#Summary

MusicApp은 음악 어플이다. 사용자로는 크게 user와 manager가 있다. 우선, 처음 접속했을 때는 main menu가 나온다. user로서 어플을 이용할 지, manager로서 어플을 이용할 지 선택한 다음, 각 사용자 권한에 맞는 menu들을 선택할 수 있다.

user menu에서는 새로운 user를 등록하거나, 기존의 user로서 로그인 할 수 있다.

user로서 로그인을 하면, [Chart 보기, Music 보기, 본인의 PlayList 보기, 보유 Coin 보기, Coin 충전하기, PlayList에 새 음악 추가하기, 삭제하기, 본인의 정보 보기, 탈퇴하기] 기능을 이용할 수 있다.

manager menu에서는 기존의 manager로 로그인할 수 있다.

manager로서 로그인을 하면, [Music 보기, 새로운 Music 등록하기, Music 삭제하기, 사용자 보기, 사용자 관리] 기능을 이용할 수 있다.

manager은 5명으로 고정되어 있으며, 한 사람 당 하나의 genre에 해당하는 음악을 관리하고 (GENRE배열에서 manager_num-1 index) , 자신의 manager_num % 5의 값을 가진 user를 관리한다. (manager_num%5값이 0이라면 5번 manager가 관리)

모든 메뉴에서 이전 메뉴로 돌아갈 수 있으며, main menu에서 Exit을 하면 프로그램이 종료된다.

#Code Description

총 12개의 method로 구성되어 있다.

#ini_USERNUM( )

```
String query = "select MAX(user_num) from user_";
        ...
if (rs.next()) {
    USER_NUM = rs.getInt(1);
}
```

ini_USERNUM( )은 프로그램을 처음 시작할 때, MusicApp의 데이터베이스에 저장된 user의 user_num 중 가장 큰 값을 불러와 static 변수 `USER_NUM`에 저장한다. 이는 새로운 user을 등록할 때 user_num을 오류 없이 설정하는 데에 필요하다.

> select MAX(user_num) from user_
>
> -> user_ table에서 user_num 중 최댓값을 가져온다.

#ini_REGISTERNUM( )

```
String query = "select MAX(register_num) from music";
        ...
if (rs.next()) {
    REGISTER_NUM = rs.getInt(1);
}
```

ini_USERNUM( )은 프로그램을 처음 시작할 때, MusicApp의 데이터베이스에 저장된 music의 register_num 중 가장 큰 값을 불러와 static 변수 `REGISTER_NUM`에 저장한다. 이는 새로운 music을 등록할 때 register_num을 오류 없이 설정하는 데에 필요하다.

> select MAX(register_num) from music
>
> -> music table에서 register_num 중 최댓값을 가져온다.

#music_cnt( )

```
String query1 = "select chart.music_cnt from chart where chart.chart_name='TopChart'";
String query2 = "update chart set music_cnt=?";
        ...
if (rs.next()) {
    originCnt = rs.getInt(1);
}
pstmt.setInt(1, originCnt+1);
```

music_cnt( )는 chart에 몇개의 music이 진입했는지를 update한다. user가 playlist에 새로운 음악을 추가할 경우, 추가하기 전 그 음악의 score가 0이었을 때 호출한다. chart의 현재 music_cnt를 불러와 `originCnt`에 저장한 다음, 그 값에 1을 더하여 update한다.

> select MAX(user_Num) from user_
>
> -> user_ table에서 user_num중 최댓값을 가져온다.

#startMenu( )

```
System.out.println("0. Exit");
System.out.println("1. User Menu");
System.out.println("2. Manager Menu");
System.out.print("Input: ");
input = scan.nextInt();
```

startMenu( )는 어플을 처음 실행했을 때 나오는 MainMenu 이다. 선택지를 출력한 다음, 사용자에게 번호를 입력받아 각 번호에 해당하는 함수를 실행한다. 만약 선택지에 없는 번호를 입력할 경우, 에러메시지를 출력하며 startMenu( ) 함수를 재호출한다.

#userMenu( )

```
System.out.println("0. Return to Main Menu");
System.out.println("1. Register New User");
System.out.println("2. Log in as User");
System.out.print("Input: ");
```

userMenu( )는 startMenu에서 "1. User Menu"를 선택했을 경우의 선택지를 보여준다. 사용자에게 번호를 입력받아 각 번호에 해당하는 함수를 실행한다. 0번을 입력한다면, 다시 startMenu( )로 돌아가며, 나머지의 경우 새로운 user을 등록하거나, 기존의 user로 로그인한다. 만약 선택지에 없는 번호를 입력할 경우, 에러메시지를 출력하며 userMenu( ) 함수를 재호출한다.

