TEAM PROJECT.

MY SQUAD MAKER

MEMBER.

양홍엽, 이철민, 장원진

목차

PART 1 - 프로젝트 개요

GUI / JSP

PART 1 프로젝트 개요

주제 : E-SPORTS 팀 만들기

선정 배경 :

LOL E-SPORTS 10개의 팀에서 나만의 팀을 포지션 별로 선수를 선택해 공유하기 위해서, GUI - DB - JSP 사용하여 커뮤니티를 만들게 되었습니다.

주요 기능 :

- ✓ 선수 관리 기능 : GUI 이용해서 관리자가 추가, 수정, 삭제, 검색 ,목록 기능을 사용하여 JSP에 출력이 되도록 합니다 .
- ✓ 팀 만들기 기능: 2가지 DB를 연동해 JSP사용하여 포지션별로 선수를 선택할 수 있고 게시글을 작성 할 수 있습니다.
- ✓ 정보 관리: 선수 개인의 정보와, 게시판 사용자의 정보를 저장, 관리 합니다.

기획의도

- ✓ 점차 커지는 E스포츠 산업과 팬의 증가로 인하여 주제를 선정하게 되었습니다.
- ✓ 매해 스토브리그가 있고 그에 따라 성공한 팀이 있는 반면 실패한 팀도 있고 컨셉에 따라 자기만의 색갈이 있는 팀도 있습니다.
- ✓ E-스포츠의 팬들이 나만의 팀을 만들어 토론하고 그에 관해 장점과 단점을 쓰고 스토브리그에 적용이 된다면 좋은 데이터로 쓰이고 E-스포츠 스토브리그의 좋은 발전을 위해 기획 하게 되었습니다.

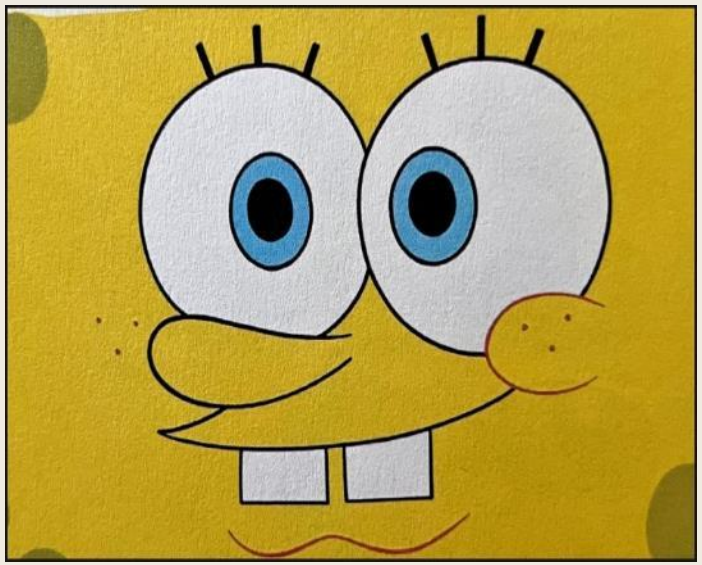
E-스포츠 시장 우 상향 그래프

PART 2
팀 구성 및 역할

PART 2

프로젝트 팀 구성 / 역할

프로젝트 개발 담당자 배치



-담당자 : 양홍엽 ★

-개발 항목 : **JAVA, GUI, DB, HTML, CSS, JSP**

-개발 내용 : **JAVA GUI** 입출력, 검색 **(CONTROLLER VIEW)** **DB**구축 및 연동, **JSP** 게시판 작성, 선수목록, 선수 상세정보, **FOOTER**, 게시판 목록,게시판 상세정보 게시판 수정,삭제, **PDF** 변환

-담당자 : 이철민

-개발 항목 : **DB, HTML, CSS, JSP**

-개발 내용 : **DB**구축 및 연동, **JSP** 게시판 **HEADER**, 선수목록, **FOOTER** 글쓰기 수정 및 삭제, 상세정보 게시판 목록, 게시판 업데이트, **PDF** 변환

-담당자 : 장원진

-개발 항목 : **DB, HTML, CSS, JSP**

-개발 내용 : **DB**구축 및 연동, **JSP** 게시판 **HEADER**, 선수목록, 글쓰기 수정 및 삭제, 상세정보, 게시판 목록, 게시판 업데이트, **PDF** 변환

PART 3
수행 절차 및 방법

PART 3

활용 및 재료

DEVELOPER :

INTELLIJ IDEA
ECLIPSE IDE

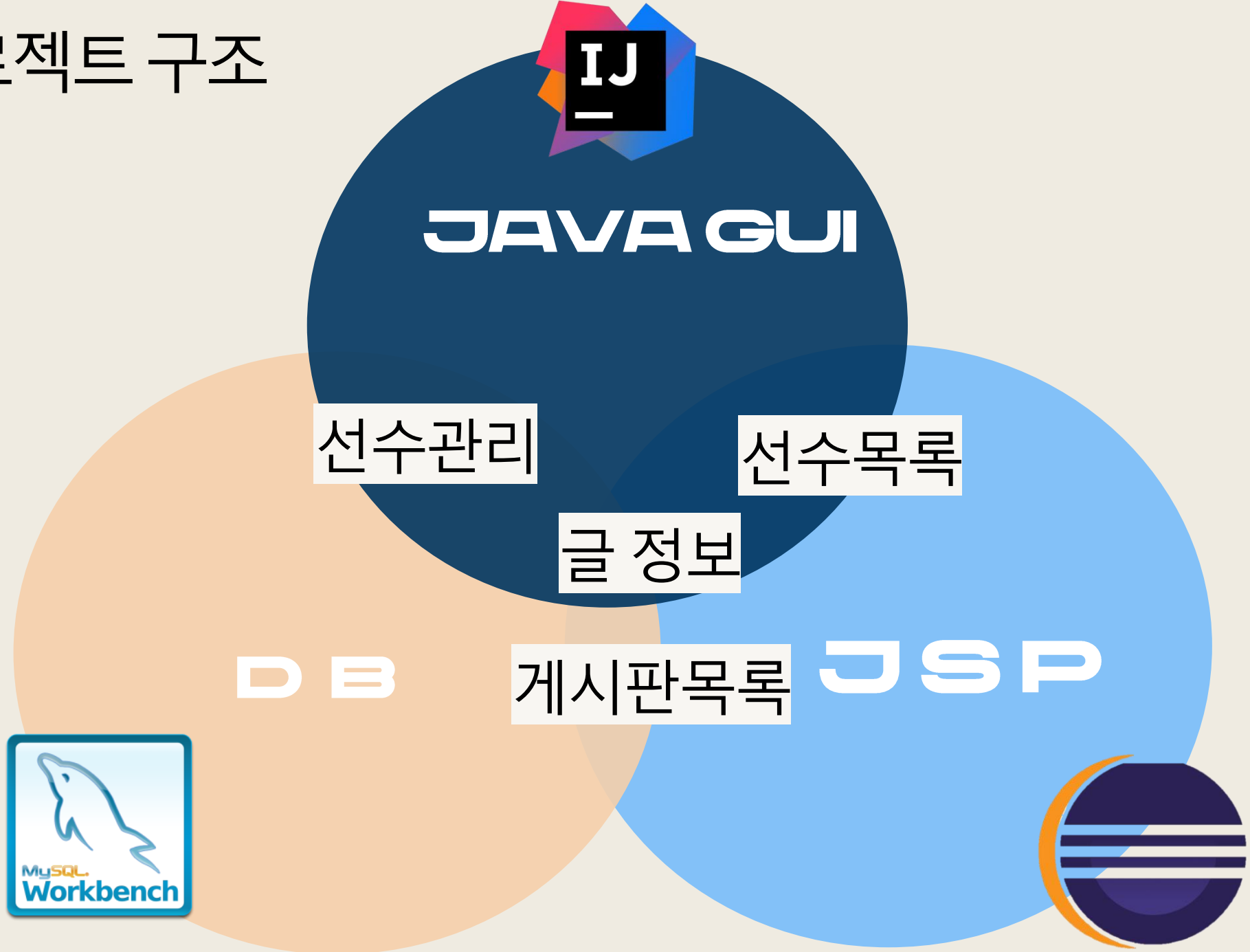
LAGNGUEGE :

JAVA(JDK 22, JDK 18)

DB:

MYSQL_WORKBENCH 8.0
MYSQL_CONNECTOR 8.4.0

PART 3 프로젝트 구조



PART 3

구조 실행 순서

GUI

선수 추가 , 포지션, 팀 변경, 삭제 UI 를 사용하여 데이터베이스 선수관리 DB 로 저장하고 GUI 에 출력합니다.

&&

GUI-DB

선수관리 DB 사진, 데뷔, 커리어 추가 저장합니다.

