

모듈 프로젝트 산출물

클라우드 기반 정보 수집 시스템

2022년 4월 27일

클라우드 보안 융합 (5조)

서상윤

오송희

유동섭

한승훈

목차

| | |
|----------------------------|-----------|
| 1. 프로젝트 개요 | 3 |
| 1-1. 클라우드 관련 기술 정리 | 3 |
| 1-2. 클라우드 관련 기술 현황 및 향후 전망 | 5 |
| 2. 프로젝트 기획 | 6 |
| 2-1. 프로젝트 구조 설명 | 6 |
| 2-2. 사용 툴 소개 | 6 |
| - Git | 6 |
| - Docker | 7 |
| - Trello | 8 |
| 3. 프로젝트 설계 | 9 |
| 3-1. 인터페이스 정의 | 9 |
| 3-2. 요구사항 정의 | 10 |
| 3-3. Database 설계 | 11 |
| 3-4. 간트 차트 | 12 |
| 4. 프로젝트 개발 | 13 |
| 4-1. 개발 및 운영 환경 세팅 | 13 |
| - nginx | 13 |
| - tomcat | 15 |
| - mysql | 16 |
| - python | 17 |
| 4-2. Python-CLI | 18 |
| - 로그인 | 18 |
| - 관리자 | 18 |
| - 일반 사용자 | 22 |
| 4-3. 웹 로그 수집 및 콘텐츠 저장 | 25 |
| - 웹 서버 운영 로그 수집 | 25 |
| - 웹 페이지 로그인 | 28 |
| - 웹 콘텐츠 저장 | 29 |
| 5. 프로젝트 산출물 | 31 |
| 5-1. 실행 화면 캡처 | 31 |
| 5-2. 프로젝트 결과에 따른 리소스 비교 | 31 |
| 5-3. 프로젝트 진행 소감 | 35 |

1. 프로젝트 개요

1-1. 클라우드 관련 기술 정리

지금까지 기업을 포함한 많은 사용자들이 시스템 및 데이터를 자체적으로 보유하고 관리해왔습니다. 하지만 이제는 필요한 만큼 사용할 수 있고, 물리적인 저장소가 필요하지 않은 클라우드 컴퓨팅이 각광받고 있습니다. **클라우드 컴퓨팅의 특징으로는 높은 경제성, 유연성, 가용성, 그리고 빠른 구축 속도가 있습니다.** 초기 투자 비용 및 운영 비용이 상대적으로 적고 필요 용량에 따라 유연하게 확장/축소가 가능합니다.

효과적인 클라우드 서비스를 제공하기 위해서는 다양한 기술이 필요합니다. 우선 **서버 가상화 기술**이 요구됩니다. 물리적 서버를 여러 개로 분할, 또는 병합하여 하나의 서버 환경으로 운용되어야 정상적인 클라우드 서비스가 가능해집니다. 또한 **서버 가상화는 서버의 남은 리소스를 효율적으로 관리할 수 있어서 공간 및 비용 절감에 큰 도움**이 되고 있습니다.

컨테이너는 가상화 기술 중 대표적인 기능으로 하나의 OS 환경에서 애플리케이션과 앱을 구동하는 환경이 격리된 공간을 의미합니다. 기존 가상화 기술로 많이 사용한 하이퍼바이저는 게스트 OS가 필요했던 반면 **컨테이너는 프로세스를 격리하여 모듈화된 프로그램 패키지를 제공할 수** 있습니다. **컨테이너는 프로그램의 구동 환경이 달라지면 각종 오류를 발생시키는 것을 해결하기** 위해 등장했습니다. 하나의 컨테이너에 실행에 필요한 다양한 리소스를 패키지로 묶어서 배포하면, 구동 환경이 달라져도 실행 파일과 리소스가 함께 따라다니기 때문에 오류를 최소화할 수 있어 현재도 많이 사용되고 있는 기술입니다.

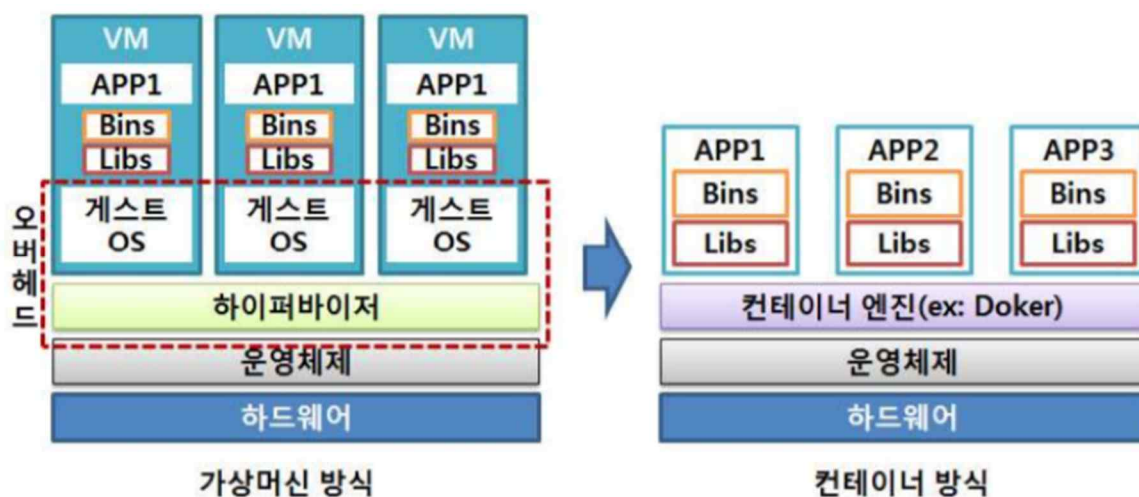


그림 1. 컨테이너의 개념

클라우드 서비스에 가장 많이 요구되는 또 다른 기술로는 분산 처리 기술이 있습니다. **분산 처리 기술이란 양이 많고 다양한 데이터를 여러 서버에 분산시켜 병렬 방식으로 빠르고 효율적이게 처리하는 기술**입니다. 특히 최근 빅데이터가 미래 산업으로 주목받고 있는 만큼 비즈니스에서 분산 처리의 필요성이 더욱 중요해지고 있습니다. 특히 분산 처리 기술은 기존의 슈퍼컴퓨터의 성능을 다수의 저성능 컴퓨터를 이용해 동일한 퍼포먼스를 보여줄 수 있어 **비용 절감 면에서 엄청난 장점**을 보입니다.

대표적인 서비스로 구글의 MapReduce가 있습니다. MapReduce는 일반적으로 많이 사용하는 Map 함수와 Reduce 함수를 기반으로 처리해야 할 데이터를 병렬화합니다. 데이터가 병렬화되면 클러스터의 각 노드로 보내져서 동시에 처리되기 때문에 높은 효율성을 자랑합니다.

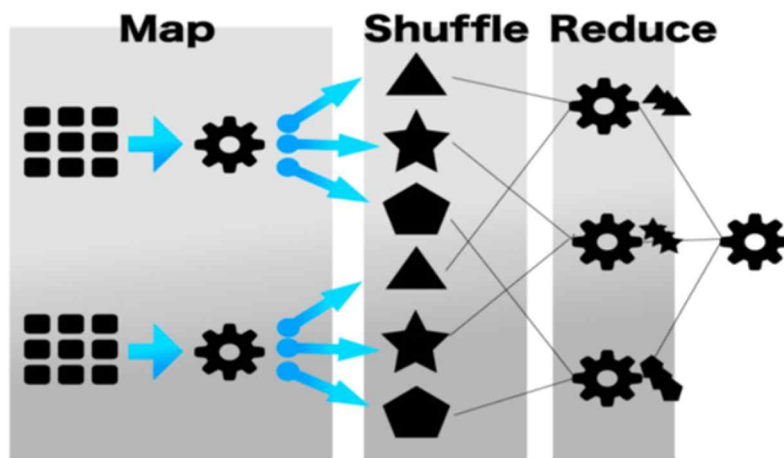


그림 2. MapReduce 데이터 처리 과정

출처: <https://dataonair.or.kr/db-tech-reference/d-lounge/technical-data/?mod=document&uid=235894>

1-2. 클라우드 관련 기술 현황 및 향후 전망

최근 국내 클라우드 시장이 주목받고 있습니다. ‘2021년 국내 클라우드 산업 실태 조사 결과 보고서’에 의하면, 2020년 국내 클라우드 시장은 4조 200억 원 규모로 전년 대비 19% 증가하였습니다. 유형별로는 IaaS가, 이용 형태별로는 하이브리드 클라우드(온프레미스 + 클라우드)가 가장 높은 비중을 보였습니다. 이는 현재 국내 기업 및 공공기관이 사용 중인 온프레미스 인프라를 클라우드로 전환하는 작업이 활발히 이어지고 때문에, **시장이 점점 확대되고 있다는 의미로 해석되고** 있습니다. PaaS 시장 또한 상대적으로 비중이 낮지만, 시장 규모가 1300억 원에서 2670억 원으로 두 배 이상 성장했습니다. 기업과 기관이 PaaS를 도입하고 관련 솔루션 및 서비스를 요구하기 때문에 시장이 활성화되었고, 클라우드 시장 내 가장 큰 성장 폭을 기록했습니다.

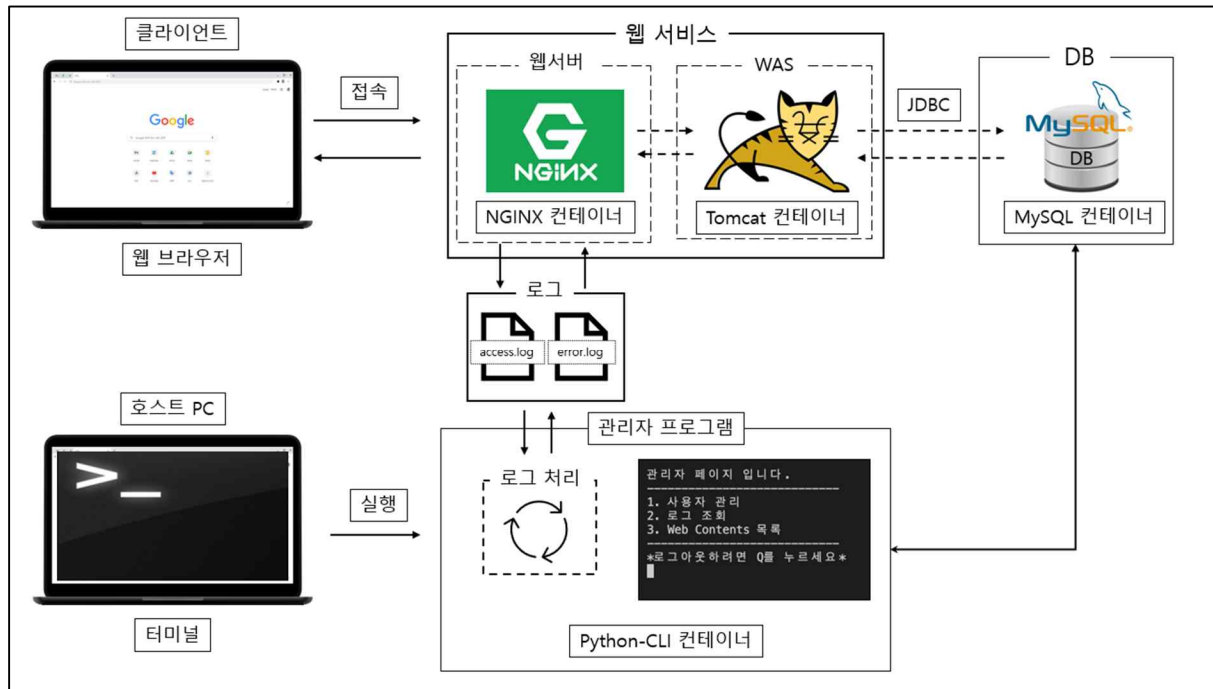
클라우드 담당자 인력 현황도 흥미로웠습니다. 전체 22,834명 중 연구&개발(R&D) 종사자가 38.3%로 가장 높게 나타났습니다. 현재 국내 클라우드 서비스가 MS, AWS, 구글과 같은 글로벌 클라우드 서비스 제공사에 비해 상대적으로 기술력이 떨어지기 때문에 **기업에서 기술 개발에 힘을 쏟고 있다고 해석**됩니다. 하지만 클라우드 인력의 업무 절반 이상은 운영 업무를 맡기 때문에, 종사자는 보고서에 명시된 인원보다 훨씬 많을 것으로 예상된다고 합니다.