#user_reg( )

```
System.out.print("Name: ");
name = scan.next();
        ...
coin = 0;
user_num = ++USER_NUM;
manager_num = user_num%5;
if (user_num%5==0) manager_num = 5;

String query = "insert into user_(name_, sex, bdate, address, email, id, password_,
user_num, coin, manager_num) values(?,?,?,?,?,?,?,?,?,?)";
pstmt = conn.prepareStatement(query);

pstmt.setString(1, name);
        ...
String query = "insert into playlist(user_num, playlist_num) values(?,?)";

pstmt = conn.prepareStatement(query);
pstmt.setInt(1, user_num);
        ...
```

user_reg( )는 userMenu( )에서 "1. Register new User"을 선택했을 경우 실행된다. 새로 등록할 사용자의 정보를 입력받고, `coin`은 0으로 초기화, `user_num`은 기존 user_num중 가장 큰 값+1 로 초기화한다. user들을 5명의 manager가 관리하는 구조이기 때문에, `manager_num`은 `user_num`%5로 초기화하고 만약 `user_num`이 5로 나누어 떨어지면 `manager_num`을 5로 설정한다.

앞에서 초기화한 값들을 user의 attribute value로 넣어 user_에 insert하고, 메시지를 출력한다. 또한 user가 등록되면 자동으로 해당 user의 playlist가 만들어지도록 설계하였기 때문에, `user_num`과 `playlist_num`을 현재 등록한 user_num으로 설정하여 playlist에 insert한다.

> insert into user*(name*, sex, bdate, address, email, id, password_, user_num, coin, manager_num) values(?,?,?,?,?,?,?,?,?,?)
>
> -> user에 입력값을 attribute value로 하여 새로운 tuple을 insert한다.
>
> String query = "insert into playlist(user_num, playlist_num) values(?,?)";
>
> -> playlist에 입력값을 attribute value로 하여 새로운 tuple을 insert한다.

#user_log( )

```
System.out.print("User_num: ");
user_num = scan.nextInt();
System.out.print("ID: ");
id = scan.next();
System.out.print("Password: ");
password = scan.next();
        ...
String query = "select user_.name_, user_.sex, user_.bdate, user_.address, user_.email,
user_.id, user_.password_, user_.user_num, user_.coin, user_.manager_num from user_ where
user_.user_num="+user_num
                + " and user_.id='"+id+"' and user_.password_='"+password+"'";
        ...
if (rs.next()) {
    name = rs.getString(1);
    sex = rs.getString(2);
        ...
```

user_log( )는 userMenu( )에서 "2. Log in as User"를 선택했을 경우 실행된다. user_num, id, password를 입력받고, 그 값들에 해당하는 tuple을 찾아 각각의 속성을 static 변수에 할당하여 현재 로그인 한 정보로 어플을 이용할 수 있게 해준다. 로그인이 완료되면, 메시지를 출력하고 만약 해당하는 사용자가 없다면 에러 메시지를 출력한다.

> select user.*name*, user.*sex, user*.bdate, user.*address, user*.email, user.*id, user*.password*, user*.user_num, user.*coin, user*.manager_num from user_ where user.*user_num="+user_num" and user*.id='"+id+"' and user.*password*='"+password+"' '"
>
> -> 입력받은 user_num, id, password와 일치하는 tuple을 user_에서 select한다.

#after_userlog( )

```
System.out.println("0. Log out");
System.out.println("1. View Chart");
System.out.println("2. View Music");
System.out.println("3. View Playlist");
System.out.println("4. View Coin");
System.out.println("5. Charge Coin");
System.out.println("6. Add new music to Playlist");
System.out.println("7. Delete music from Playlist");
System.out.println("8. View User Information");
System.out.println("9. Withdraw from App");
System.out.print("Input: ");
input = scan.nextInt();
        ...
switch(input) {
    case 0:
        userMenu();
        return;
    case 1:
        String query = "select music.title, ..."
            ...
```

```
}
```

after_userlog( )는 성공적인 user_log( )이후 실행된다. [log out, view chart, view music, view playlist, view coin, charge coin, add new music to playlist, delete music from playlist, view user information, withdraw from app]의 선택지가 있고, 사용자에게 번호를 입력받는다.