>>

JSP

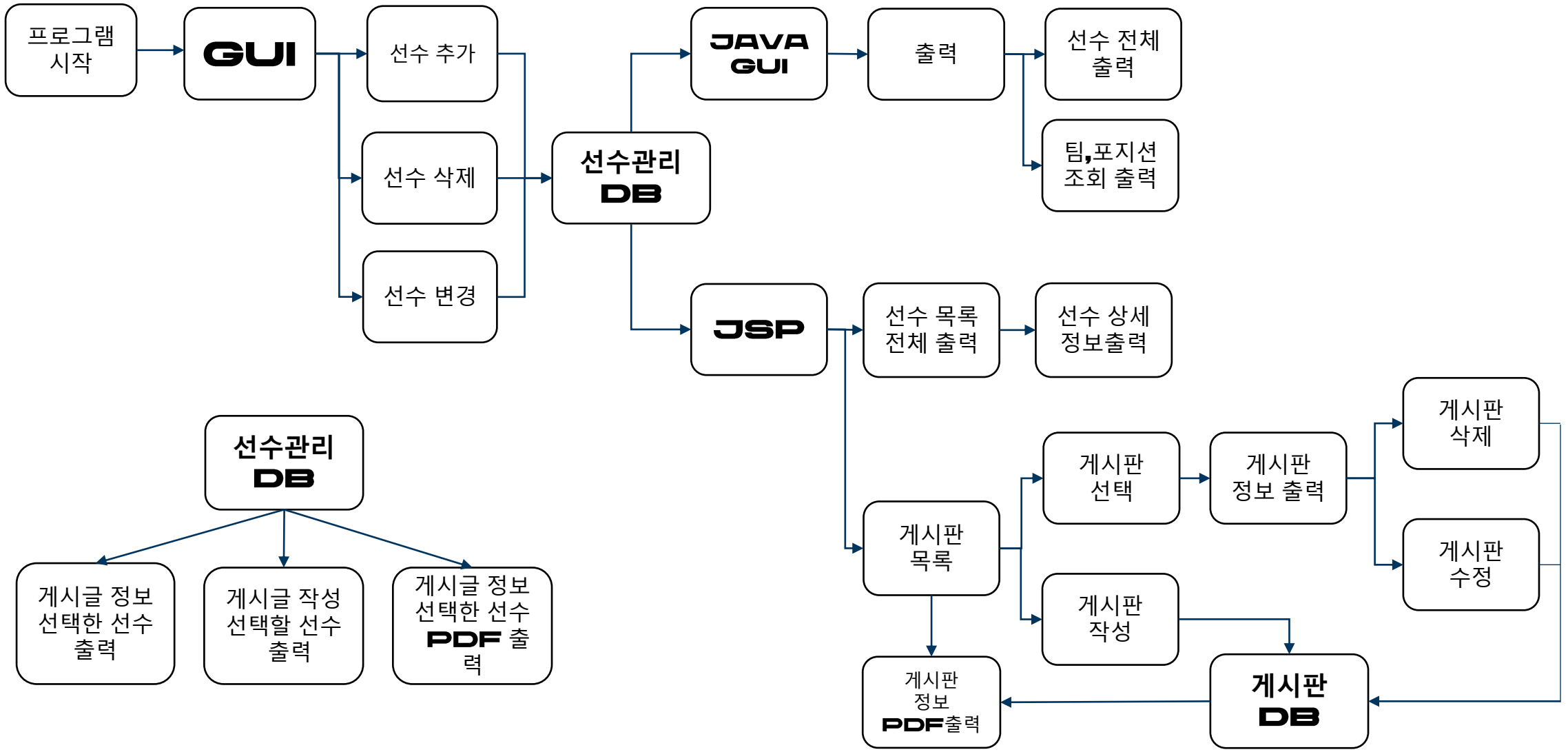
전체 선수 목록, 선수 상세정보 게시판 목록 정보 보기, 게시판 작성, 수정, 삭제 사용하여 게시판 DB 저장합니다.

&&

JSP-DB

게시판 정보 보기, 작성, 수정의 선택한 선수, 선택할 선수의 정보는 선수 관리 DB 에서 불러와 표시합니다.

PART 1 상세 구조



PART 3 수행 절차 및 방법

5월 27일

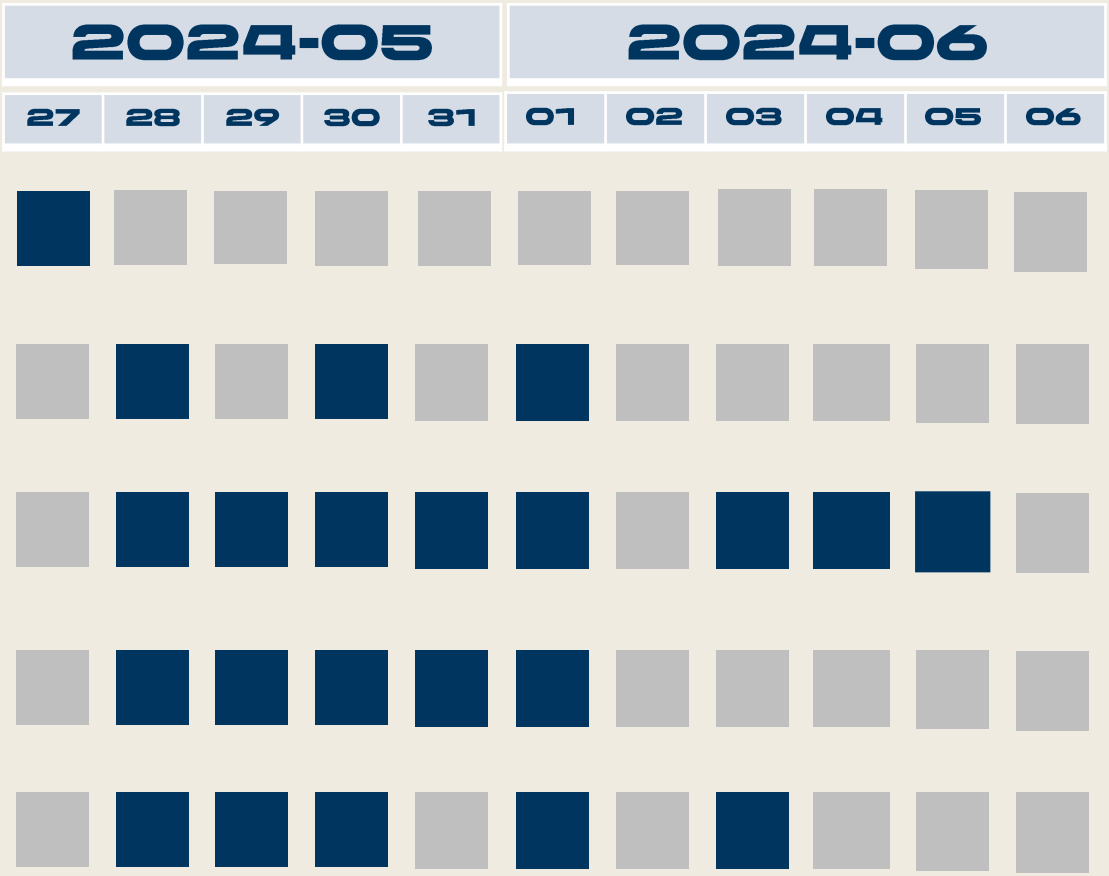
주제 및 세부내용 선정

5월 28일~ 6월 7일

GUI 로직 개발,**UI** 및 수정
선수관리 **DB** 연동 완료

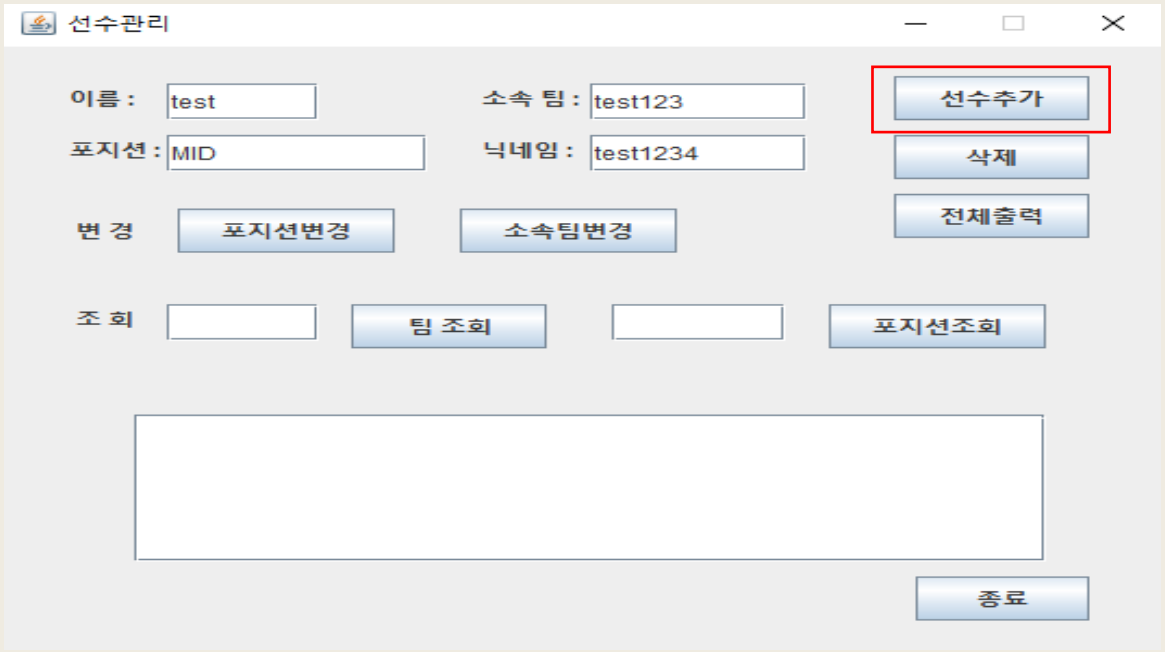
5월 28일~ 6월 4일

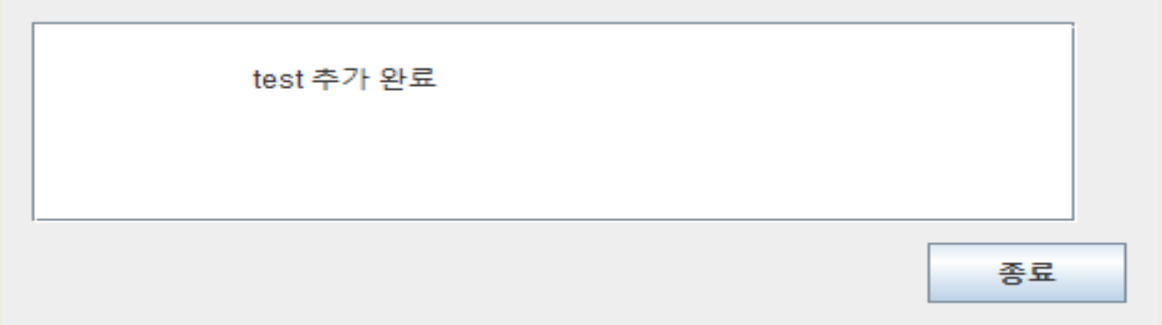
JSP 선수목록, 선수 상세정보, 게시판목록, **PDF** 변환
게시판작성, 게시판정보-수정, 삭제 / 헤더, **FOOTER**, 로직, 디자인 개발 , 선수관리 **DB** , 게시판 **DB** 연동, 완료