보고서에 의하면, 향후 **국내 클라우드 산업의 활성화는 보안 기술에 좌우될 것으로 전망**하고 있습니다. 클라우드는 리소스를 직접 관리하는 것이 어렵기 때문에 보안 기술이 더욱 강조되고 있으며, 이에 대한 신뢰를 형성하는 것이 시장 활성화의 큰 열쇠가 될 것으로 예상됩니다. 특히 대기업의 경우 보안 이슈가 생기면 기업에 큰 타격을 줄 수 있기 때문에 현직자들이 가장 중요하게 생각하는 분야로 여겨지고 있습니다. 또한 최근 공공기관 및 금융권에서 클라우드 망 분리 규제 완화 등 중요한 보안 이슈가 계속 수면 위로 떠오르고 있습니다. 빅데이터로 대변되는 4차 산업 시대에서 클라우드 기술과 보안 기술은 선택이 아닌 필수가 되고 있습니다.

2. 프로젝트 기획

2-1. 프로젝트 구조 설명

■ 시스템 구성도



2-2. 사용 툴 정리

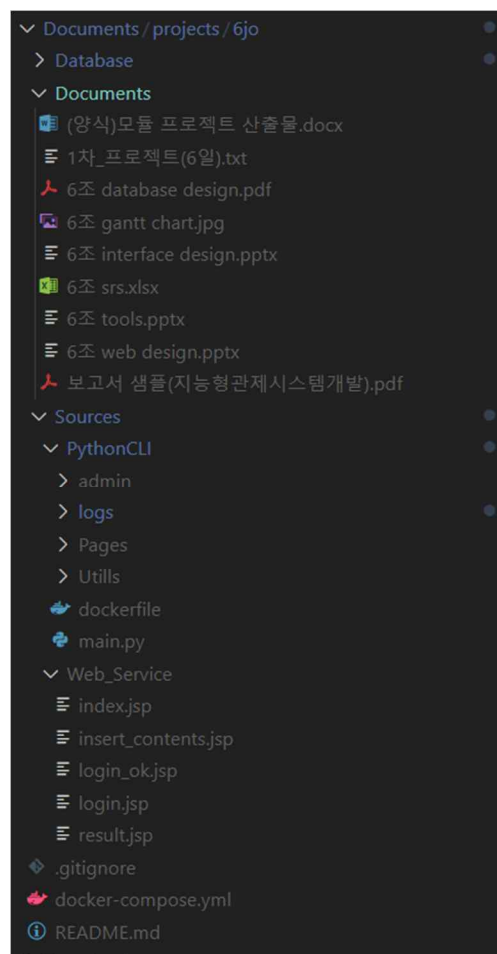
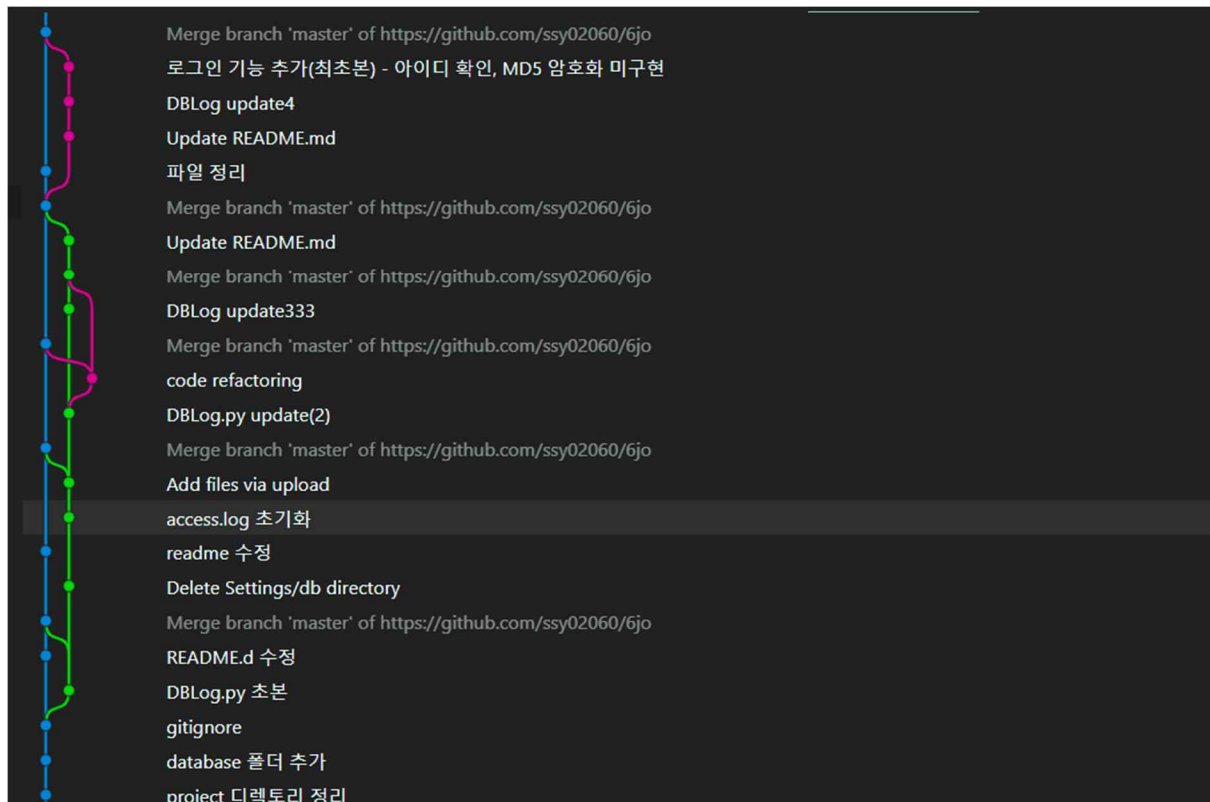


■ 정의

형상 관리 도구 (버전 관리 시스템)

■ 장점

- ① 소스코드를 주고 받을 필요 없이, 같은 파일을 여러 명이 동시에 작업하는 **병렬 개발**이 가능하다.
- ② **분산 버전 관리**이기 때문에 인터넷이 연결되지 않은 곳에서도 개발을 진행할 수 있다.
- ③ pull을 통한 업데이트 및 patch 파일을 통한 프로그램을 배포하는데 유용하다.



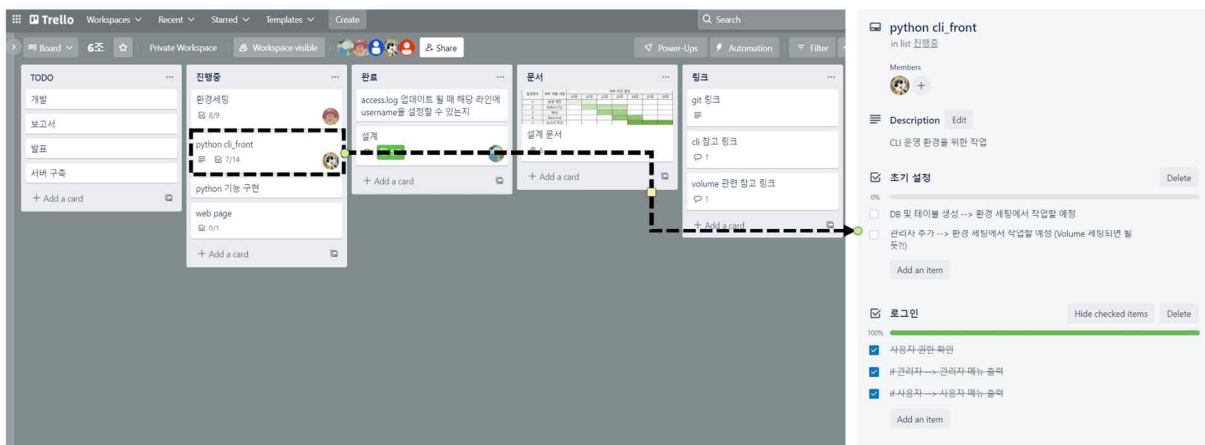


■ 정의

리눅스 컨테이너 기반으로 하는 오픈소스 가상화 플랫폼

■ 장점

- ① 가상화를 통해 안정성을 높이며 리소스를 최대한 활용할 수 있다.
- ② OS 레벨의 가상화로 프로세스를 격리시켜 동작하는 방식으로 이루어진 컨테이너를 기반으로 한다.
- ③ 성능 향상과 뛰어난 이식성, 쉽게 Scale Out을 할 수 있는 유연성을 가졌다.



■ 정의

프로젝트 관리형 협업 툴

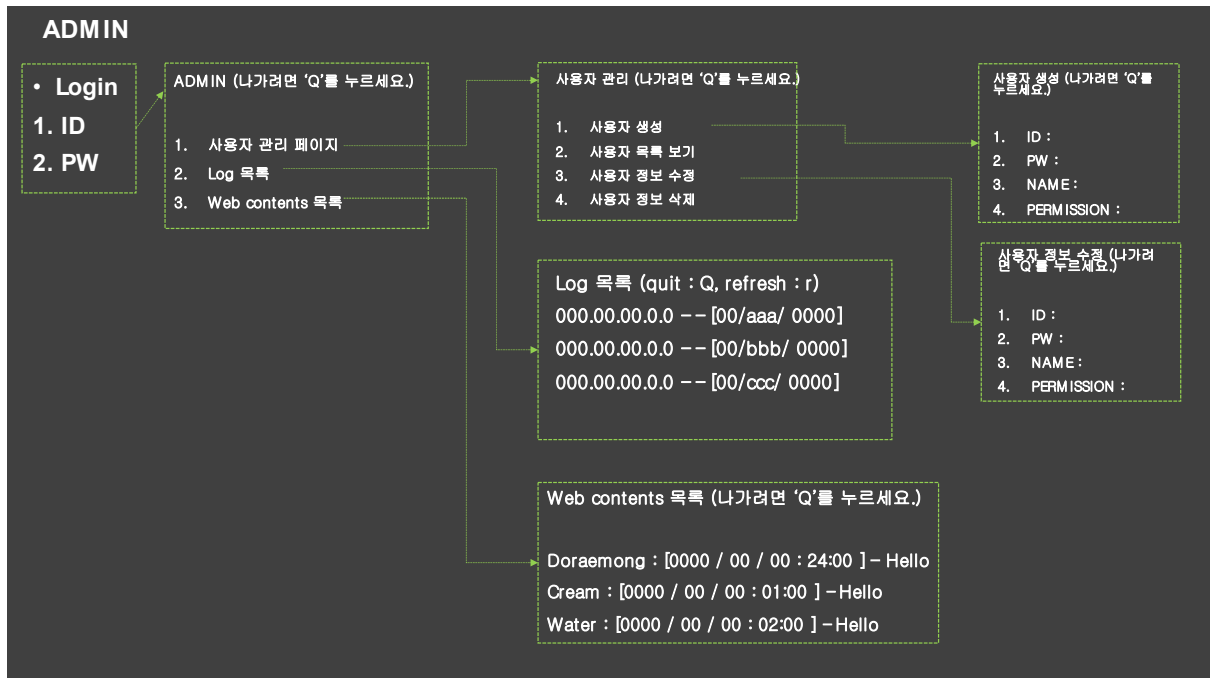
■ 장점

- ① 여러 사람과 소통하며 할 일들을 관리할 수 있다.
- ② 프로젝트를 효율적으로 관리, 진행하도록 도와준다.