0. 이전의 메뉴(userMenu( ))로 돌아간다.
1. 차트를 확인한다. charscore가 0이 아니면 chart에 진입해 있으므로, 그 조건을 만족하는 모든 music의 title, artist, chartscore를 score가 높은 순서대로 출력한다. 모든 실행이 끝난 후 다시 after_userlog( )를 호출한다.
2. 등록된 모든 음악의 title, artist를 확인한다. `cnt`가 0이라면, 즉 만약 등록된 음악이 없다면, after_userlog( )를 호출한다. 등록된 음악이 하나 이상일 경우에는, 음악의 세부적인 정보를 볼 수 있도록 메시지를 출력한다. 만약 원하지 않는다면 0을 입력하도록 하고 after_userlog( )를 호출하고, title을 입력하면 해당 music의 장르, 앨범명 등을 출력한 후 after_userlog( )를 호출하며 종료한다.
3. 로그인한 user의 playlist를 확인한다. playlist에 있는 모든 음악의 title, artist를 출력한다.
4. 로그인한 user의 coin을 확인한다. 현재 user_num을 가진 user를 찾아, 그 user에 해당하는 coin을 출력한다.
5. 로그인한 user의 coin을 충전한다. 사용자로부터 충전하고 싶은 coin을 입력받아 `wantCoin`에 저장하고, 해당 사용자의 원래 coin을 `originCoin`에 저장한다. 그리고 `originCoin`에 사용자가 입력한 만큼의 coin을 더하여 사용자의 coin을 update함과 동시에 static 변수 coin을 그와 같은 값으로 설정하여 다른 작업 수행 시 query문을 반복하지 않도록 한다.
6. 로그인한 user의 playlist에 음악을 추가한다. 한 곡당 1000원을 지불해야 추가할 수 있으므로, 만약 가진 coin이 1000 미만이라면, 에러메시지를 출력하고 after_userlog( )를 호출하며 종료한다. 그렇지 않으면, 어플에 존재하는 모든 음악의 title, artist, register number을 출력하여 선택이 편리하도록 하고, 사용자로부터 추가를 원하는 음악의 register number을 입력받아 `mnumber`에 저장한다. `mnumber`에 해당하는 음악과 사용자의 정보를 attribute value로 하여 reg_in_pla와 buy_music에 insert하고, 메시지를 출력한다. 이미 playlist에 있는 음악을 중복하여 추가하려고 한다면, 기존의 음악에 덮어쓴다.

   user가 음악을 구매하면, 해당 음악의 chartscore가 올라간다. 따라서 해당 음악의 원래 chartscore을 `originScore`에 저장한다. 만약 `originScore`가 0이면 chart에 새로운 음악이 진입했음을 표시하기 위해 앞서 정의한 music_cnt( )를 호출하여 메시지를 출력한다. 그리고 해당 음악의 chartscore을 `originScore`+1로 update해준다.

   user가 음악을 구매하면, 해당 user의 coin이 한 곡당 1000 감소한다. 따라서 해당 user의 원래 coin을 `OriginCoin`에 저장하고, user의 coin을 `OriginCoin`-1000으로 update하고 static 변수 `coin`을 같은 값으로 설정하며 메시지를 출력한다.

   모든 실행이 끝난 후, after_userlog( )를 호출하며 종료한다.
7. 로그인한 user의 playlist에서 음악을 delete한다. 먼저 user의 playlist에 있는 모든 음악의 정보를 register_num을 포함하여 출력한다. 만약 음악이 없다면, 에러메시지를 출력한다. 그렇지 않다면 사용자로부터 삭제할 음악의 register_num을 입력받아 `mnumber`에 저장하고, `mnumber`에 해당하는 register_num을 가진 음악을 delete하며 메시지를 출력하고, after_userlog( )를 호출하며 종료한다.
8. 로그인한 user의 정보를 모두 출력한다. 현재 static 변수 `user_num`이 로그인한 user의 user_num로 설정되어 있으므로, `user_num`에 해당하는 user의 이름, 성별 등을 모두 출력하고 after_userlog( )를 호출하며 종료한다.
9. 로그인한 user을 어플에서 탈퇴시킨다. 한번 더 확인을 위해 id와 user_num을 입력받아 각각 `usrid`와 `usrnum`에 저장한다. 그리고 그 두가지에 모두 해당하는 user을 찾아 delete하고 메시지를 출력한다. user가 탈퇴했으므로 더이상 로그인후의 서비스를 이용할 수 없어, userMenu( )를 호출하며 종료한다.