PART 4
기능 및 수행 결과

PART 4 프로젝트 GUI 내용

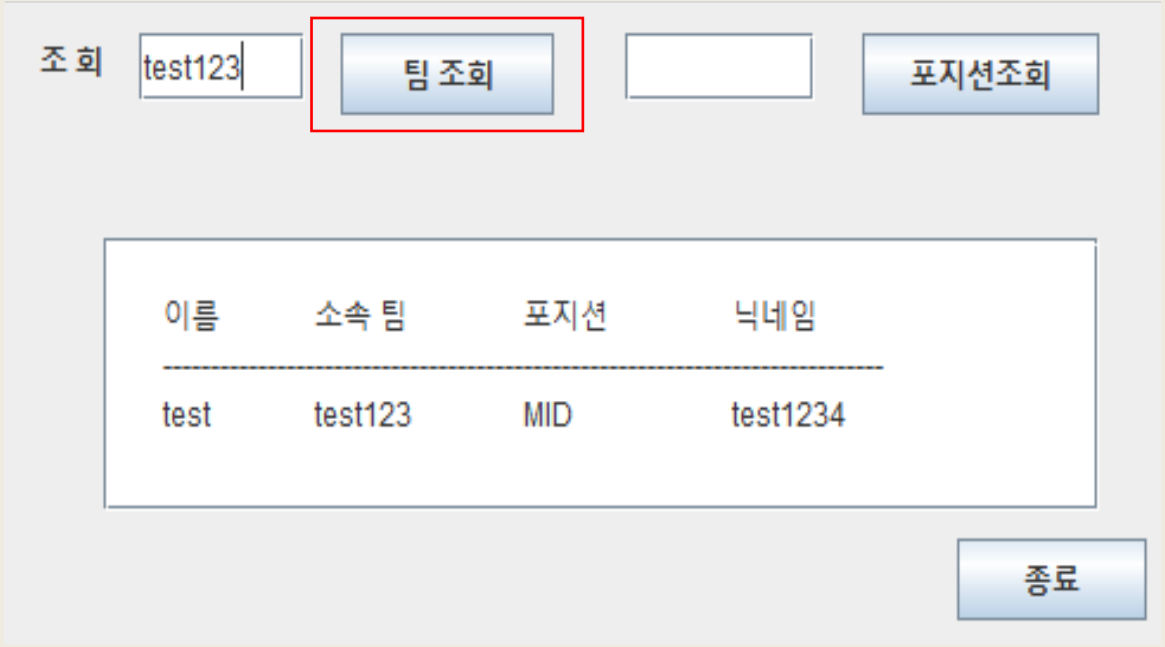




Description

이름, 소속팀 , 포지션 , 닉네임 입력 칸의 값을
DATA 클래스에 저장하고 불러와서 DB와 연결해 선수관리 DB 에 추가 합니다.
//파일,데뷔,커리어,번호 값은 입력 값이 없어서 NULL 로 입력합니다.

PART 4 프로젝트 GUI 내용



Description

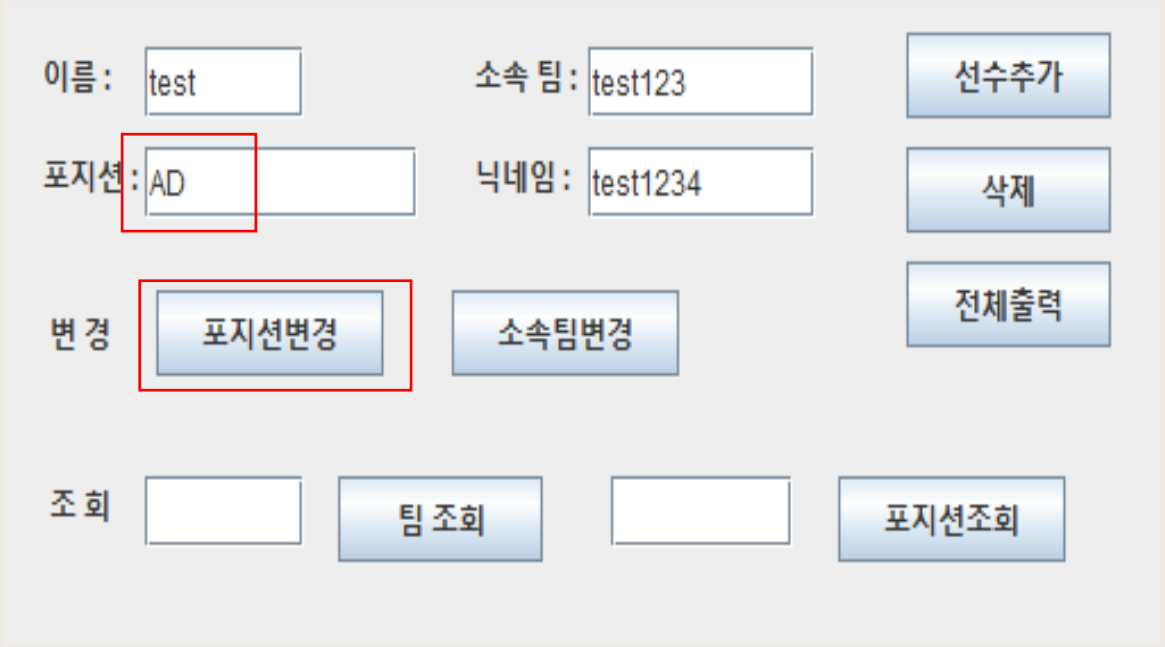
팀 조회 입력 칸의 값을 가져와서 선수관리 DB연결하고 조건문 TEAM 칼럼에 맞는 데이터를 ARRAYLIST<DATA> 에 저장하고 출력합니다. 출력- FOR문으로 APPEND 사용했습니다.

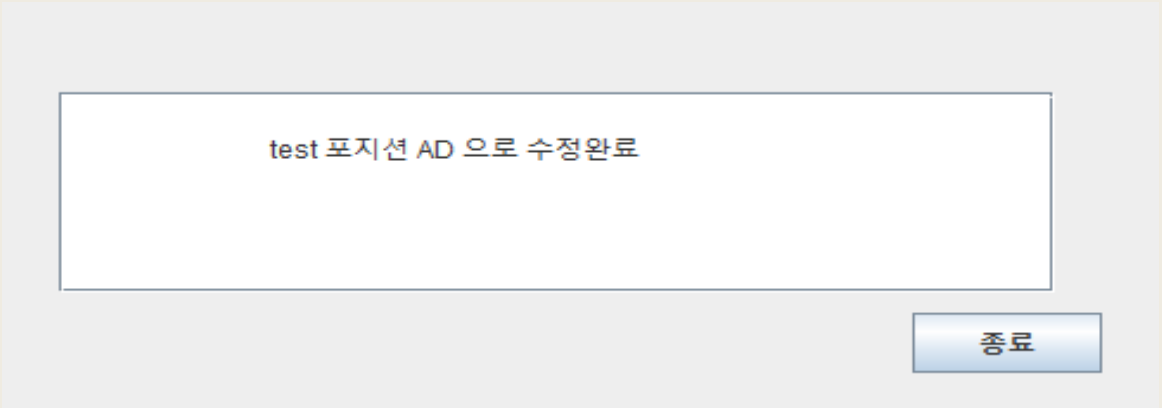
PART 4 프로젝트 GUI 내용

Description

포지션 조회 입력 칸의 값을 가져와서 선수관리 DB연결하고 조건문 POSITION 칼럼에 맞는 데이터를 ARRAYLIST<DATA>에 저장하고 출력합니다. 출력- FOR문으로 APPEND 사용했습니다.