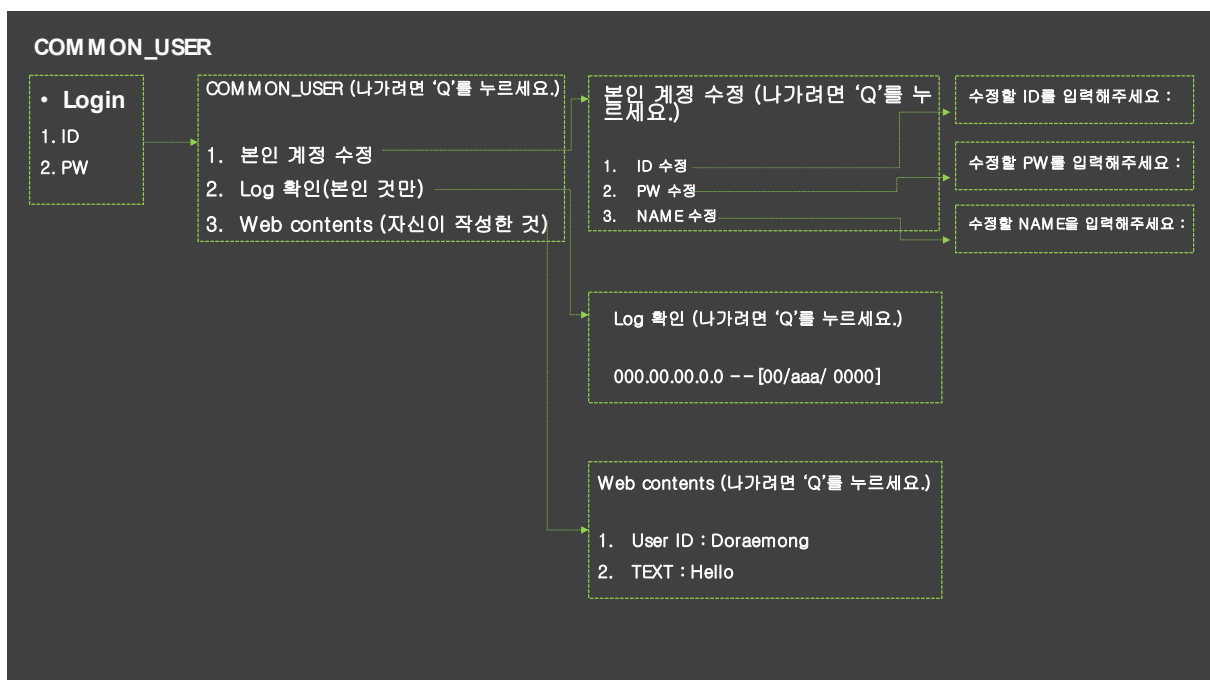
3. 프로젝트 설계

3-1. 시스템 인터페이스 정의

■ 관리자 인터페이스



■ 일반 사용자 인터페이스



3-2. 시스템 요구사항 정의

■ Python-CLI 요구사항 정의서

| Python-CLI 요구사항 정의서 | | | | | | | |
|---------------------|--------|-------------|-------------------|-----------------|--|---------------------------|---|
| 번호 | 환경 | 위치 | 요구사항 ID | 요구사항 이름 | 내용 | 예외 처리 | 비고 |
| 1 | PY-CLI | LOGIN | LOGIN-01 | 로그인 | 사용자의 계정 정보를 확인하고 로그인 승인 | 내용이 포맷에 맞게 입력 되었는지 확인 필요 | ID: PW: |
| 2 | | MAIN | MAIN-01 | 메뉴얼 출력 | ADMIN, COMMON_USER 구분 메인 페이지에 사용할 수 있는 기능을 출력시킨다. | 1~3 를 입력했는지 확인 필요 | ADMIN 1. 사용자 관리 페이지 2. Log 목록 3. Web content 목록 COMMON_USER 1. 본인 계정 수정 2. 로그 확인 3. Web contents |
| 3 | | ADMIN | ADMIN-01 | 사용자 관리 페이지 | 사용자 관리에 사용할 수 있는 기능을 출력시킨다. Q를 눌러 나갈 수 있다. | 1~4를 입력했는지 확인 필요 | 1. 사용자 생성 2. 사용자 목록 보기 3. 사용자 정보 수정 4. 사용자 삭제 |
| 4 | | | ADMIN-01-01 | 사용자 생성 | 사용자 정보를 입력받아 사용자를 생성한다. Q를 눌러 나갈 수 있다. | 내용들이 포맷에 맞게 입력 되었는지 확인 필요 | ID: PW: NAME: PERMISSION: |
| 5 | | | ADMIN-01-02 | 사용자 목록 보기 | DB에서 사용자 목록을 받아와서 출력한다. Q를 눌러 나갈 수 있다. | | |
| 6 | | | ADMIN-01-03 | 사용자 정보 수정 | 사용자 정보를 입력받아 사용자의 정보를 수정한다. | 내용들이 포맷에 맞게 입력 되었는지 확인 필요 | ID: PW: NAME: PERMISSION: |
| 7 | | | ADMIN-01-04 | 사용자 삭제 | 사용자 정보를 삭제한다. (1. User ID를 통해 삭제하는 방법 2. 목록을 보고 관리자가 선택하여 삭제하는 방법) | | 방법 1. 사용자 ID 입력: 방법 2. * doraemong ssy02060 jmo2060 |
| 8 | | | ADMIN-02 | Log 목록 | DB에서 Log 목록을 받아와서 출력한다. Q를 눌러 나갈 수 있다. | | |
| 9 | | | ADMIN-03 | Web contents 목록 | DB에서 Web contents를 받아와서 출력한다. Q를 눌러 나갈 수 있다. | | |
| 10 | | COMMON-USER | COMMON-USER-01 | 본인 계정 수정 | 본인의 계정을 수정할 수 있는 기능을 출력한다. Q를 눌러 나갈 수 있다. | 1~3 를 입력했는지 확인 필요 | 1. ID 수정 2. PW 수정 3. NAME 수정 |
| 11 | | | COMMON-USER-01-01 | ID 수정 | 수정할 ID를 입력하여 ID를 수정한다. | 내용이 포맷에 맞게 입력 되었는지 확인 필요 | 수정할 ID를 입력해주세요: |
| 12 | | | COMMON-USER-01-02 | PW 수정 | 수정할 PW를 입력하여 PW를 수정한다. | 내용이 포맷에 맞게 입력 되었는지 확인 필요 | 수정할 PW를 입력해주세요: |
| 13 | | | COMMON-USER-01-03 | NAME 수정 | 수정할 NAME을 입력하여 NAME을 수정한다. | 내용이 포맷에 맞게 입력 되었는지 확인 필요 | 수정할 NAME을 입력해주세요: |
| 14 | | | COMMON-USER-02 | Log 확인 | DB에서 본인의 Log 목록을 받아와서 출력한다. Q를 눌러 나갈 수 있다. | | |
| 15 | | | COMMON-USER-03 | Web contents 확인 | DB에서 본인이 작성한 Web contents를 받아와서 출력한다. Q를 눌러 나갈 수 있다. | | |

■ Web 요구사항 정의서

| Web 요구사항 정의서 | | | | | | | |
|--------------|--------------|-------|----------|-----------------|-------------------------|--------------------------|------------|
| 번호 | 환경 | 위치 | 요구사항 ID | 요구사항 이름 | 내용 | 예외처리 | 비고 |
| 1 | WEB-CONTENTS | LOGIN | LOGIN-01 | 로그인 | 사용자의 계정 정보를 확인하고 로그인 승인 | 내용이 포맷에 맞게 입력 되었는지 확인 필요 | ID: PW: |
| 2 | | MAIN | MAIN-01 | Web contents 입력 | 사용자가 입력한 내용이 DB에 저장된다. | | 내용: |

3-3. Database 설계

■ 시스템의 사용자 정보를 저장하는 테이블

| Table Name | user | | | | |
|-------------|-------------------|-------------|--------------|------------|----------------|
| Description | 사용자 테이블 | | | | |
| Create | 환경 세팅 시점에 테이블 생성 | | | | |
| Insert | 관리자가 사용자 추가 | | | | |
| Delete | 관리자가 사용자 삭제 | | | | |
| Update | 관리자 혹은 사용자 본인이 수정 | | | | |
| No. | Field | Type | Description | Update | Remark |
| 1 | auth | int | 사용자 권한 | 관리자만 변경 가능 | 0: 관리자, 1: 사용자 |
| 3 | id | varchar(32) | 사용자 ID | | primary key |
| 4 | passwd | varchar(32) | 사용자 Password | | md5 값 저장 |
| 5 | name | varchar(32) | 사용자 이름 | | |

■ 웹 로그를 저장하는 테이블

| Table Name | agentlog | | | | |
|-------------|----------------------|--------------|--------------|--------|--------|
| Description | Nginx 웹 로그 테이블 | | | | |
| Create | 환경 세팅 시점에 테이블 생성 | | | | |
| Select | 로그 조회 메뉴 선택 시 테이블 조회 | | | | |
| No. | Field | Type | Description | Update | Remark |
| 1 | user.id | varchar(32) | user 테이블의 id | | |
| 2 | ipAddr | varchar(32) | IP 주소 | | |
| 3 | time | varchar(255) | 접속 시간 | | |
| 4 | method | varchar(32) | method | | |
| 5 | url | varchar(255) | 접속 url | | |
| 6 | code | int | 응답 코드 | | |
| 7 | details | varchar(255) | 나머지 정보 | | |

■ 웹 콘텐츠와 관련된 정보를 저장하는 테이블

| Table Name | web_contents | | | | |
|-------------|-------------------------|--------------|----------------|--------|--------|
| Description | 웹 콘텐츠 테이블 | | | | |
| Create | 환경 세팅 시점에 테이블 생성 | | | | |
| Insert | 사용자가 웹에 접근했을 때 추가 | | | | |
| Select | 웹 콘텐츠 조회 메뉴 선택 시 테이블 조회 | | | | |
| No. | Field | Type | Description | Update | Remark |
| 1 | id | varchar(32) | 웹 페이지에 로그인한 id | | |
| 2 | time | varchar(255) | 웹 콘텐츠 작성 시간 | | |
| 3 | contents | varchar(255) | 웹 콘텐츠 내용 | | |