존재하지 않는 선택지가 입력된다면, 에러메시지를 출력하고 after_userlog( )를 호출한다.

"select music.title, music.artist, music.chartscore from chart, music where not music.chartscore=0 order by music.chartscore desc

-> chart에 진입한 음악을 보기 위해 chartscore가 0이 아닌 음악들의 title, artist, chartscore을 chartscore가 높은 순서 음악부터 select한다.

select music.title, music.artist from music

-> music table에서 모든 음악의 title과 artist를 select한다.

"select music.genre, music.album, music.release_date, music.lyricist, music.composer, music.title, music.artist, music.chartscore, music.price, music.register_num " + "from music where music.title='"+mtitle+"'

->mtitle에 해당하는 음악의 장르, 앨범, 등등을 music table에서 select한다.

"select music.title, music.artist from reg_in_pla, music where reg_in_pla.user_num="+user_num+" and reg_in_pla.register_num=music.register_num";

-> reg_in_pla와 music table로부터 reg_in_pla의 user_num과 현재 user_num이 같고, reg_in_pla의 resgiter_num과 music의 register_num이 같은 tuple에서 music의 title, artist를 select한다.

select user.*coin from user* where user_.user_num="+user_num

-> user table에서 현재 user_num과 user_의 user_num이 같은 것을 찾아 coin을 select한다.

"select user.*coin from user* where user.*user_num="+user_num "update user* set coin=? where user_num="+user_num

-> 현재 user_num과 같은 user_num을 가진 user의 coin을 select하고, 그 user의 coin을 update한다.

select music.title, music.artist, music.register_num from music

-> music table로부터 모든 음악의 title, artist, register_num을 select한다.

"insert into reg_in_pla (register_num, user_num, playlist_num) values ("+mnumber+", "+user_num+", "+user_num +") on duplicate key update reg_in_pla.register_num= "mnumber+", reg_in_pla.user_num="+user_num+", reg_in_pla.playlist_num="+user_num

-> reg_in_pla table에 중복된 register_num, user_num key값이 들어가면 update를 해주며 insert한다.

"insert into buy_music (register_num, user_num) values ("+mnumber+", "+user_num+") on duplicate key update buy_music.register_num= "mnumber+", buy_music.user_num="+user_num

-> buy_music에 중복된 register_num, user_num key 값이 들어가면 update를 해주며 insert한다.

"select music.chartscore from music where music.register_num="+mnumber;

"update music set chartscore=? where music.register_num="+mnumber;

-> 입력받은 mnumber와 같은 register_num을 가진 음악의 chartsocre을 select하고, ?값으로 update한다.

"select user.*coin from user* where user.*user_num="+user_num "update user* set coin=? where user_.user_num="+user_num

-> 현재 user_num과 같은 user_num을 가진 user의 coin을 select하고 ?값으로 update한다.

"select music.title, music.artist, music.register_num from music, reg_in_pla where reg_in_pla.user_num="+user_num+" and music.register_num= reg_in_pla.register_num"

> -> 현재 usr_num과 reg_in_pla table의 user_num이 같고 music table의 register_num과 reg_in_pla의 register_num이 같은 tuple의 title, artist, register_num을 select한다.
>
> delete from reg_in_pla where reg_in_pla.register_num="+mnumber+" and reg_in_pla.user_num="+user_num
>
> -> 입력받은 mnumber와 reg_in_pla table의 register_num이 같고 현재 user_num과 reg_in_pla table의 user_num이 같은 tuple을 reg_in_pla에서 delete한다.
>
> select user.*name*, user.*sex, user*.bdate, user.*address, user*.email,"+"user.*id, user*.user_num, user.*manager_num from user* where user_.user_num="+user_num
>
> -> 현재 user_num과 같은 user_num을 가진 user의 모든 정보를 select한다.
>
> "delete from user_ where user.*id='"+usrid+"' and user*.user_num="+usrnum
>
> -> 입력받은 usrnum, usrid와 같은 user_num, id를 가진 user을 user_ table로부터 delete한다.

#managerMenu( )

```
System.out.println("0. Return to Main Menu");
System.out.println("1. Log in as Manager");
System.out.print("Input: ");
```

managerMenu( )는 startMenu에서 "1. Manager Menu"를 선택했을 경우의 선택지를 보여준다. 사용자에게 번호를 입력받아 각 번호에 해당하는 함수를 실행한다. 0번을 입력한다면, 다시 startMenu( )로 돌아가며, 1번을 입력한다면 manager로 로그인한다. (manager은 새로 가입할 수 없다) 만약 선택지에 없는 번호를 입력할 경우, 에러메시지를 출력하며 managerMenu( ) 함수를 재호출한다.