PART 4 프로젝트 GUI 내용

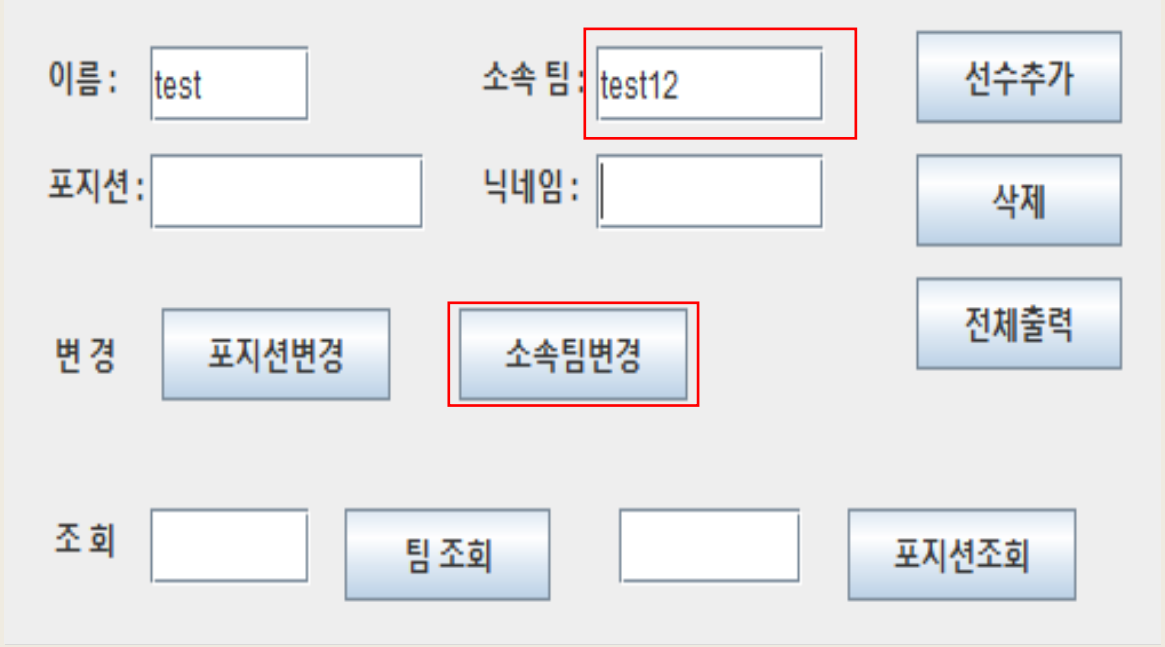


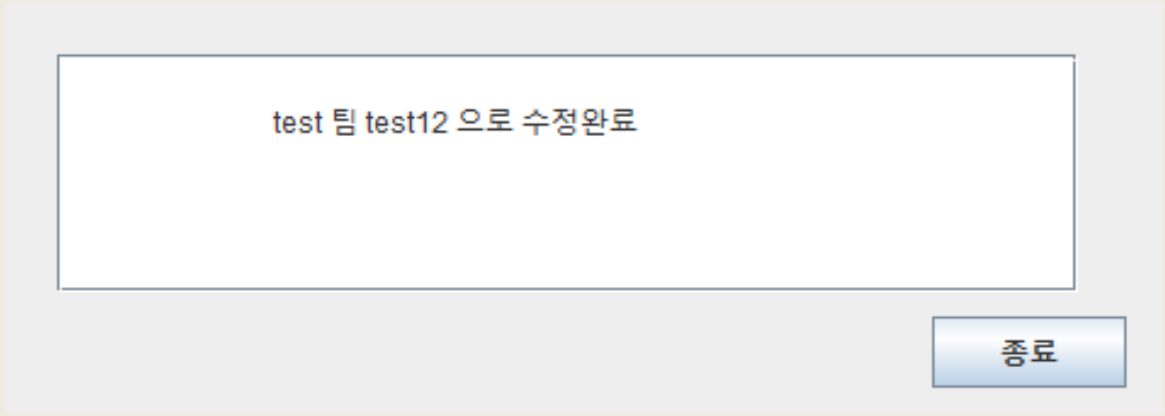


Description

이름, 포지션 입력 값을 가져와서 선수관리 DB에 연결하고 이름 조건의 포지션 값을 변경합니다.

PART 4 프로젝트 GUI 내용

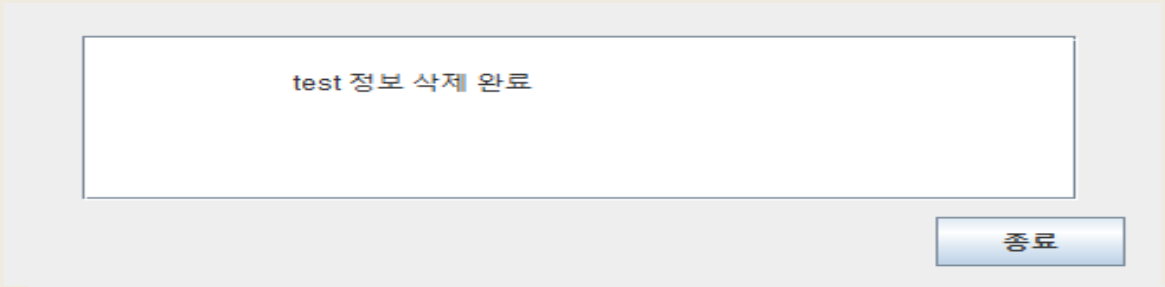


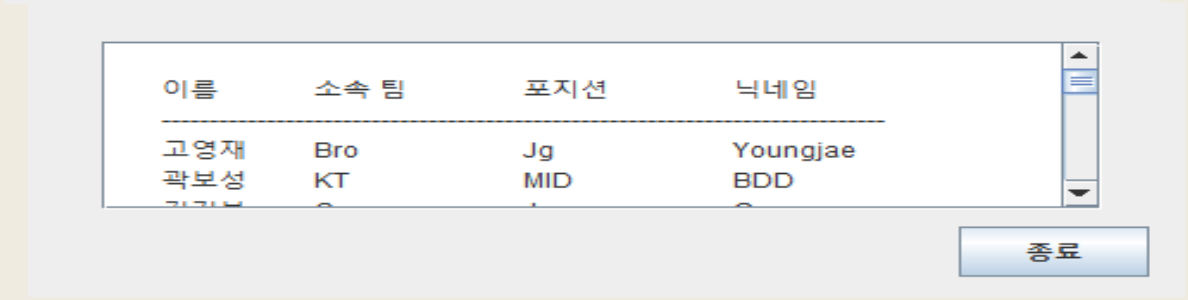


Description

이름, 소속 팀 입력 값을 가져와서 선수관리 DB에 연결하고 이름 조건의 소속 팀 값을 변경합니다.

PART 4 프로젝트 GUI 내용

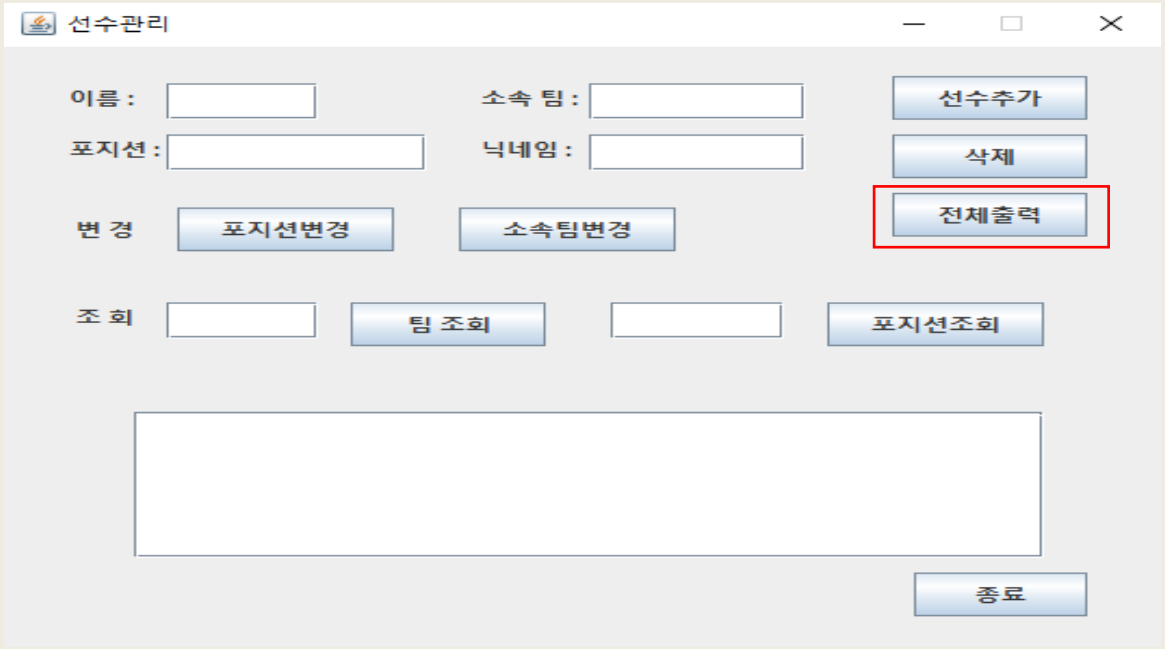


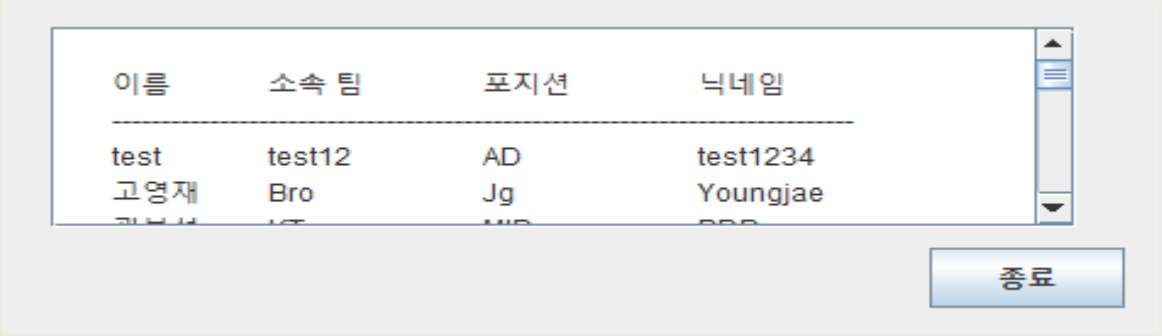


Description

이름 입력 값을 가져와서 선수관리 DB에 연결하고 이름 조건의 데이터를 삭제합니다.