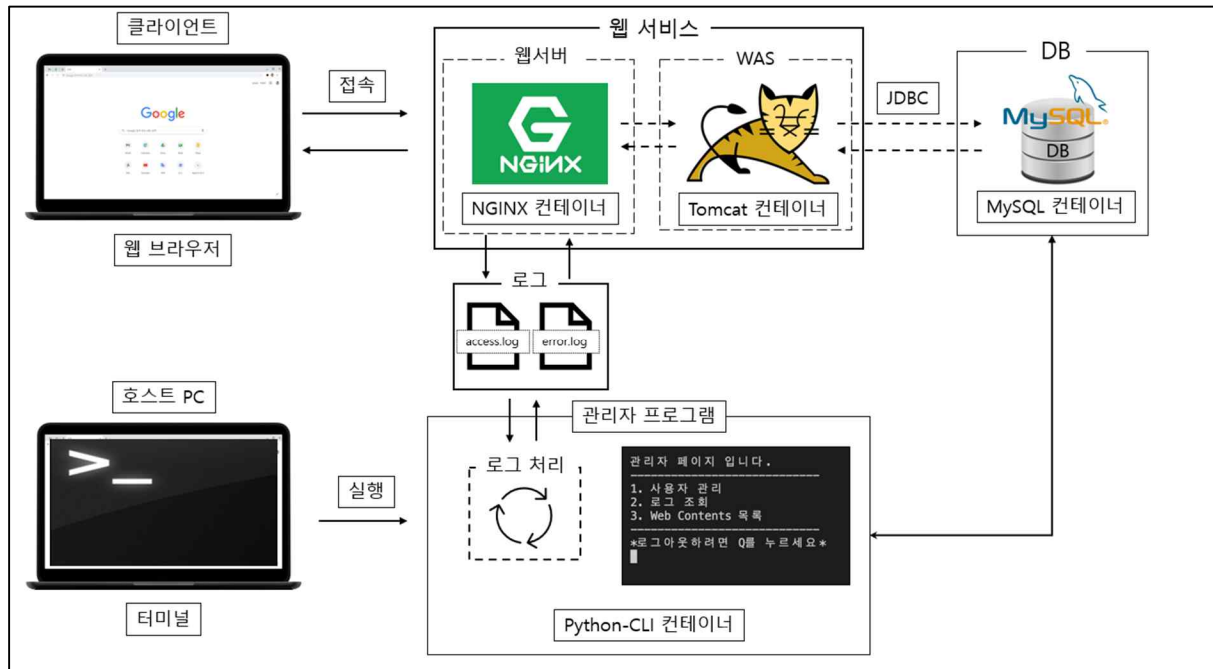
3-4. 시스템 개발 일정

| 일련번호 | 세부 개발 내용 | 세부 추진 일정 | | | | | | |
|------|------------|----------|-----|-----|-----|-----|-----|-----|
| | | 20일 | 21일 | 22일 | 23일 | 24일 | 25일 | 26일 |
| 1 | 환경 세팅 | | | | | | | |
| 2 | Python-CLI | | | | | | | |
| 3 | Web | | | | | | | |
| 4 | Back-end | | | | | | | |
| 5 | 보고서 작성 | | | | | | | |

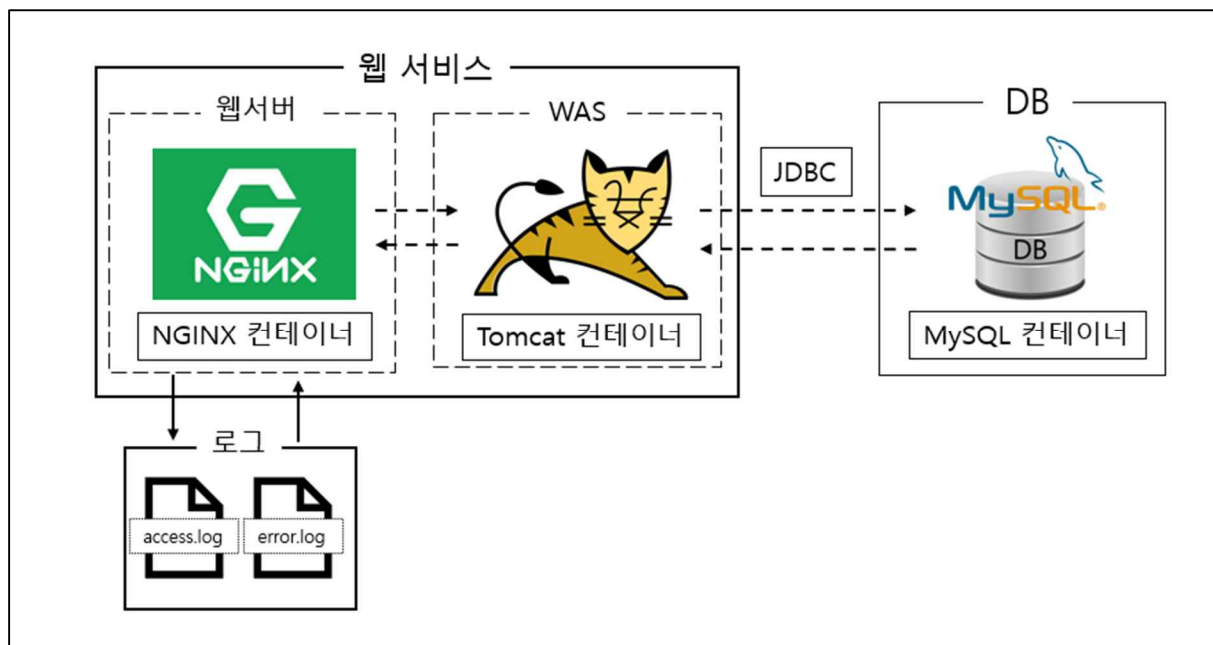
4. 프로젝트 개발

4-1. 개발 및 운영 환경 세팅

■ 시스템 구성도



■ 웹 서비스 구성



a. NGINX + Tomcat 연동 Docker 컨테이너 구성

Nginx 컨테이너의 /etc/nginx/conf.d/default.conf 내 proxy_pass에 Tomcat 서버의 주소를 입력합니다.

```
server {
    listen      80;
    listen  [::]:80;
    server_name localhost;

    access_log  /var/log/nginx/default/access.log main;
    error_log   /var/log/nginx/default/error.log main;

    index index.html index.htm index.jsp;
    location / {
        # root   /usr/share/nginx/html;
        root    /var/www/html;
        proxy_pass http://172.33.0.3:8080/;
    }
}
```

그림 3. /etc/nginx/conf.d/default.conf

이때 Nginx에서 수집한 로그를 Host PC에 공유하기 위해서 볼륨으로 마운트가 필요합니다. 해당 내용은 docker-compose.yaml 파일에 아래와 같이 작성하였습니다.

```
nginx-web:
  image: ssy02060/nginx-web:latest
  restart: always
  ports:
    - "80:80"
  container_name: nginx-web
  volumes:
    - ./Sources/PythonCLI/logs:/var/log/nginx/default

  networks:
    broker-net:
      ipv4_address: 172.33.0.04
```

그림 4. docker-compose 내 nginx 설정 내용

컨테이너 간 통신이 이루어지기 위해서는 network를 구성해 주어야 하는데 네트워크 생성 명령어는 아래와 같습니다.

```
$ sudo docker network create --gateway 172.33.0.1 --subnet 172.33.0.0/16 broker
```

컨테이너 간 broker 네트워크를 공유한다는 의미로 docker-compose.yaml 파일에 아래와 같이 작성하였습니다.

```
networks:
  broker-net:
    external:
      name: broker
```

해당 내용을 docker-compose를 사용하지 않고 명령어로 입력하면 아래와 같습니다.

```
sudo docker run -d --network=broker -p 80:80 --restart=always -v /home/six-jo/Documents/projects/6jo/Sources/PythonCLI/logs:/var/log/nginx/default nginx-web ssy02060/nginx-web:latest
```

docker-compse를 활용하면 훨씬 직관적이게 정의할 수 있다는 사실을 알 수 있습니다.

b. Tomcat + MySQL 연동 Docker 컨테이너 구성

Tomcat 서버와 MySQL 서버를 연동하려면 JDBC를 활용해야 하는데 이때 Tomcat 컨테이너에서 JDBC를 사용하려면 JDBC를 설치하고 환경변수를 세팅해야 합니다.

JDBC 설치에 아래 두 명령어를 JDBC가 설치된 호스트 PC 디렉토리에서 입력하면 됩니다.

```
sudo docker cp ./mysql-connector-java-8.0.28.jar:/usr/local/openjdk-11/
sudo docker cp ./mysql-connector-java-8.0.28.jar:/usr/local/tomcat/
```

환경변수 세팅은 Docker 컨테이너 내부의 /etc/profile 파일에 해당 내용을 추가하면 됩니다.

```
export CLASSPATH=.:$JAVA_HOME/lib/mysql-connector-java-8.0.28.jar:$CATALINA_HOME/lib/mysql-connector-java-8.0.28.jar
PATH=$PATH:$JAVA_HOME/bin:$CATALINA_HOME/bin
```

추가적으로 Tomcat 서버를 구현하기 위한 jsp 파일들을 호스트 PC에서 편집하기 위해 볼륨을 마운트 해주었고, 컨테이너 간 통신을 위해 broker 네트워크로 묶어주었습니다.

해당 내용은 docker-compose.yaml 파일에 아래와 같이 작성하였습니다.

```
tomcat-server:
  image: ssy02060/tomcat-server:latest
  restart: always
  ports:
    - "8080:8080"
    - "8009:8009"
  container_name: tomcat-server
  depends_on:
    - mysql-server
  volumes:
    - sangyun, 5일 전 • tomcat 웹 콘텐츠 jsp ...
    - ./Sources/Web_Service:/usr/local/tomcat/webapps/ROOT
  networks:
    broker-net:
      ipv4_address: 172.33.0.03
```

그림 5. docker-compose 내 Tomcat 설정 내용

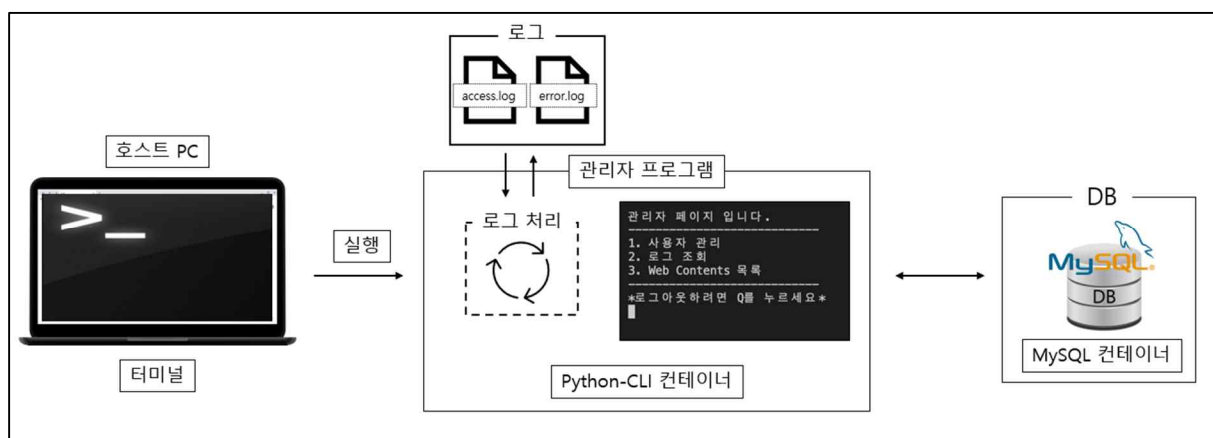
c. MySQL Docker 컨테이너 구성

```
mysql-server:
  image: ssy02060/db-mysql:latest
  restart: always
  container_name: mysql-server
  environment:
    - MYSQL_DATABASE=ccloud
    - MYSQL_ROOT_PASSWORD=abcd
    - TZ=Asia/Seoul
  ports:
    - "3306:3306"
  networks:
    broker-net:
      ipv4_address: 172.33.0.2
  command:
    - --character-set-server=utf8mb4
    - --collation-server=utf8mb4_unicode_ci
  volumes:
    - ./Database/conf.d:/etc/mysql/conf.d
    - ./Database/data:/var/lib/mysql
    - ./Database/initdb.d:/docker-entrypoint-initdb.d
```

그림 6. docker-compose 내 MySQL 설정 내용

MySQL에 필요한 기본 설정에 대한 정의는 docker-compose.yaml 파일에 위와 같이 정의 하였습니다. 데이터베이스의 데이터가 유지되려면 Host PC와 volume으로 마운트 되어야 합니다. 마찬가지로 컨테이너 간 통신을 위해 broker 네트워크로 묶어주었습니다.

d. Python-CLI 컨테이너 구성



Docker Container를 활용하여 Interactive 한 Python 프로그램을 구현하기 위한 방법은 두 가지가 있습니다. 첫 번째 방법은 ubuntu 이미지를 생성하고 해당 컨테이너에 python을 설치하여 사용하는 방법입니다. 두 번째 방법은 python 이미지를 생성하고 컨테이너를 실행할 때 tty(터미널 사용 여부) 와 stdin_open(표준 입출력 사용 여부) option을 설정하여 사용하는 방법이 있습니다.