#manager_log( )

```
System.out.print("Manager_num: ");
manager_num = scan.nextInt();
System.out.print("RRN: ");
rrn = scan.next();
        ...
String query = "select manager.name_, manager.rrn, manager.sex, manager.bdate,
manager.manager_num from manager where manager.manager_num="+manager_num+" and
manager.rrn='"+rrn+"'";
                ...
if (rs.next()) {
  if(rs.next()) {
      name = rs.getString(1);
      rrn = rs.getString(2);
        ...
```

manager_log( )는 managerMenu( )에서 "1. Log in as Manager"를 선택했을 경우 실행된다. manager_num, rrn을 입력받고, 그 값들에 해당하는 tuple을 찾아 각각의 속성을 static 변수에 할당하여 after_managerlog( )를 호출하여 현재 로그인 한 정보로 manager로서 어플을 이용할 수 있게 해준다. 로그인이 완료되면, 메시지를 출력하고 만약 해당하는 관리자가 없다면 에러 메시지를 출력하고 managerMenu( )를 다시 호출하여 되돌아간다.

> "select manager.name_, manager.rrn, manager.sex, manager.bdate, manager.manager_num from manager where manager.manager_num="+manager_num+" and manager.rrn='"+rrn+"'"
>
> -> 입력받은 manager_num, rrn과 같은 manager_num, rrn 값을 가진 manager의 정보를 select한다.

#after_managerlog( )

```
System.out.println("0. Log out");
System.out.println("1. View Music");
System.out.println("2. Add new music to MusicList");
System.out.println("3. Delete music from MusicList");
System.out.println("4. View Users");
System.out.println("5. Manage Users");
System.out.print("Input: ");
input = scan.nextInt();
        ...
switch(input) {
    case 0:
        managerMenu();
        return;
    case 1:
        String query = "select music.title, ..."
            ...
}
```

after_managerlog( )는 성공적인 manager_log( )이후 실행된다. [log out, view music, add new music to MusicList, delete music from MusicList, view users, Manage users] 의 선택지가 있고, 사용자에게 번호를 입력받는다.

0. 이전의 메뉴(managerMenu( ))로 돌아간다.
1. 등록된 모든 음악의 title, artist를 확인한다. `cnt`가 0이라면, 즉 만약 등록된 음악이 없다면, after_userlog( )를 호출한다. 등록된 음악이 하나 이상일 경우에는, 음악의 세부적인 정보를 볼 수 있도록 메시지를 출력한다. 만약 원하지 않는다면 0을 입력하도록 하고 after_userlog( )를 호출하고, title을 입력하면 해당 music의 장르, 앨범명 등을 출력한 후 after_userlog( )를 호출하며 종료한다.
2. 새로운 music을 등록한다. 먼저 현재 로그인한 관리자가 해당 음악을 등록할 자격이 있는 지 체크한다. 장르를 번호로 입력받아, 장르 번호와 manager_num이 같다면, `genre`에 해당 번호에 해당하는 장르를 문자열로 저장한다. 그리고 나머지 attribute value들도 입력받고, `price`는 1000, `register_num`에 `REGISER_NUM`에서 1증가시킨 값을 저장하고 `manager_num`에는 현재 log in하며 설정된 manager_num을 다시한번 저장, `chart_name`을 "TopChart"로 설정한다. 그리고 그 정보들을 attribute value로 하여 music에 insert한다. 장르 번호와 manager_num이 같지 않다면, 메시지를 출력하고 다시 after_managerlog( )를 호출하며 종료한다.
3. music을 삭제한다. 우선 모든 음악들을 출력한다. 만약 기존에 music이 존재하지 않는다면, 메시지를 출력하고 after_managerlog( )를 호출하며 종료한다. 존재한다면, 삭제할 음악을 register_num으로 입력받는다. register_num에 해당하는 음악의 장르를 받아, tmpgen에 저장한 후 현재 manager_num의 관리 하에 있는 장르와 같은지 비교한다. 만약 같지 않다면, 권한이 없다는 메시지를 출력한 후 after_managerlog( )를 호출하며 종료한다. 같다면, register_num에 해당하는 음악을 delete한다.