PART 4 프로젝트 GUI 내용



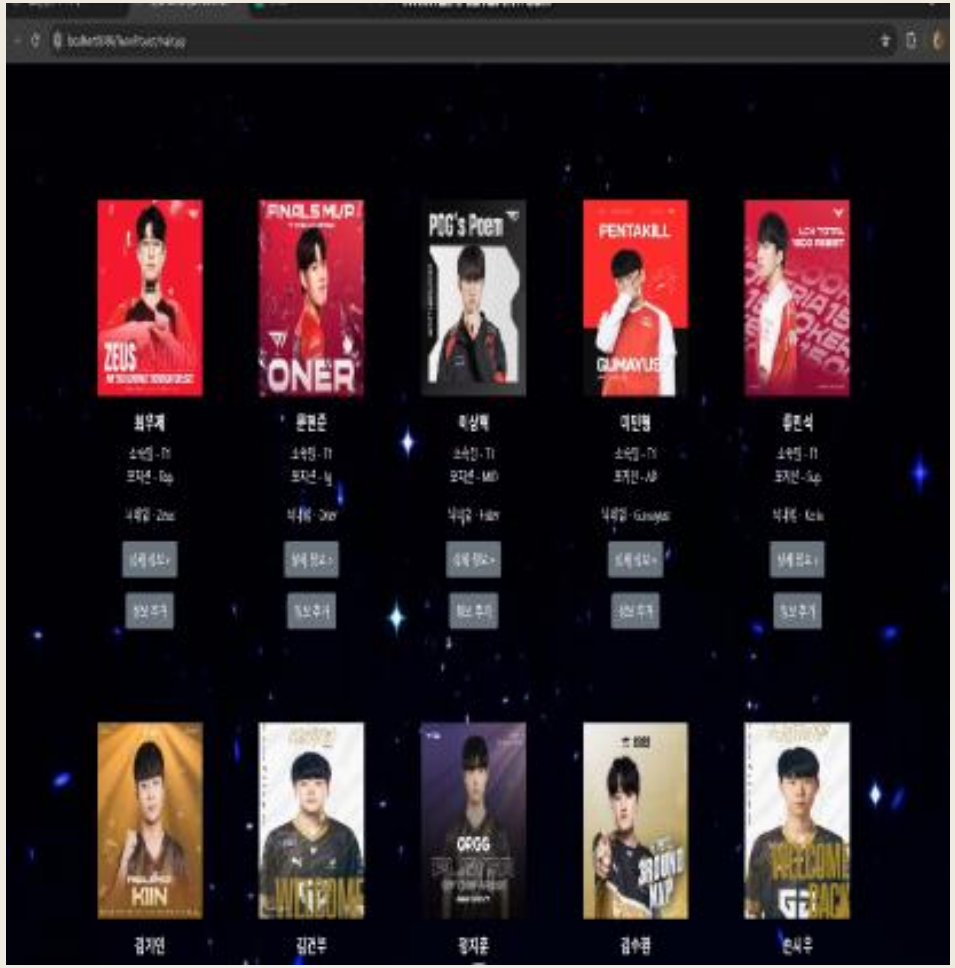


Description

선수관리 **DB**에 연결
ARRAYLIST<DATA>
에 저장하고 출력합니다. 출력-
FOR문으로 **APPEND** 사용
했습니다.

PART 4

MAIN

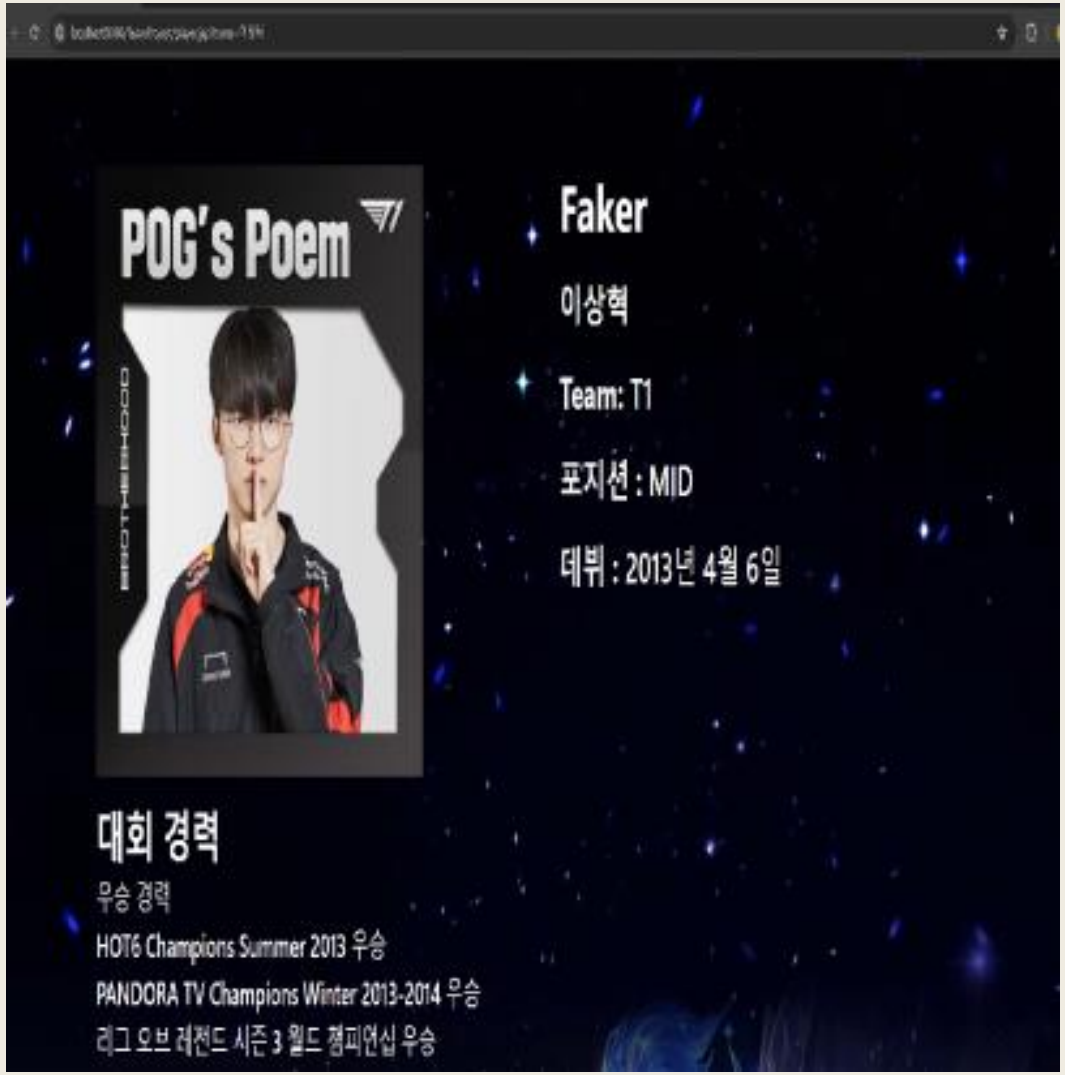


Description

헤더- 로고-> 선수 목록 , 게시판 -> 게시판목록
선수관리 DB를 연동
사진, 이름 ,팀 , 포지션 , 닉네임, 상세 정보(버튼)
출력이 됩니다.