저희 팀은 이미 우분투가 설치된 가상머신에 다시 ubuntu를 설치한다는 것이 리소스 낭비라 판단하여 두 번째 방법으로 도커 컨테이너를 구성하였습니다. 또한 Nginx에서 기록되는 로그를 Python-CLI 프로그램에서 실시간으로 처리하기 위해 볼륨 파일을 마운트 했습니다. 마찬가지로 DB 컨테이너와 통신하기 위해 broker 네트워크로 묶어주었습니다. 해당 내용은 docker-compose.yaml 파일에 아래와 같이 작성하였습니다.

```
python-cli:
  image: python-cli:latest
  container_name: python-cli
  build:
    context: .
    dockerfile: ./Sources/PythonCLI/dockerfile
  depends_on:
    - mysql-server
  volumes:
    - ./Sources/PythonCLI/logs:/app/cli/logs/
  tty: true
  stdin_open: true
  networks:
    broker-net:
      ipv4_address: 172.33.0.5
```

그림 7. docker-compose.yaml 내 python-cli 설정 내용

도커 컨테이너가 실행되려면 이미지가 존재해야 하는데 위의 세 컨테이너와는 다르게 python-cli 컨테이너는 dockerfile 을 구성하여 빌드하도록 하였습니다.

```
sangyun, 어제 | 2 authors (sangyun and others)
FROM python:3.9
LABEL maintainer="six-jo"

COPY ./Sources/PythonCLI /app/cli

WORKDIR /app/cli

RUN pip install pymysql[rsa]
CMD ["python", "main.py"]
```

그림 8. dockerfile

4-2. Python-CLI

a. 로그인

시스템을 실행하면 가장 먼저 로그인 인터페이스가 출력됩니다. 입력한 ID 와 Password 에 대한 사용자 정보를 확인하고, 권한에 따라 관리자와 일반 사용자로 구분됩니다. 사용자 정보가 없는 경우, 다시 ID 와 Password 를 입력받습니다. 'q'를 입력하면 시스템이 종료됩니다.

```
<로그인 해주세요. (종료 : q)>
ID: █
```

그림 9. 로그인 인터페이스

사용자의 정보를 저장하는 user 테이블에는 사용자의 권한, ID, Password, 이름에 대한 정보가 저장됩니다. 이 중 ID 는 사용자를 구분하는 기본 키(Primary Key)이므로 중복하여 등록될 수 없습니다. 사용자가 입력한 Password 는 MD5 알고리즘을 통해 암호화되어 데이터베이스에 저장됩니다.

```
mysql> select * from user;
+-----+-----+-----+-----+
| auth | id      | passwd                                     | name |
+-----+-----+-----+-----+
| 0    | admin  | 21232f297a57a5a743894a0e4a801fc3      | admin |
| 1    | han    | 4a7d1ed414474e4033ac29ccb8653d9b      | sh    |
| 1    | oh     | 4a7d1ed414474e4033ac29ccb8653d9b      | sh    |
| 1    | seo    | 4a7d1ed414474e4033ac29ccb8653d9b      | sy    |
| 1    | yoo    | 4a7d1ed414474e4033ac29ccb8653d9b      | ds    |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

그림 10. user 테이블에 저장된 사용자 정보

b. 관리자

관리자 인터페이스는 다음과 같습니다. 'q' 입력 시 관리자 페이지를 나와 로그인 화면으로 돌아갈 수 있습니다.

```
관리자 페이지 입니다.
-----
1. 사용자 관리
2. 로그 조회
3. Web Contents 목록
-----
*로그아웃하려면 Q를 누르세요*
█
```

그림 11. 관리자 인터페이스

i. 사용자 관리

1 번 선택 시 사용자 관리 메뉴가 출력됩니다. 각각의 옵션을 선택하여 원하는 작업을 수행할 수 있습니다. 사용자 생성 시 이미 등록된 ID 를 입력하는 경우, 사용자가 생성되지 않습니다.

사용자 관리 항목을 선택해 주세요 .

- 1. 사용자 생성
- 2. 사용자 목록 보기
- 3. 사용자 정보 수정
- 4. 사용자 정보 삭제

나가려면 Q를 입력하세요

■

그림 12. 사용자 관리 페이지

사용자 생성 메뉴입니다 . 사용자 정보를 입력하세요 .

ID : kim
 PASSWORD : 0000
 PASSWORD 확인 : 0000
 이름 : js
 권한 : 1
 kim 님이 추가되었습니다 . 나가려면 아무 키나 입력하세요
 나가려면 아무 키나 입력하세요

■

그림 13. 사용자 생성 메뉴 (정상적으로 사용자가 생성된 경우)

전체 사용자 목록을 조회합니다 .

| ID | PW | NAME | AUTH |
|-------|----------------------------------|-------|------|
| admin | 21232f297a57a5a743894a0e4a801fc3 | admin | 0 |
| han | 4a7d1ed414474e4033ac29ccb8653d9b | sh | 1 |
| kim | 4a7d1ed414474e4033ac29ccb8653d9b | js | 1 |
| oh | 4a7d1ed414474e4033ac29ccb8653d9b | sh | 1 |
| seo | 4a7d1ed414474e4033ac29ccb8653d9b | sy | 1 |
| yoo | 4a7d1ed414474e4033ac29ccb8653d9b | ds | 1 |

나가려면 아무 키나 입력하세요

■

그림 14. 사용자 목록 보기 메뉴

사용자 수정 페이지입니다 . (사용자 목록)

| ID | PW | NAME | AUTH |
|-------|----------------------------------|-------|------|
| admin | 21232f297a57a5a743894a0e4a801fc3 | admin | 0 |
| han | 4a7d1ed414474e4033ac29ccb8653d9b | sh | 1 |
| oh | 4a7d1ed414474e4033ac29ccb8653d9b | sh | 1 |
| seo | 4a7d1ed414474e4033ac29ccb8653d9b | sy | 1 |
| yoo | 4a7d1ed414474e4033ac29ccb8653d9b | ds | 1 |

수정할 유저 ID를 입력하세요 . (나가려면 q를 입력하세요)

해당 유저는 존재하지 않습니다 . ID를 확인해 주세요

ID : ■

그림 15. 사용자 정보 수정 메뉴

수정 할 항목을 선택해주세요 .

- 1. Password
- 2. Name
- 3. Permission

나가려면 Q를 입력하세요

■

그림 16. 수정 항목 선택

수정 할 Password를 입력해주세요 :

1111

수정이 완료되었습니다 .

나가려면 아무 키나 입력하세요

■

그림 17. Password 수정

수정 할 Name을 입력해주세요 :

shoh

수정이 완료되었습니다 .

나가려면 아무 키나 입력하세요

■

그림 18. Name 수정

권한을 입력하세요 (0 또는 1) : 0

수정이 완료되었습니다 .

나가려면 아무 키나 입력하세요

■

그림 19. 권한 수정

전체 사용자 목록을 조회합니다 .

| ID | PW | NAME | AUTH |
|-------|----------------------------------|-------|------|
| admin | 21232f297a57a5a743894a0e4a801fc3 | admin | 0 |
| han | 4a7d1ed414474e4033ac29ccb8653d9b | sh | 1 |
| kim | 4a7d1ed414474e4033ac29ccb8653d9b | js | 1 |
| oh | b59c67bf196a4758191e42f76670ceba | shoh | 0 |
| seo | 4a7d1ed414474e4033ac29ccb8653d9b | sy | 1 |
| yoo | 4a7d1ed414474e4033ac29ccb8653d9b | ds | 1 |

나가려면 아무 키나 입력하세요

■

그림 20. 수정된 사용자 정보 확인

그림 20은 oh 사용자의 정보를 수정한 후 사용자 목록을 조회한 내용입니다. 그림 14과 비교했을 때 Password, Name, 권한이 수정된 것을 확인할 수 있습니다.

사용자 삭제 페이지에서는 삭제할 사용자의 ID를 입력합니다. ID가 존재하지 않는 경우, 다시 입력을 받으며 ID가 존재하는 경우, 관리자에게 'y' 또는 'n' 을 입력받아 삭제를 수행할지 한 번 더 확인합니다.

```

사용자 삭제 페이지입니다. (사용자 목록)
-----
ID      PW      NAME      AUTH
admin   21232f297a57a5a743894a0e4a801fc3      admin   0
han     4a7d1ed414474e4033ac29ccb8653d9b      sh      1
kim     4a7d1ed414474e4033ac29ccb8653d9b      js      1
oh      b59c67bf196a4758191e42f76670ceba      sh      1
seo     4a7d1ed414474e4033ac29ccb8653d9b      sy      1
yoo     4a7d1ed414474e4033ac29ccb8653d9b      ds      1
수정할 유저 ID를 입력하세요. (나가려면 q를 입력하세요)
ID : █
  
```

그림 21. 사용자 정보 삭제 메뉴

```

kim님을 삭제 하시겠습니까? (y/n)
y
kim님이 삭제되었습니다.
*나가려면 아무 키나 입력하세요*
█
  
```

그림 22. 정상적으로 사용자가 삭제된 경우

ii. 로그 조회

관리자는 모든 사용자의 웹 서버 접속 로그를 조회할 수 있습니다. 로그를 조회하여 특정 사용자의 이상 행위를 파악하고 보안을 점검할 수 있습니다. 'r' 입력 시 로그를 새로 조회하여 시스템 실행 중에 추가된 로그를 확인할 수 있습니다.