4. 등록되어 있는 모든 user의 name, id, user_num을 보여준다. 만약 user가 존재하지 않는다면, 메시지를 출력한다. 그리고 after_managerlog( )를 호출하며 종료한다.
5. 회원을 강제로 탈퇴시킨다. 우선 등록되어 있는 모든 user을 출력한다. user가 존재한다면, delete할 user의 user_num을 입력받아 `input`에 저장한다. 만약 `input`%5가 0이라면, input을 0으로 초기화시켜 아래의 조건문에서 input%5를 manager_num과 비교하기 편하도록 한다. 만약 input%5가 manager_num과 같지 않다면, 권한이 없다는 메시지를 출력한 후 after_managerlog( )를 호출하며 종료한다. 같다면, user_num에 해당하는 user을 delete하고 메시지를 출력한다. 그리고 after_managerlog( )를 호출하며 종료한다.

> "select music.title, music.artist from music"
>
> -> music table로부터 모든 음악의 title, artist를 select한다.
>
> "select music.genre, music.album, music.release_date, music.lyricist, music.composer, music.title, music.artist, music.chartscore, music.price, music.register_num " +"from music where music.title='"+mtitle+"'"
>
> -> music table로부터 사용자에게 입력받은 mtitle에 해당하는 음악의 모든 속성을 select한다.
>
> "insert into music(genre, album, release_date, lyricist, composer, title, artist, chartscore, price, register_num, manager_num, chart_name) values(?,?,?,?,?,?,?,?,?,?,?,?)"
>
> 입력받은 값을 attribute value로 하여 music table에 insert한다.
>
> "select music.title, music.artist, music.genre, music.register_num from music";
>
> -> music table로부터 모든 음악의 title, artist, genre, register_num을 select한다.
>
> "select music.genre from music where music.register_num="+mnumber;
>
> -> 입력받은 mnumber에 해당하는 register_num을 가진 음악의 genre를 select한다.
>
> "delete from music where music.register_num="+mnumber
>
> -> 입력받은 mnumber에 해당하는 register_num을 가진 음악을 music table에서 delete한다.
>
> "select user.*name*, user.*id, user*.user_num from user_"
>
> -> user_ table에서 모든 user의 id, user_num을 select한다.
>
> "delete from user_ where user_.user_num = "+input
>
> -> input과 같은 user_num을 가진 user를 user_ table에서 delete한다.

#main( )

```
try {
    Class.forName("org.mariadb.jdbc.Driver");
    String url = "jdbc:mariadb://localhost:3306/MusicApp";
    String user = "db";
    String psw = "db!";
    conn = DriverManager.getConnection(url, user, psw);
} catch (ClassNotFoundException e) {
    e.printStackTrace();
}

ini_USERNUM();
ini_REGISTERNUM();
startMenu();
```

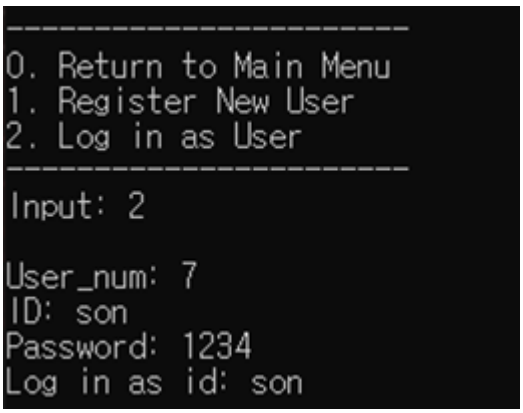
main( )에서는 mariadb와 연결한다. 그리고 ini_USERNUM( ), ini_REGISERNUM( )을 호출하여 전역변수를 초기화시키고, startMenu( )를 호출하여 어플을 실행한다.

#Screenshot

* MainMenu

* MainMenu에서 1 선택

* UserMenu에서 1 선택, 새로운 회원 가입



* UserMenu에서 2 선택, 기존 회원으로 로그인

* Log in 후 1 선택, chart 보기

```
9. Withdraw from App
-----------------------------
Input: 2

-----------------------------------
Title                 Artist
-----------------------------------
this is first song    cc
this is second song   ff
this is third song    ii
this is fourth song   ll
this is fifth song    oo
this is sixth song    rr
this is seventh song  uu
this is eighth song   xx
-----------------------------------

To see more Detailed information, enter the title. If not, enter '0'
Input: this is first song
Genre: ballade
Album: new
Release_date: 2013-03-24
Lyricist: aa
Composer: bb
Title: this is first song
Artist: cc
ChartScore: 3
Price: 1000
Register_num : 1
```