PART 4

PLAYER

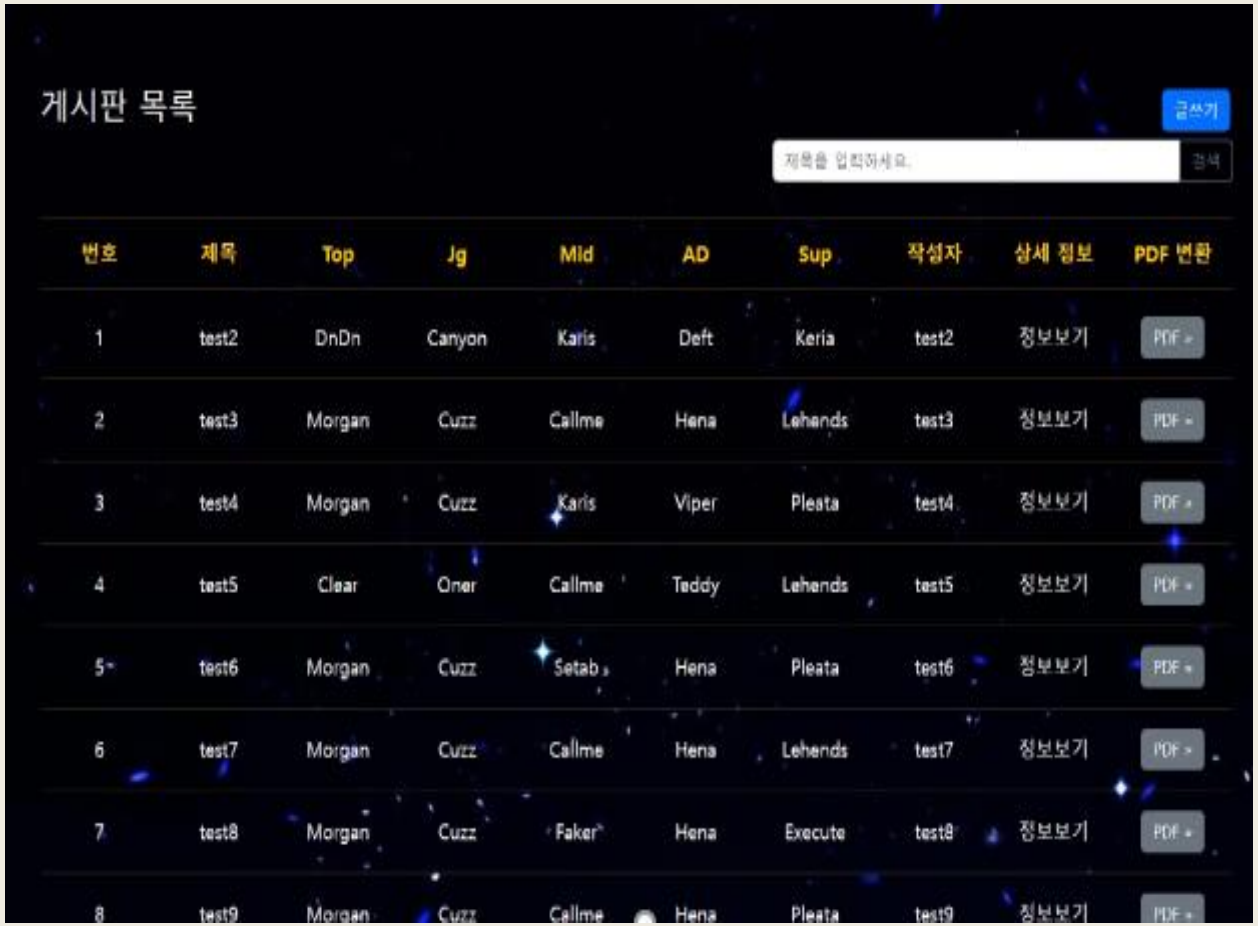


Description

상세정보(버튼) 클릭 선수관리 DB 연결
선수의 사진, 닉네임 , 이름 , 팀 , 포지션 , 데뷔일, 커리어가 출력 됩니다.
커리어는 SPLIT 을 사용하여(,) 기준으로 잘라 배열로 저장 하여 출력합니다.

PART 4

BOARDLIST



Description

게시판 목록 -> 게시판 DB 연동 하여 데이터를 가져옵니다.
게시글의 총 출력 개수를 10개로 지정하고, 자동으로 다음페이지에 출력 하게 했습니다.
번호 , 제목(버튼) , 포지션(TOP, JG, MID, AD, SUP) ,
작성자 , 상세정보(버튼) , PDF 변환(버튼) , 출력 됩니다.

PART 4

BOARDWRITE 소스

Description

제목, 작성자, 비밀번호 , 비밀번호 확인, 포지션 **(TOP,JG, MID, AD, SUP)** 선수 선택 , 내용 입력 , **(**등록,취소**)** 버튼이 출력 됩니다.
선택 할 선수의 정보
포지션 별로 선수는 선수관리**DB** 에서 포지션 조건 으로 가져와서 **5** 개로 나뉘어 지며 선택할 수 있으며 선수관리 **DB** 에 있는 데이터만 표시 됩니다.

PART 4

BOARDWRITE 쿼리/변수

```
<script type="text/javascript">
   function checkForm() {

        var password = document.newWrite.password.value.trim();
        var passwordConfirm = document.newWrite.password1.value.trim();


      if (!document.newWrite.subject.value) {
            alert("제목을 입력하세요.");
            return false;
         }
         if (!document.newWrite.content.value) {
            alert("내용을 입력하세요.");
            return false;
         }
         if (!document.newWrite.prod1.value) {
             alert("Top 포지션을 선택하세요.");
             return false;
          }
         if (!document.newWrite.prod2.value) {
             alert("Jungle 포지션을 선택하세요.");
             return false;
          }
         if (!document.newWrite.prod3.value) {
             alert("Mid 포지션을 선택하세요.");
             return false;
          }
         if (!document.newWrite.prod4.value) {
             alert("ADC 포지션을 선택하세요.");
             return false;
          }
```

```
 if (!document.newWrite.prod5.value) {
     alert("Support 포지션을 선택하세요.");
     return false;
  }name
  if (!document.newWrite.name.value) {
      alert("작성자를 입력하세요..");
      return false;
   }
 if (!password) {
     alert("비밀번호를 입력하세요.");
     return false;
 }
 if (!passwordConfirm) {
     alert("비밀번호 확인을 입력하세요.");
     return false;
 }
 if (password !== passwordConfirm) {
     alert("비밀번호와 비밀번호 확인이 일치하지 않습니다.");
     return false;
 }
 else{
     alert("등록 되었습니다.")
     return true;
```

BOARDWRITE

선수관리 **DB** 에서 포지션 조건으로 데이터를 가져오기 위해 쿼리문을 **5**개를 사용했습니다.
변수-함수
제목,작성자 ,비밀번호, 비밀번호확인 ,포지션 **(TOP, JG , MID , AD , SUP)** 선택 **OR** 입력이 되어야 하고 비밀번호가 서로 같아야 가능합니다.

```
// 각 포지션별 데이터를 가져오는 쿼리
String sqlTop = "SELECT  *FROM 선수관리 WHERE position = 'TOP'";
String sqlJungle = "SELECT * FROM 선수관리 WHERE position = 'jg'";
String sqlMid = "SELECT * FROM 선수관리 WHERE position = 'MID'";
String sqlAdc = "SELECT * FROM 선수관리 WHERE position = 'AD'";
String sqlSup = "SELECT * FROM 선수관리 WHERE position = 'SUP'";

PreparedStatement pstmtTop = conn.prepareStatement(sqlTop);
PreparedStatement pstmtJungle = conn.prepareStatement(sqlJungle);
PreparedStatement pstmtMid = conn.prepareStatement(sqlMid);
PreparedStatement pstmtAdc = conn.prepareStatement(sqlAdc);
PreparedStatement pstmtSup = conn.prepareStatement(sqlSup);

ResultSet rsTop = pstmtTop.executeQuery();
ResultSet rsJungle = pstmtJungle.executeQuery();
ResultSet rsMid = pstmtMid.executeQuery();
ResultSet rsAdc = pstmtAdc.executeQuery();
ResultSet rsSup = pstmtSup.executeQuery();
%>
```