```

-----
사용자 로그를 조회합니다.
-----
ID  IP      DATE          METHOD  URL          STATUS
None 10.0.2.2 [26/Apr/2022:04:23:03 +0000] POST / HTTP/1.1 200
*나가려면 q, 새로고침하려면 r을 입력하세요*
█
  
```

그림 23. 로그 조회 페이지

```

-----
사용자 로그를 조회합니다.
-----
ID  IP      DATE          METHOD  URL          STATUS
None 10.0.2.2 [26/Apr/2022:04:23:03 +0000] POST / HTTP/1.1 200
han 10.0.2.2 [26/Apr/2022:04:25:14 +0000] POST /login.jsp HTTP/1.1 200
han 10.0.2.2 [26/Apr/2022:04:25:15 +0000] POST /login_ok.jsp HTTP/1.1 200
*나가려면 q, 새로고침하려면 r을 입력하세요*
█
  
```

그림 24. 로그 새로고침

iii. Web Contents 목록

뿐만 아니라 관리자는 사용자가 작성한 모든 웹 콘텐츠를 조회할 수 있습니다.

전체 글 목록을 조회합니다.

| INDEX | ID | CONTENTS |
|-------|-----|-----------------|
| 1 | han | 안녕하세요 |
| | | *나가려면 q를 입력하세요* |

█

그림 25. Web Contents 목록 조회 페이지

c. 일반 사용자

일반 사용자 인터페이스는 다음과 같습니다. 'q' 입력 시 사용자 페이지를 나와 로그인 화면으로 돌아갈 수 있습니다.

사용자 페이지 입니다 .

1. 사용자 수정
2. 사용자 로그 조회
3. 작성한 Web Contents 확인

로그아웃하려면 Q를 누르세요

█

그림 26. 일반 사용자 인터페이스

i. 사용자 수정

1 번 메뉴 선택 시 사용자 본인의 정보를 수정할 수 있으며 데이터베이스에 즉시 반영됩니다.

수정할 항목을 선택해주세요 .

1. Password
2. Name

나가려면 Q를 입력하세요

그림 27. 수정 항목 선택

수정할 Password를 입력해주세요 :

█

그림 28. Password 수정

수정할 Name을 입력해주세요 :

그림 29. Name 수정

전체 사용자 목록을 조회합니다.

| ID | PW | NAME | AUTH |
|-------|----------------------------------|-------|------|
| admin | 21232f297a57a5a743894a0e4a801fc3 | admin | 0 |
| han | 4a7d1ed414474e4033ac29ccb8653d9b | sh | 1 |
| oh | 4a7d1ed414474e4033ac29ccb8653d9b | sh | 1 |
| seo | 4a7d1ed414474e4033ac29ccb8653d9b | sy | 1 |
| yoo | 4a7d1ed414474e4033ac29ccb8653d9b | ds | 1 |

나가려면 아무 키나 입력하세요

그림 30. han 사용자의 Password, Name 수정 전

전체 사용자 목록을 조회합니다.

| ID | PW | NAME | AUTH |
|-------|----------------------------------|-------|------|
| admin | 21232f297a57a5a743894a0e4a801fc3 | admin | 0 |
| han | 25bbdc06c32d477f7fa1c3e4a91b032 | hand | 1 |
| oh | 4a7d1ed414474e4033ac29ccb8653d9b | sh | 1 |
| seo | 4a7d1ed414474e4033ac29ccb8653d9b | sy | 1 |
| yoo | 4a7d1ed414474e4033ac29ccb8653d9b | ds | 1 |

나가려면 아무 키나 입력하세요

그림 31. han 사용자의 정보 수정 후

그림 31은 han 사용자의 정보를 수정한 후 사용자 목록을 조회한 내용입니다. 그림 30과 비교했을 때 Password와 Name이 수정된 것을 확인할 수 있습니다.

ii. 사용자 로그 조회

2 번 메뉴 선택 시 웹에 접속한 사용자의 로그를 조회할 수 있습니다. 일반 사용자는 본인 ID 로 접속한 로그만을 조회할 수 있으며, 로그를 통해 웹에 접속한 IP 대역과 시간을 확인하여 보안을 점검할 수 있습니다. 'r' 입력 시 로그를 즉시 동기화하여 조회할 수 있고, 'q' 입력 시 로그 조회 페이지에서 나와 로그인 화면으로 돌아갑니다.

사용자 로그를 조회합니다.

| ID | IP | DATE | METHOD | URL | STATUS |
|----------------------------|----|------|--------|-----|--------|
| *나가려면 q, 새로고침하려면 r을 입력하세요* | | | | | |

그림 32. 사용자 로그 조회 페이지


```
-----
사용자 로그를 조회합니다.
-----
ID   IP       DATE                METHOD  URL                STATUS
han  10.0.2.2 [26/Apr/2022:05:36:11 +0000] POST  /login.jsp HTTP/1.1 200
han  10.0.2.2 [26/Apr/2022:05:36:12 +0000] POST  /login_ok.jsp HTTP/1.1 200
*n아가려면 q, 새로고침하려면 r을 입력하세요*
█
```

그림 33. 로그 새로고침

iii. 작성한 Web Contents 확인

3 번 메뉴 선택 시 사용자가 웹에 작성한 콘텐츠를 확인할 수 있습니다.

```
-----
han님의 글 목록을 조회합니다.
-----
ID      CONTENTS
han     반갑습니다
*n아가려면 q를 입력하세요*
█
```

그림 34. 사용자가 작성한 Web Contents 조회 페이지

4-3. 웹 로그 수집 및 콘텐츠 저장

a. 웹 로그 수집

Nginx 를 실행한 뒤 원하는 로그 정보를 수집할 수 있습니다. Nginx 설정 파일에서 포맷을 지정하여 사용자가 웹 서비스에 액세스 할 때마다 host pc 의 access.log 파일에 정보가 저장되게 설정하였습니다.

```
$remote_user HTTP Authorization 접속시 접속한 유저
$remote_addr 방문자 IP
$time_local 요청을 처리한 서버 시간
$status HTTP 응답코드
$request 클라이언트 요청
$http_user_agent 클라이언트가 사용한 브라우저
$http_referer 해당 페이지 이전에 거쳐온 URL
$body_bytes_sent 보낸 데이터 크기 (byte)
$http_x_forwarded_for 프록시 서버를 거치기 전의 접속 IP
$request_time 요청을 처리하는데 걸린시간
$connection 로그를 남길 당시의 커넥션 수
```

그림 35. 지정한 Nginx Logging 포맷

```
http {
    include      /etc/nginx/mime.types;
    default_type application/octet-stream;

    log_format  main  '$remote_addr [$request_body] [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    access_log  /var/log/nginx/access.log  main;
}
```

그림 36. nginx.conf 파일의 로그 설정

```
Documents > Projects > gjo > Sources > PythonCLI > logs > access.log
1 10.0.2.2 [-] [26/Apr/2022:06:37:05 +0000] "GET / HTTP/1.1" 200 379 "-" Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML
2 10.0.2.2 [id=han&pw=0001&submit=Login] [26/Apr/2022:06:37:10 +0000] "POST /login.jsp HTTP/1.1" 200 277 "http://localhost/" Mozilla/5.0 (Wi
3 10.0.2.2 [-] [26/Apr/2022:06:37:11 +0000] "GET /login_ok.jsp HTTP/1.1" 200 368 "http://localhost/login.jsp" Mozilla/5.0 (Windows NT 10.0; I
4 10.0.2.2 [contents=test&btn=%EB%93%B1%EB%A1%9D] [26/Apr/2022:06:37:15 +0000] "POST /insert_contents.jsp HTTP/1.1" 200 217 "http://localhost
5 10.0.2.2 [-] [26/Apr/2022:06:37:16 +0000] "GET /result.jsp HTTP/1.1" 200 196 "http://localhost/insert_contents.jsp" Mozilla/5.0 (Windows N
6 |
```

그림 37. access.log에 저장된 로그

access.log 파일을 읽어서 수집한 로그를 슬라이싱하여 데이터베이스의 agentlog 테이블에 저장하였습니다. agentlog 테이블의 컬럼은 id(현재 접속 중인 유저의 id), ip, time, method(GET / POST), url, code(HTTP 상태 코드), details(그 외)로 설정하였습니다.

```
mysql> desc agentlog;
```

| Field | Type | Null | Key | Default | Extra |
|---------|--------------|------|-----|---------|-------|
| id | varchar(32) | YES | | NULL | |
| ipAddr | varchar(32) | YES | | NULL | |
| time | varchar(255) | YES | | NULL | |
| method | varchar(32) | YES | | NULL | |
| url | varchar(255) | YES | | NULL | |
| code | int | YES | | NULL | |
| details | varchar(255) | YES | | NULL | |

7 rows in set (0.02 sec)

그림 38. agentlog 테이블

로그 파일 슬라이싱 작업은 파이썬 코드를 통해 구현하였으며, 해당 로그의 id 값 갱신을 위해 읽어온 로그 라인의 바로 이전 값의 id와 ip를 저장하여 비교하였습니다.

```
def logFileSlice(log: list, prev_id, prev_ip) -> list:
    # (0 : ip, 1: request_body, 2: 시간, 3: 메서드, 4: URI, 5: html_code, 6: etc)

    return_list = []
    #1 ip 출력
    end_ip_idx = log.find(' ', 0)
    current_ip = log[:end_ip_idx]
    return_list.append(current_ip)
```

그림 39. 로그 슬라이싱 함수

파이썬 실행 환경에서 로그 조회 요청 시, 새로운 로그 데이터를 데이터베이스에 저장합니다. 이때, 더 이상 저장할 데이터가 없으면 넘어갑니다.

```
def loadLog(self):
    # DB_line 과 Log_line을 비교하여 데이터베이스 최신화
    DB_line = len(DBLog.selectAllLog())
    log_line = 0
    _file = open("/app/cli/logs/access.log", "r")
    _list = _file.readlines()

    for log in _list:
        if log != '\n':
            log_line += 1

    if log_line != DB_line:
        if DB_line == 0:
            DBLog.autoSaveLog(0) # 최초 실행 시 처음부터 전부 저장
        else:
            DBLog.autoSaveLog(DB_line) # 그 외 새로운 로그만 저장
```

그림 40. 로그기록 데이터베이스 최신화

이제 최신화 된 데이터베이스를 출력합니다. 권한이 관리자이면 전체 기록을, 사용자이면 사용자 로그만 출력합니다.