* Log in 후 2 선택, 모든 음악 보기

```
8. View User Information
9. Withdraw from App
-----------------------------
Input: 3

-----------------------------------
Title                 Artist
-----------------------------------
this is first song    cc
-----------------------------------
```

* Log in 후 3 선택, 내 playlist 보기

```
7. Delete music from Playlist
8. View User Information
9. Withdraw from App
-----------------------------
Input: 4

Coin: 0
```

* Log in 후 4 선택, 보유 Coin 확인하기

```
8. View User Information
9. Withdraw from App
-----------------------------
Input: 5

How much do you want to charge?
1000
Coin successfully Charged!
```

* Log in 후 5 선택, Coin 충전하기

```
9. Withdraw from App
------------------------------
Input: 6

-------------------------------------------------------------------
Title                   Artist                  Register number
-------------------------------------------------------------------
this is first song      cc                      1
this is second song     ff                      2
this is third song      ii                      3
this is fourth song     ll                      4
this is fifth song      oo                      5
this is sixth song      rr                      6
this is seventh song    uu                      7
this is eighth song     xx                      8
-------------------------------------------------------------------

Select Music by Register Number to add to Playlist
input: 2

Successfully Added

number 2 music's score up
son's coin decreased
```

* Log in 후 6 선택, 음악 구매하기

```
7. Delete music from Playlist
8. View User Information
9. Withdraw from App
------------------------------
Input: 6

Not enough Coin, you can't buy Music
```

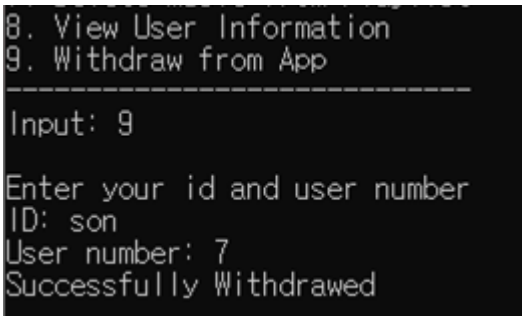
+ 가진 coin이 1000미만일 경우 구매 불가

```
8. View User Information
9. Withdraw from App
------------------------------
Input: 7

-------------------------------------------------------------------
Title                   Artist                  Register number
-------------------------------------------------------------------
this is first song      cc                      1

this is second song     ff                      2

-------------------------------------------------------------------
Select music to delete by Register number: 1
Successfully Deleted
```

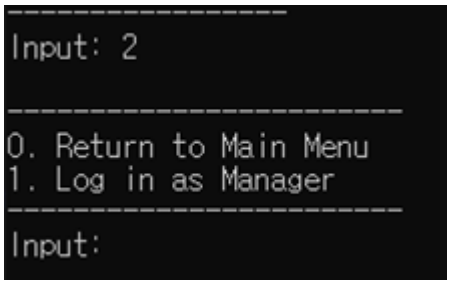
* Log in 후 7 선택, 음악 삭제

```
8. View User Information
9. Withdraw from App
------------------------------
Input: 8

name: son
sex: m
birth date: 1989-03-23
address: seoul
email: son@naver.com
id: son
user number: 7
manager number: 2
```