PART 4

PROCESSADDBOARD

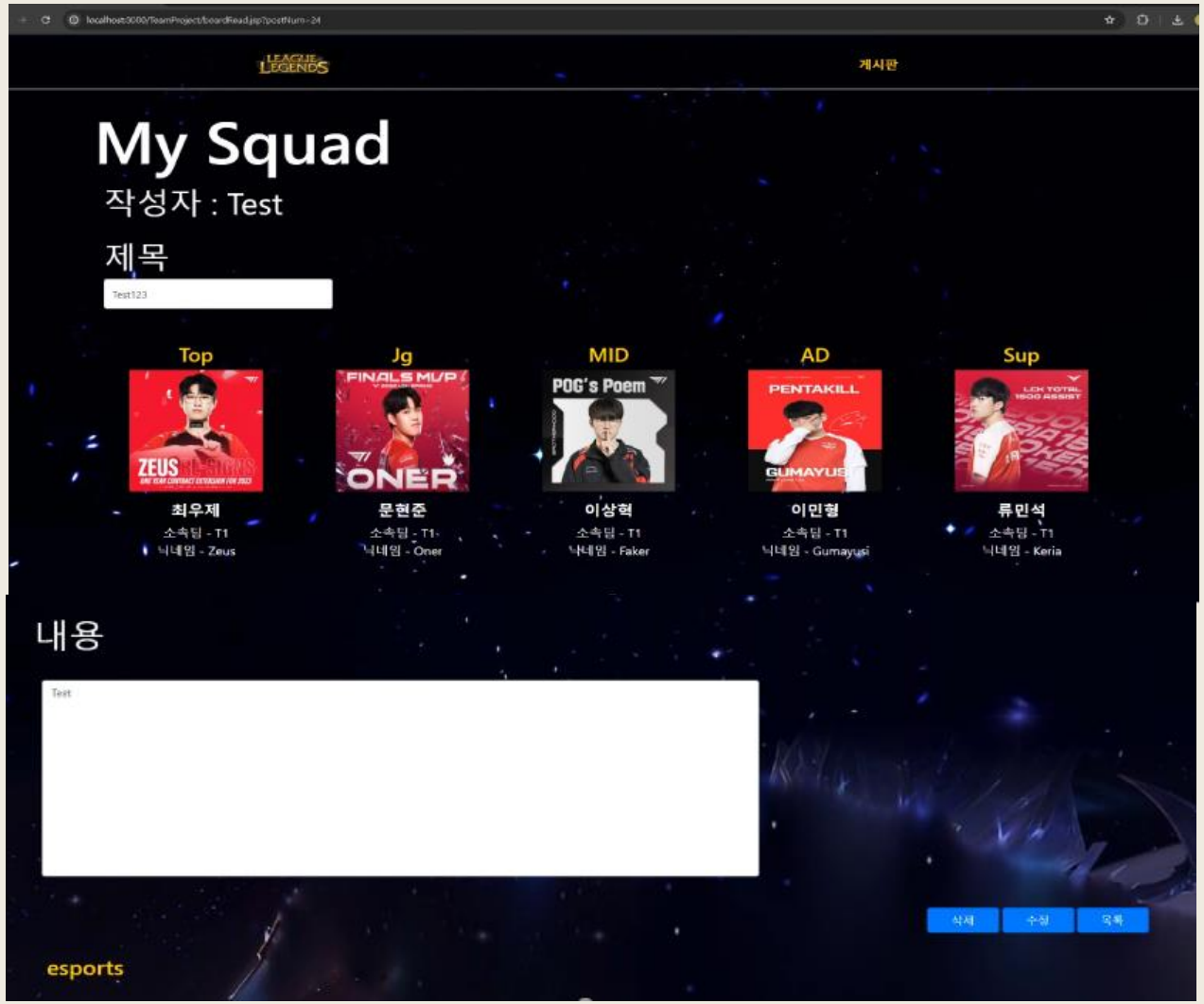
PROCESSADDBOARD

게시글 추가 -> 작성한 데이터 값을 가져와서
번호-자동입력,제목,작성자,
선택한 선수 (TOP, JG, MID , AD, SUP) , 내용 , 비밀번호
게시판 DB를 연동해 저장합니다.

```jsp
<%
    // View에 있는 데이터 추출
    Date date = new Date();
    SimpleDateFormat simpleDate = new SimpleDateFormat("yyyy-MM-dd");
    String strDate = simpleDate.format(date); // 날짜
    request.setCharacterEncoding("UTF-8");

    String subject = request.getParameter("subject"); // 제목
    String prod1 = request.getParameter("prod1"); // 게시글
    String prod2 = request.getParameter("prod2");
    String prod3 = request.getParameter("prod3");
    String prod4 = request.getParameter("prod4");
    String prod5 = request.getParameter("prod5");
    String content = request.getParameter("content");
    String name = request.getParameter("name");
    String password = request.getParameter("password");


    //SQL 데이터 전송
    String sql = "insert into boardlist values(null, ?, ?, ?, ?, ?, ?, ?, ?, ? )";
    PreparedStatement pstmt = conn.prepareStatement(sql);

    pstmt.setString(1, subject);
    pstmt.setString(2, prod1);
    pstmt.setString(3, prod2);
    pstmt.setString(4, prod3);
    pstmt.setString(5, prod4);
    pstmt.setString(6, prod5);
    pstmt.setString(7, content);
    pstmt.setString(8, name);
    pstmt.setString(9, password);

    pstmt.executeUpdate();

    if(pstmt != null){
        pstmt.close();
    }
    if(conn != null){
        conn.close();
    }

    response.sendRedirect("boardList.jsp");
%>
```

PART 4

BOARDREAD

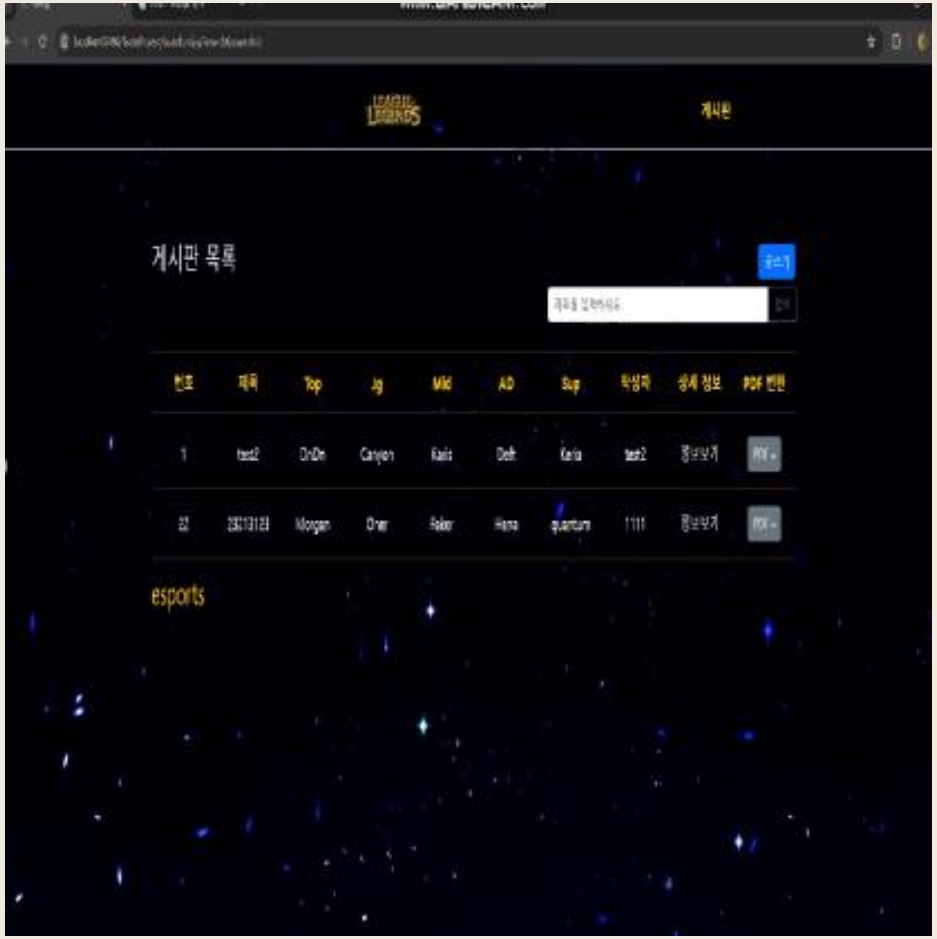


Description

제목 , 작성자 , 포지션 별 선택한 선수 정보-사진,이름,소속팀 , 닉네임 ,수정 할 선수 의 정보(기본값- 선택한선수), 내용,
(수정,삭제, 목록)-버튼 출력 됩니다.
선택한 선수와 수정 할 선수의 정보는 선수관리 DB 연결해 선택한 선수의 데이터를 가져옵니다.

PART 4

게시글 SEARCH & BOARDREAD



Description

게시판 DB 연결
데이터를 번호 조건에 맞게 삭제를 합니다.
삭제와 동시에 게시판 번호를 자동 정렬합니다.

PART 4

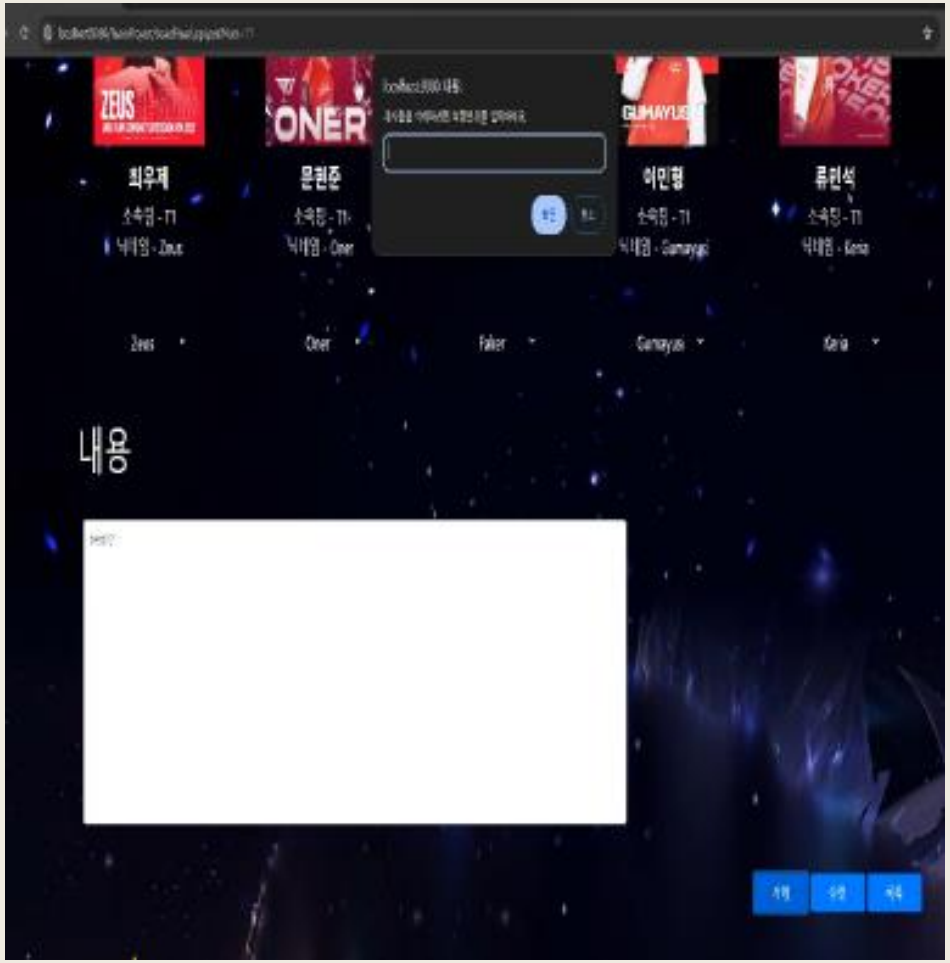
BOARDUPDATE

Description

게시판 **DB**를 연결
변경할 제목, 포지션**(TOP, JG, MID, AD, SUP)**-디폴트 값은 입력했던 선수 입니다.
내용을 글 번호 조건에 맞게 게시판 **DB**에 데이터를 변경 합니다.

PART 4

BOARDDELETE



Description

게시판 **DB** 연결
데이터를 번호 조건에 맞게 삭제를 합니다.
삭제와 동시에 게시판 번호를 자동 정렬 합니다.