```
def printLogs(self):
    # 관리자: 모든 로그 출력
    logs = DBLog.selectAllLog()

    print("ID      IP      DATE      METHOD  URL      STATUS")
    for log in logs:
        userId = log[0]
        userIP = log[1]
        date = log[2]
        method = log[3]
        url = log[4]
        status = str(log[5])
        details = log[6]
        print(userId,userIP,date,method,url,status)
```

그림 41. 관리자 로그 출력

```
def printLogs(self):
    # 사용자: 사용자 로그만 출력
    logs = DBLog.selectLog(self.userId)

    print("ID      IP      DATE      METHOD  URL      STATUS")
    for log in logs:
        userId = log[0]
        userIP = log[1]
        date = log[2]
        method = log[3]
        url = log[4]
        status = str(log[5])
        details = log[6]
        print(userId,userIP,date,method,url,status)
```

그림 42. 일반 사용자 로그 출력

 사용자 로그를 조회합니다.

| ID | IP | DATE | METHOD | URL | STATUS |
|--------|--------------|------------------------------|--------|--------|--------|
| None | 192.168.52.1 | [26/Apr/2022:04:46:14 +0000] | POST | /login | |
| yoo | 192.168.52.1 | [26/Apr/2022:04:51:36 +0000] | POST | /login | |
| yoo | 192.168.52.1 | [26/Apr/2022:04:51:36 +0000] | POST | /login | |
| yoo | 192.168.52.1 | [26/Apr/2022:04:55:00 +0000] | POST | / HTTP | |
| 1231 | 192.168.52.1 | [26/Apr/2022:04:55:03 +0000] | POST | /login | |
| admin0 | 192.168.52.1 | [26/Apr/2022:04:55:34 +0000] | POST | /log | |
| None | 192.168.52.1 | [26/Apr/2022:05:13:04 +0000] | POST | /login | |
| None | 192.168.52.1 | [26/Apr/2022:05:13:04 +0000] | POST | /login | |
| None | 192.168.52.1 | [26/Apr/2022:05:28:46 +0000] | POST | /inser | |
| None | 192.168.52.1 | [26/Apr/2022:05:28:47 +0000] | POST | /resu | |

나가려면 q, 새로고침하려면 r을 입력하세요

그림 43. 로그 조회 결과

b. 로그인 기능

사용자는 웹 페이지에 접속하여 로그인 할 수 있습니다.

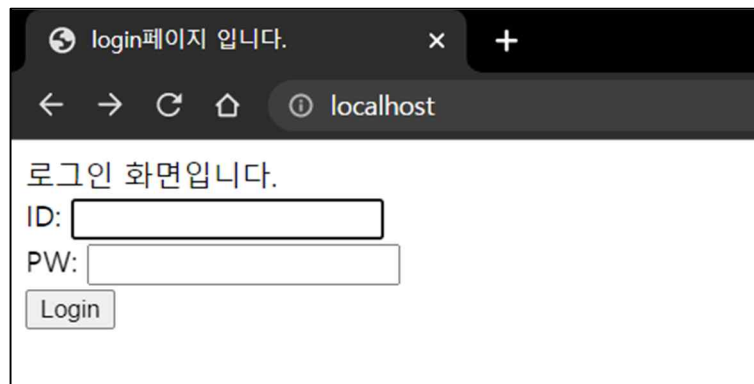


그림 44. 로그인 화면

사용자가 웹 페이지에서 입력한 ID와 Password를 데이터베이스와 연동하여 확인합니다. 이때 페이지에서 받은 패스워드(page_pw)를 MD5 암호화한 뒤 데이터베이스에 저장된 정보와 같은 지 비교합니다.

```
// 페이지에서 파라미터 값 가져오기
String page_id = request.getParameter("id");
String page_pw = request.getParameter("pw");
out.println(page_id);

try{
// SQL 문
String sql = "select * from user where id = ?;";
pstmt = conn.prepareStatement(sql);
pstmt.setString(1, page_id);
rs = pstmt.executeQuery();

int result = 0;
int cnt = 0;
if (rs.next()) {

    // 입력값 암호화
    String originalPasswd = page_pw;
    byte[] bytData = originalPasswd.getBytes();
    md.update(bytData);
    byte[] digest = md.digest();

    String strENCDData = "";
    for(int i =0;i < digest.length;i++){
        strENCDData = strENCDData + Integer.toHexString(digest[i] & 0xFF);
    }
    out.println(strENCDData);

    // passwd 확인 (-MD5 된 암호를 사용해야함)
    if(strENCDData.equals(rs.getString("passwd"))){
        result = 1; // 로그인 성공
    }
}
```

그림 45. login.jsp 내 로그인 정보 확인

로그인 정보가 맞으면 '로그인 성공' 알림창을 띄우고 확인 버튼 클릭 시 login_ok.jsp 페이지로 이동합니다. 아이디 혹은 비밀번호가 틀리면 알림창을 띄워 사용자에게 안내한 뒤 다시 입력을 받습니다.

The image shows three sequential alert dialog boxes. Each box has a title bar with the IP address '192.168.52.133' and the word '내용:' (Content:). The first box displays '로그인 성공' (Login Success) and a blue '확인' (OK) button. The second box displays '존재하지 않는 아이디입니다' (Invalid ID) and a blue '확인' button. The third box displays '비밀번호가 틀립니다' (Invalid Password) and a blue '확인' button.

그림 46. 알림창

c. 웹 콘텐츠 저장

웹 페이지에서 로그인을 성공하면 웹 콘텐츠를 작성할 수 있습니다.

The image shows a web form for registering content. It has a text area with the placeholder 'yoo님, 환영합니다.' (Hello yoo, welcome). Below this is a label '내용' (Content) followed by a large text input field containing 'hello world!'. To the right of the input field is a button labeled '등록' (Register).

그림 47. 웹 콘텐츠 등록 화면

'등록' 버튼을 눌러 POST 요청을 전송하면 작성한 내용이 데이터베이스의 web_contents 테이블에 저장됩니다. JSP 파일 내부에서 기능을 구현하였으며 web_contents의 컬럼은 idx(PK), user_id(유저 아이디), contents(내용)로 이루어져 있습니다.

```
mysql> desc web_contents;
```

| Field | Type | Null | Key | Default | Extra |
|----------|-------------|------|-----|---------|----------------|
| idx | int | NO | PRI | NULL | auto_increment |
| user_id | varchar(32) | YES | MUL | NULL | |
| contents | text | YES | | NULL | |

그림 48. web_contents 테이블

```
<%
request.setCharacterEncoding("UTF-8");

if(request.getMethod().equals("POST")){
    Connection conn = null;
    PreparedStatement pstmt = null;
    String url = "jdbc:mysql://172.33.0.2:3306/cloud?";
    String id = "root"; //MySQL에 접속을 위한 계정의 ID
    String pwd = "abcd"; //MySQL에 접속을 위한 계정의 암호
    Class.forName("com.mysql.cj.jdbc.Driver");
    try{
        conn = DriverManager.getConnection(url, id, pwd);
        String userId = (String) session.getAttribute("id");
        String contents = request.getParameter("contents");
        String query = "INSERT INTO web_contents(user_id, contents) VALUES(?, ?)";
        session.setAttribute("contents", contents);
        pstmt = conn.prepareStatement(query);
        pstmt.setString(1, userId);
        pstmt.setString(2, contents);
        pstmt.executeUpdate();
        pstmt.close();
        conn.close();
    }catch(Exception e){
        out.println(e);
    }
}
```

그림 49. JSP 내 입력 값 DB 저장

5. 프로젝트 산출물

5-1. 실행 화면 캡처

실행화면 캡처는 4-2 Python-CLI 부분의 그림으로 대체합니다.

5-2. 프로젝트 결과에 따른 리소스 비교

클라우드 환경에서 작업에 필요한 리소스를 구성하고 싶은 경우 컴퓨팅 서비스를 이용합니다. 컴퓨팅 서비스의 종류로는 크게 가상머신 컴퓨팅과 컨테이너 컴퓨팅이 있습니다. 가상머신 컴퓨팅에는 가상 프로세서, 메모리, 저장소 및 네트워크 리소스가 포함되며, 운영체제를 직접 호스트합니다. 반면, 컨테이너 컴퓨팅은 가상화 환경이지만 가상머신과는 달리 운영체제를 포함하지 않고, 컨테이너를 실행하는 호스트 환경의 운영체제를 참조합니다.

본 프로젝트에서는 도커 컨테이너를 사용하여 시스템을 구성하였으므로 **도커를 사용한 경우(컨테이너 컴퓨팅)와 도커를 사용하지 않은 경우(가상머신 컴퓨팅)의 리소스를 비교**하겠습니다.

a. 물리적 리소스 비교

- 도커를 사용하지 않은 경우, 시스템 구성

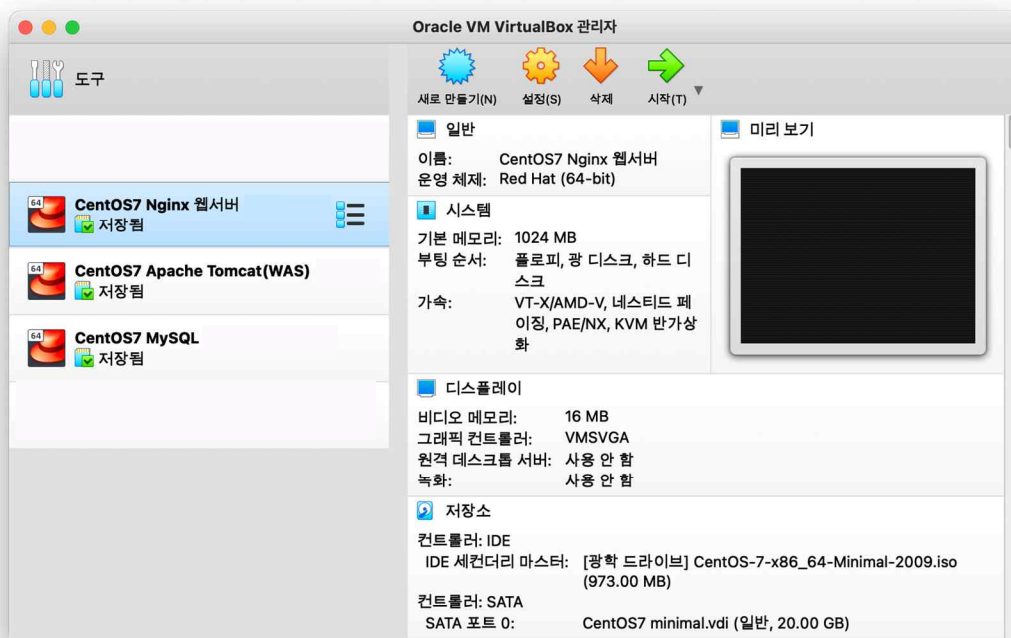


그림 50. 도커를 사용하지 않았을 때 시스템 구성

서버 3 대를 이용하여 nginx, apache tomcat, mysql 을 구성합니다. 기본적으로 각 서버의 메모리는 1024MB, 하드 디스크는 20GB 로 설정하였습니다.