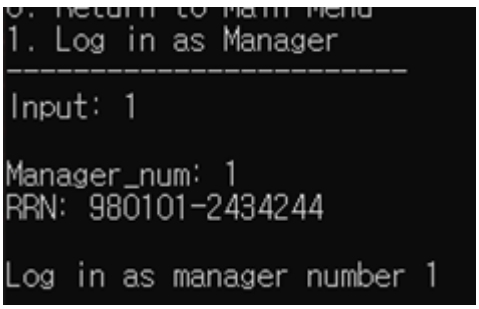
* Log in 후 8 선택, 내 정보 확인



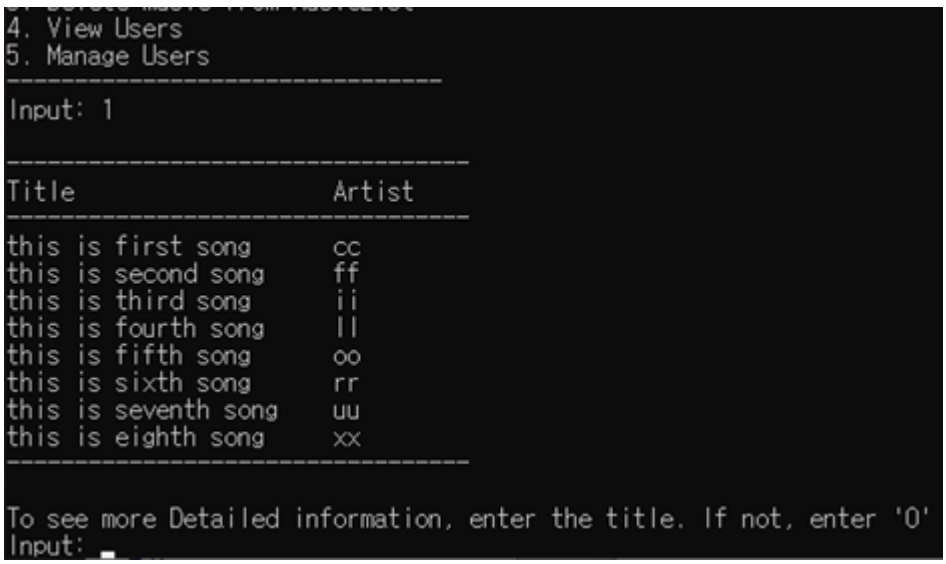
* Log in 후 9 선택, 탈퇴



* MainMenu에서 2 선택



* ManagerMenu에서 1 선택해서 Log in



* Log in 후 1 선택, 음악 보기

```
4. View Users
5. Manage Users
---------------------------------
Input: 2

Genre
1. ballade
2. dance
3. classic
4. agitation
5. etc
:1

Album: new album
Release_date: 2019-12-03
Title: new title
Artist: new artist
Lyricist: new lyricist
Composer: new composer
Successfully Added
```

* Log in 후 2 선택, 음악 추가

```
5. Manage Users
---------------------------------
Input: 2

Genre
1. ballade
2. dance
3. classic
4. agitation
5. etc
Input: 2

No permissions, you can only add music of ballade
Return to previous menu
```

+ 권한이 없는 장르 선택 시

```
5. Manage Users
---------------------------------
Input: 3

-------------------------------------------------------------------------------------------
Title                   Artist                  Genre                   Register number
-------------------------------------------------------------------------------------------
this is first song      cc                      ballade                 1
this is second song     ff                      dance                   2
this is third song      ii                      classic                 3
this is fourth song     ll                      agitation               4
this is fifth song      oo                      etc                     5
this is sixth song      rr                      dance                   6
this is seventh song    uu                      etc                     7
this is eighth song     xx                      ballade                 8
new title               new artist              ballade                 10
-------------------------------------------------------------------------------------------

Select music to delete by Register number
Input: 10

Successfully Deleted
```

* Log in 후 3 선택, 음악 삭제

```
5. Manage Users
--------------------------------
Input: 3

-------------------------------------------------------------------------------
Title                Artist               Genre                Register number
-------------------------------------------------------------------------------
this is first song   cc                   ballade              1
this is second song  ff                   dance                2
this is third song   ii                   classic              3
this is fourth song  ll                   agitation            4
this is fifth song   oo                   etc                  5
this is sixth song   rr                   dance                6
this is seventh song uu                   etc                  7
this is eighth song  xx                   ballade              8
-------------------------------------------------------------------------------

Select music to delete by Register number
Input: 2

No permissions, you can only delete music of ballade
```

+ 권한이 없는 장르 선택 시

```
4. View Users
5. Manage Users
--------------------------------
Input: 4

-----------------------------------------------------------------
Name                 Id                   User Number
-----------------------------------------------------------------
park                 park                 1
kim                  kim                  2
choi                 choi                 3
kang                 kang                 4
lee                  lee                  5
han                  han                  6
-----------------------------------------------------------------
```

* Log in 후 4 선택, 모든 user 보기

```
--------------------------------
Input: 5

-----------------------------------------------------------------
Name                 Id                   User Number
-----------------------------------------------------------------
park                 park                 1
kim                  kim                  2
choi                 choi                 3
kang                 kang                 4
lee                  lee                  5
han                  han                  6
-----------------------------------------------------------------

Select user by user number to delete
Input: 6
Successfully Deleted
```

* Log in 후 5 선택, user 강퇴

```
kang                 kang                 4
lee                  lee                  5
-----------------------------------------------------------------

Select user by user number to delete
Input: 2
No permissions, you can only delete only users under your control
```

+ 권한이 없는 user 선택 시