PART 4

PDF 소스

```
// PDF 생성을 위한 Document 객체 생성
Document document = new Document();
ByteArrayOutputStream baos = new ByteArrayOutputStream();
PdfWriter writer = PdfWriter.getInstance(document, baos);
document.open();

// 한글 폰트 설정
String fontPath = request.getServletContext().getRealPath("/WEB-INF/fonts/arialuni.ttf");
BaseFont bf = BaseFont.createFont(fontPath, BaseFont.IDENTITY_H, BaseFont.EMBEDDED);
Font font = new Font(bf, 12);

// 테이블 생성
PdfPTable table = new PdfPTable(10);
table.setWidthPercentage(100);
// 테이블 헤더 추가
PdfPCell cell = new PdfPCell(new Paragraph("번호", font));
cell.setHorizontalAlignment(PdfPCell.ALIGN_CENTER);
table.addCell(cell);
cell = new PdfPCell(new Paragraph("작성자", font));
cell.setHorizontalAlignment(PdfPCell.ALIGN_CENTER);
table.addCell(cell);
```

PDF

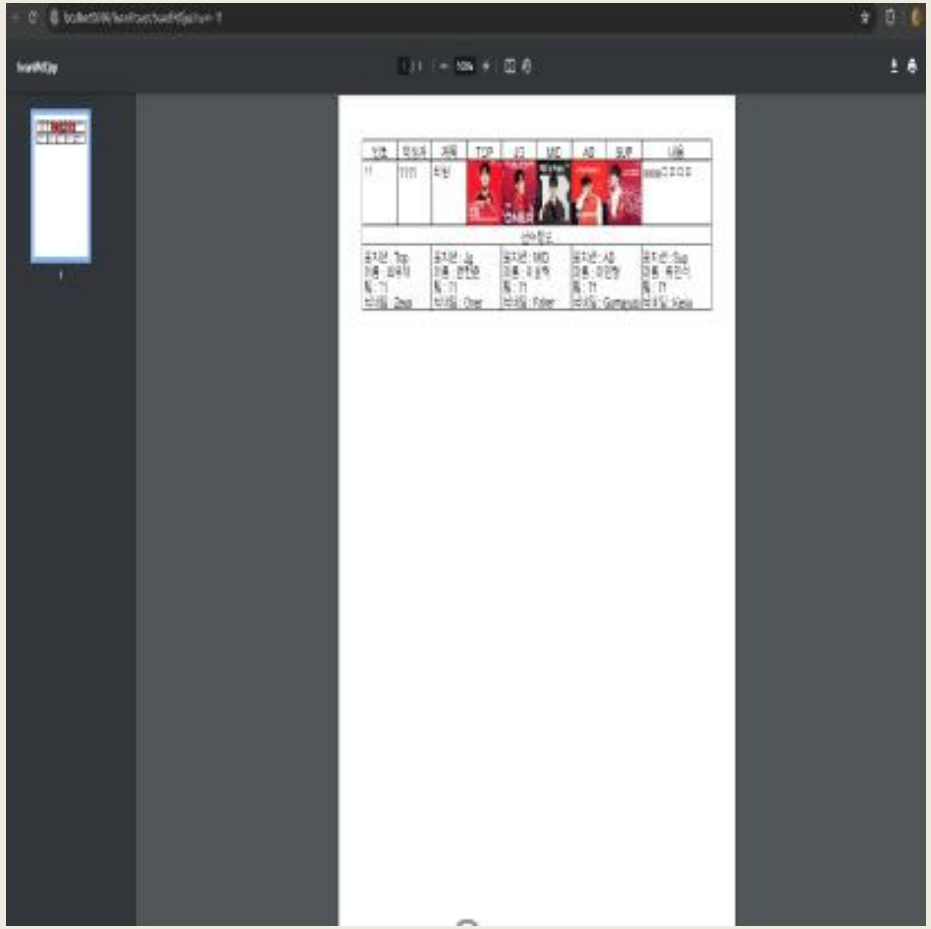
프로젝트에 **FONTS** 폴더를 생성하여 폰트를 담고 적용시킵니다.
테이블을 생성하고 **10**개의 열, 너비는 **100**으로 설정합니다.
여유가 필요한 셀은 **SET.COLSPAN(NUM)**으로 좌우 셀을 병합시킵니다.
헤더가 될 **CELL**에 들어갈 텍스트와 폰트를 적용시켜서 테이블에 추가합니다.
내용이 될 **CELL**에 들어갈 정보를 **DB**에서 가져온 **RESULTSET** 반복문으로 추가합니다. 삼항연산자를 활용하여 **NULL**이면 빈 값, 아니면 해당 값을 **CELL**에 추가합니다.
BOARD.PDF의 이름으로 쓰여진 **BAOS**의 **PDF**데이터를 브라우저로 보냅니다.
버퍼의 데이터도 비운 후 스트림을 닫습니다.

```
// 데이터베이스에서 데이터 가져와 테이블에 추가
while(resultSet.next()) {
    String num = resultSet.getString("num");
    String name = resultSet.getString("name");
    String subject = resultSet.getString("subject");
    String prod1 = resultSet.getString("prod1");
    String prod2 = resultSet.getString("prod2");
    String prod3 = resultSet.getString("prod3");
    String prod4 = resultSet.getString("prod4");
    String prod5 = resultSet.getString("prod5");
    String content = resultSet.getString("content");

    PdfPCell numCell = new PdfPCell(new Paragraph(num != null ? num : "", font));
    table.addCell(numCell);

    PdfPCell nameCell = new PdfPCell(new Paragraph(name != null ? name : "", font));
    table.addCell(nameCell);

    PdfPCell subjectCell = new PdfPCell(new Paragraph(subject != null ? subject : "", font));
    table.addCell(subjectCell);
```

```
    // PDF 파일을 클라이언트에게 전송
    response.setContentType("application/pdf");
    response.setContentLength(baos.size());
    response.setHeader("Content-Disposition", "inline; filename=\"board.pdf\"");

    OutputStream os = response.getOutputStream();
    baos.writeTo(os);
    os.flush();
    os.close();
```

PART 4

PDF 구현영상



Description

게시물 목록 중 **PDF**로 조회를 할 게시글 클릭합니다.
PDF를 누르면 **BOARDPDF**로 링크 됩니다.
클릭한 게시글의 번호를 넘겨받아 게시글 **DB**와 연동합니다.
게시글 **DB**에서 얻은 선수의 닉네임으로 선수관리 **DB**와 연동합니다.
게시글 **DB**의 데이터와 선수관리 **DB**에서 얻은 이미지 및 데이터를 셀에 삽입합니다.
링크된 **BOARDPDF**에서 **PDF**를 출력해 줍니다.

PART 2

DATA BASE (MY SQL)

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G
name	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐
team	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐
position	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐
age	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐
file	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐
career	TEXT	☐	☐	☐	☐	☐	☐	☐	☐
debut	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐
num	INT	☑	☑	☐	☐	☐	☐	☑	☐
		☐	☐	☐	☐	☐	☐	☐	☐

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI
num	INT	☑	☑	☐	☐	☐	☐	☑
subject	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐
prod1	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐
prod2	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐
prod3	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐
prod4	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐
prod5	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐
content	TEXT	☐	☑	☐	☐	☐	☐	☐
name	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐
password	VARCHAR(10)	☐	☑	☐	☐	☐	☐	☐

PART 5

자체 평가 의견

PART 5

자체 평가 의견

Q. 사전 기획의 관점에서 **프로젝트 결과물에 대한 완성도 평가(10점 만점)**

A. **7**점

Q. 개인 또는 우리 팀이 **잘한 부분과 아쉬운 점**

A. 잘한 점 : **JAVA GUI -> DB <- JSP** 연동 **/** 게시판 작성, 수정할 때 해당 요소에 맞는 상세정보 자동적용

아쉬운 점 : **CSS**의 이해도 부족, **JSP**의 완성도 부족 **/ GIT** 활용하지 않은 점.

Q. 프로젝트 결과물의 **추후 개선점이나 보완할 점** 등 내용 정리

A. 이미지 혹은 각 선수 상세정보를 **JSP**에서도 업로드 할 수 있는 방향으로 개선할 필요가 보임.

관리자 로그인 **/** 회원가입 기능 추가 후 **JSP**에서 선수 정보 관리 기능 개선.

JSP 소스의 **CSS**부분을 일관성 있게 수정 후 보완 필요.

GIT 활용하여 **FILE** 저장 및 수정 작업의 편의성, 보완작업 개선 필요.

PART 5

자체 평가 의견

Q. 프로젝트를 수행하면서 **느낀 점이나 경험한 성과**

양홍엽 - 생각보다 시간이 많이 소요되고 에러의 변수가 많아서 배우고 알아가는 계기가 됐습니다. 전체 개발작업을 다 참여 했지만 어려운 부분이 많았고 팀 이기때문에 도움 받아 더 빨리 해결되는 부분도 많아서 좋았고 시너지가 컸습니다. 결과물은 많이 아쉽지만 목표치에 많이 근접했던 것 같아서 다행인 것 같습니다.

이철민 - 협업에 있어 소통이 매우 중요하다는 것을 느꼈으며, 에러의 이유를 찾아가는 과정과 소스의 이해도를 더 높일 수 있는 계기가 됨.

장원진 -팀 프로젝트를 처음 겪으면서 스스로 부족하다는 것을 많이 알았습니다. 실력적인 부분의 결함도 많이 느껴졌고 협업이기에 소통의 중요성도 느껴졌습니다. 팀원의 도움을 받으며 어렵고 힘든 부분을 해결하는 과정에서 실력적인 부분이 많이 향상 된 것 같습니다. 그렇게 만들어진 결과물을 봤을 때 성취감이 좋았습니다. 같이 끌고 가준 팀원분들께 감사합니다.

THANK YOU

MEMBER.

양홍엽★, 이철민, 장원진