```
[root@localhost ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        484M   0    484M   0% /dev
tmpfs           496M   0    496M   0% /dev/shm
tmpfs           496M  6.8M   489M   2% /run
tmpfs           496M   0    496M   0% /sys/fs/cgroup
/dev/mapper/centos-root 17G   2.0G   16G  12% /
/dev/sda1       1014M  168M   847M  17% /boot
tmpfs           100M   0    100M   0% /run/user/0
```

그림 51. 서버의 디스크 정보

1대의 서버에서 사용된 공간의 크기는 약 2.2GB 이므로 도커를 사용하지 않을 경우의 시스템 리소스는 다음과 같습니다.

$$\text{resource} = 2.2 * 3 = 6.6 \text{ GB}$$

- 도커를 사용한 경우, 시스템 구성

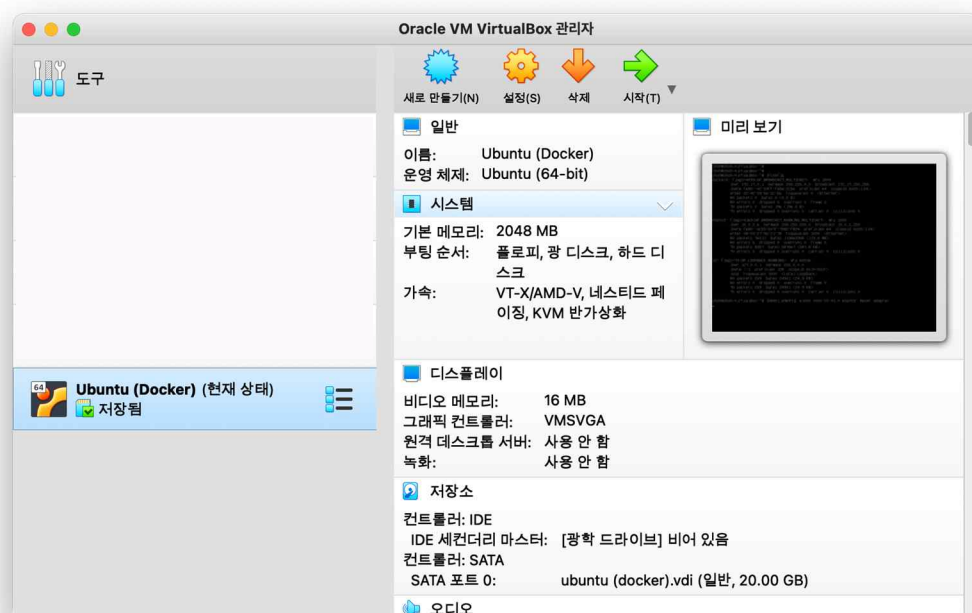
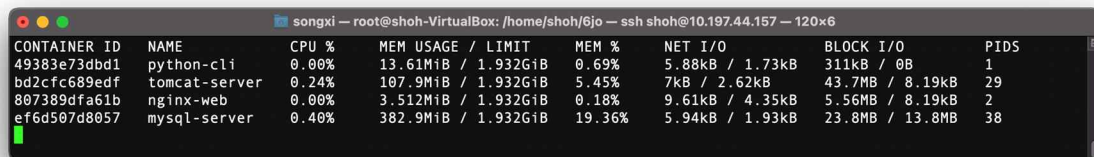


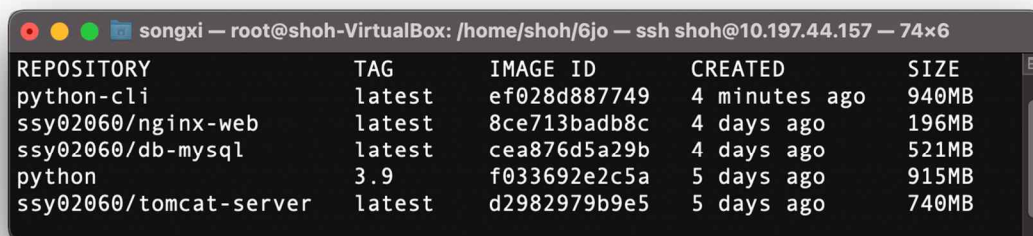
그림 52. 도커를 사용했을 때 시스템 구성

서버 1 대를 이용하여 도커 컨테이너를 구성합니다. 서버의 메모리는 2048MB, 하드 디스크는 20GB 로 설정하였습니다. nginx, apache tomcat, mysql, python-cli 등 다수의 도커 컨테이너가 필요한 환경이므로 docker-compose.yaml 파일에 여러 컨테이너에 필요한 옵션들을 정의함으로써 다수의 컨테이너가 한 번에 세팅하도록 합니다.



| CONTAINER ID | NAME | CPU % | MEM USAGE / LIMIT | MEM % | NET I/O | BLOCK I/O | PIDS |
|--------------|---------------|-------|---------------------|--------|-----------------|-----------------|------|
| 49383e73dbd1 | python-cli | 0.00% | 13.61MiB / 1.932GiB | 0.69% | 5.88kB / 1.73kB | 311kB / 0B | 1 |
| bd2cfc689edf | tomcat-server | 0.24% | 107.9MiB / 1.932GiB | 5.45% | 7kB / 2.62kB | 43.7MB / 8.19kB | 29 |
| 807389dfa61b | nginx-web | 0.00% | 3.512MiB / 1.932GiB | 0.18% | 9.61kB / 4.35kB | 5.56MB / 8.19kB | 2 |
| ef6d507d8057 | mysql-server | 0.40% | 382.9MiB / 1.932GiB | 19.36% | 5.94kB / 1.93kB | 23.8MB / 13.8MB | 38 |

그림 53. 도커 컨테이너 리소스



| REPOSITORY | TAG | IMAGE ID | CREATED | SIZE |
|------------------------|--------|--------------|---------------|-------|
| python-cli | latest | ef028d887749 | 4 minutes ago | 940MB |
| ssy02060/nginx-web | latest | 8ce713badb8c | 4 days ago | 196MB |
| ssy02060/db-mysql | latest | cea876d5a29b | 4 days ago | 521MB |
| python | 3.9 | f033692e2c5a | 5 days ago | 915MB |
| ssy02060/tomcat-server | latest | d2982979b9e5 | 5 days ago | 740MB |

그림 54. 도커 이미지 리소스

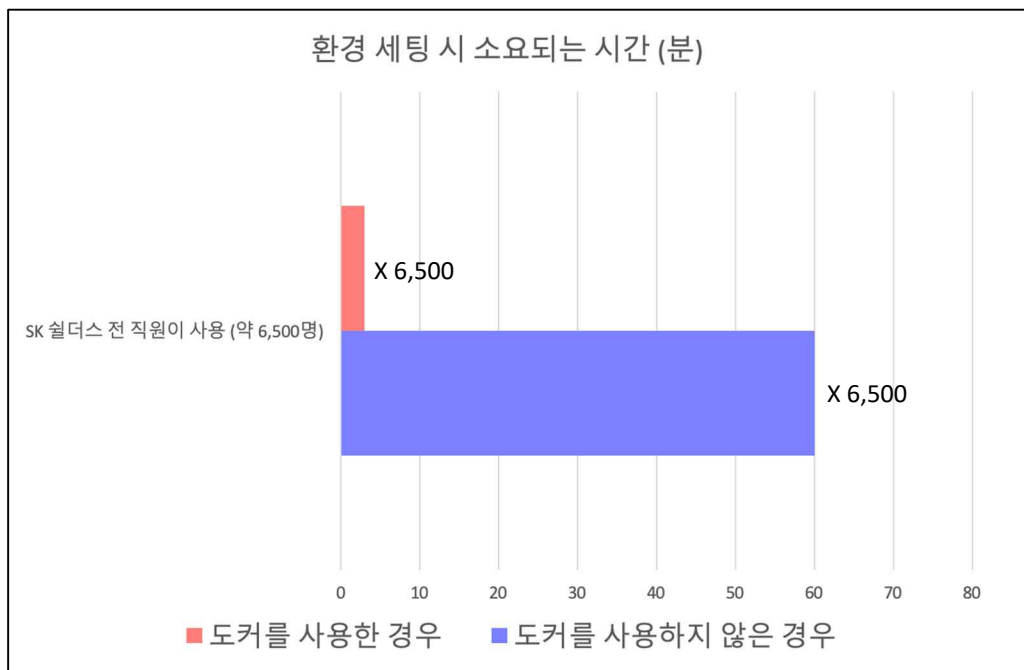
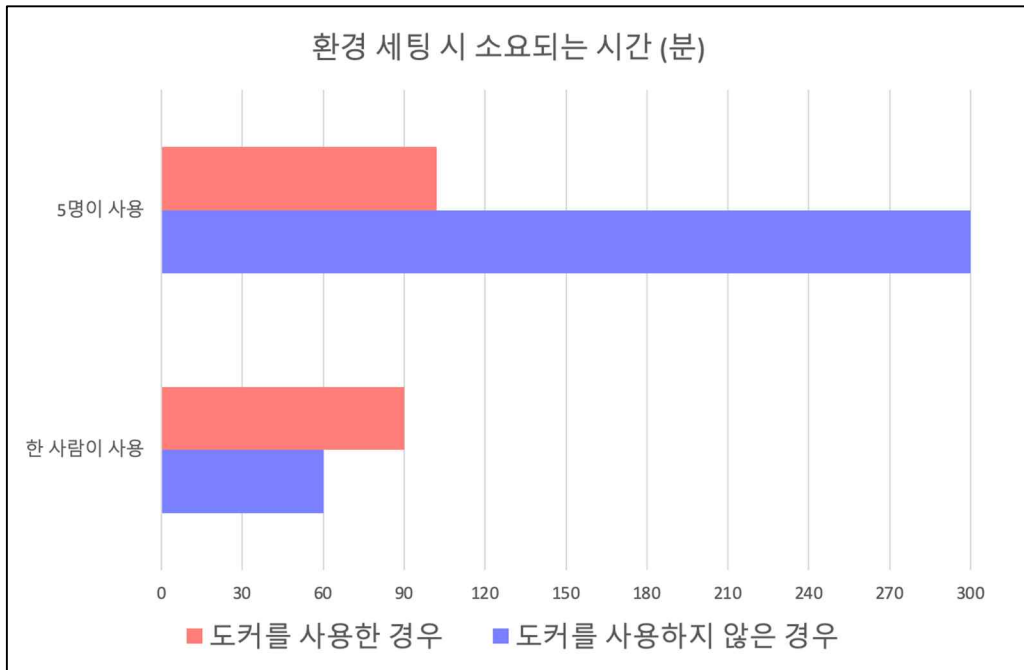
도커 컨테이너의 CPU 사용률은 총 0.64%이고, 리소스 사용량은 0.073GB 입니다. 도커 이미지들의 전체 크기는 약 3.3GB 이므로 도커를 사용한 경우의 시스템 리소스는 다음과 같습니다.

$$\text{resource} = 0.073 + 3.3 = 3.37 \text{ GB}$$

물리적 리소스 비교 결과, 컨테이너를 통해 시스템을 구성하는 것이 리소스를 줄일 수 있고, 동적 생성 및 확장이 용이하다는 것을 확인할 수 있습니다.

b. 환경 세팅 시 소요되는 시간 비교

도커를 사용하는 경우 서버에 저장된 컨테이너는 쉽게 공유 및 불러오기가 가능하며, 도커 컨테이너를 생성할 때 사용되는 명령어를 dockerfile 에 정의함으로써 개발 환경 세팅을 코드로 손쉽게 관리할 수 있습니다. 따라서 한 사람이 도커를 통해 환경을 세팅한 후, 다른 사람들이 설정된 환경을 불러오면 동일한 환경으로 세팅이 되므로 약 3 분이라는 시간이 소요될 것으로 가정하였습니다.



환경 세팅 시 소요되는 시간 비교 결과, 컨테이너를 통해 시스템을 구성하는 것이 환경을 세팅하는 데 소요되는 시간을 줄일 수 있습니다.

5-3. 프로젝트 진행 소감

| | |
|-----|---|
| 서상윤 | 클라우드 기술에 핵심 중 하나인 컨테이너 가상화 기술을 도커를 통해 경험해 볼 수 있는 유익한 시간이었습니다. 또한 팀원들과 프로젝트를 진행해보니 평소에 혼자 공부할 때는 느낄 수 없었던 것들을 느낄 수 있어서 좋았습니다. |
| 오송희 | 도커 기반의 시스템을 직접 구성하고 관리해보며 클라우드에 대해 주도적으로 학습할 수 있는 좋은 경험이었습니다. 팀원들과 함께 문제를 해결해 나가는 과정에서 많은 것을 배울 수 있었고, 함께해서 더 의미 있는 프로젝트였습니다. |
| 유동섭 | 컨테이너 기술의 대표적인 도커를 이용하여 여러 서비스를 구동해 볼 수 있는 좋은 경험이 되었습니다. 팀 과제를 통해 효율, 효과적으로 프로젝트를 끝낼 수 있어서 즐거웠습니다. 앞으로 더욱 많은 지식을 습득하여 보안 전문가로 거듭나고 싶습니다. |
| 한승훈 | 클라우드 기술 가상화와 컨테이너에 대해서 잘 알아볼 수 있는 시간이었고 많이 부족했지만 좋은 팀원들을 만나 더 배우는 시간이기도 했습니다. 이 경험을 발판 삼아 더 열심히 배우고 싶습니다. |