연결 가능
            - Load Balancing을 위해서 Web Server를 사용
            - failover(장애 극복), failback 처리에 유리
            - 특히 대용량 웹 어플리케이션의 경우(여러 개의 서버) Web Server와 WAS를 분리하여 무중단 운영을 위한 장애 극복에 쉽게 대응
            - 앞 단의 Web Server에서 오류가 발생한 WAS를 이용하지 못하도록 한 후 WAS를 재시작함으로써 사용자는 오류를 느끼지 못하고 이용할 수 있다.
        - 여러 웹 어플리케이션 서비스 가능
            - 하나의 서버에서 PHP Application과 Java Application을 함께 사용하는 경우
        - 기타
            - 접근 허용 IP 관리, 2대 이상의 서버에서의 세션 관리 등도 Web Server에서 처리하면 효율적이다.

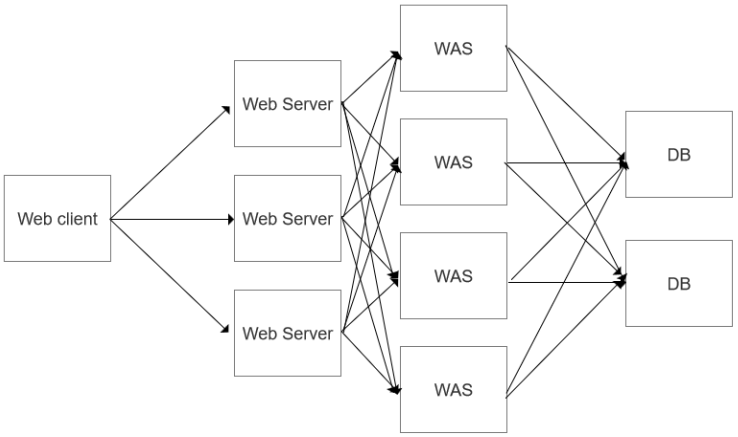
- Web Architecture
  - 단일 서버 : Web client -> Web Server/WAS/DB[11]
    - 빠른 서비스 구축, 저렴한 하드웨어
  - DB Server 분리 : Web client -> Web Server/WAS -> DB Server
    - DB 독점 사용으로 성능 향상, 웹과 DB간 네트워크 통신이 필요하므로 물리적 거리에 영향을 받기 때문에 지근 거리에 설정할 필요가 있음
  - Web Server 분리 : Web client -> Web Server -> WAS -> DB Server
    - Web Server에서 static resource 별도 처리, WAS 에서 Dynamic resource 처리함으로써 효율성 증가



  - DB Replication : Web client -> Web Server -> WAS -> DB Server(다중)
    - master/slave 혹은 Activity/Standby 로 사용



  - Load balancing : Web client -> Web Server(다중) -> WAS(다중) -> DB Server(다중)
    - 서비스 부하 분산 / Load balancing / 특정 서버 장애시에도 원활한 사용



[11] https://dotherealthing.tistory.com/19