프로젝트 수행결과서 (요약본)

MyZeneration

ver1.0

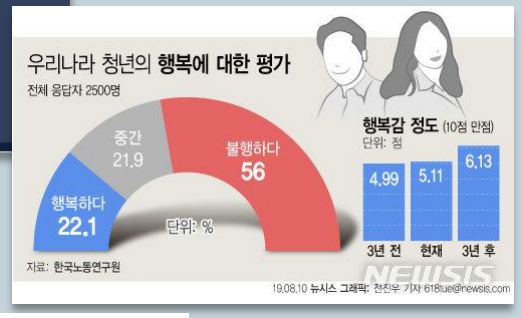
윤 희 선

00 Index 목차

Index	Title	Page #
01	프로젝트 기획서 (기획의도, 개발목표)	p.3~6
02	개발환경 및 사용기술 경험	p.7~9
03	개발 스케줄표	p. 10, 11
04	요구사항정의서	p. 12, 13
05	화면설계서	p.14~16
06	UML (Usecase, sequence, class diagram)	p.17~19
07	주요 서비스 기능 및 소스코드	p.20~26
08	소프트웨어 아키텍쳐	p.27
09	프로그램 서비스 기능 시연	p.28
10	향후 계획 및 프로젝트 수행소감	p.29,30

프로젝트 기획의도 (1)

우울감에 빠진 MZ세대



불행한 대한민국, '카·페·인' 우울증에 빠진 청년들

중앙일보 | 입력 202

20대 사망원인 57%는 극단선택, 고독사는 9년새 3.4배 늘어

지면보기

정신병원 찾는 청년이 늘고 있다 [속앓는 20대①]

"취업 스트레스로 원형탈모까지"…20대 우울증 환자, 5년새 127%↑

기사승인 2022-12-05 06:05:02

SNS에서 행해지는

보여주기식 전시로 인한 타인과의 비교

허탈함, 박탈감

&

한치 앞을 모르는

행복을 장담할 수 없는 미래

불안감, 우울감

다양한 요인에 의해 멀어지는

"MZ세대의 행복"

프로젝트 기획의도 (2)

MZ세대 만족도 향상을 위한 솔루션 제안

나는 어떻게 더 행복할 수 있을까?

연 임금?

가구 원 수?

거주지역?

근로시간?

전공?

결혼여부?

최종학력?

삶의 만족도를 높이고 싶은 MZ세대

#31살
#남자
#대학교졸업
#서울특별시 거주

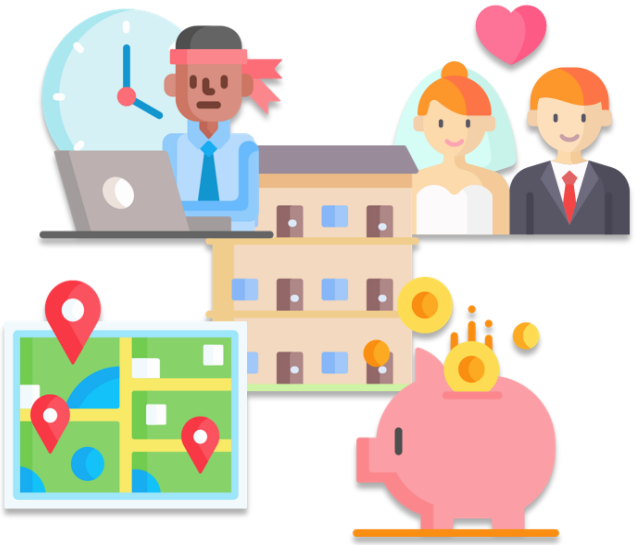
빅데이터 기반

개인의 조건 별 분석결과 제공

삶의 만족도 향상을 위한 솔루션 제안

프로젝트 개발목표 (1)

MZ세대 만족도 분석 서비스



01 나의 만족도 예측 서비스

삶의 만족도 향상을 위한 개인맞춤 솔루션 제안

*회원가입시 설문 받은 사용자의 조건을 분석하고 삶의 만족도를 높이기 위한 개선항목을 추천

02 MZ세대 만족도 비교 서비스

MZ세대의 만족도별 구성인원 비교

* 동일 성별, 나이 일 때 삶의 만족도 점수에 따른 학력, 전공분류, 연봉, 가구인원수, 거주지역 별 구성인원의 정보를 제공

프로젝트 개발목표 (2)

01 나의 만족도 예측 서비스

목표 01 -1

각 항목의 최고 점수를 기준으로 사용자의 항목별 점수를 백분율로 계산 후 데이터 시각화

목표 01 -2

개선이 필요한 항목 중 예상만족도 증가율이 높은 순으로 상위 3개 항목을 추천

목표 01 -3

개선항목 변경 시 예상 만족도를 막대그래프 형태로 나타냄

02 MZ세대 만족도 비교 서비스

목표 02 -1

사용자가 조회하고자 하는 성별, 연령, 삶의 만족도를 각각 선택하면 해당 조건에 따른 학력, 전공 분류, 연봉, 가구 원 수, 거주지역별 구성인원의 비율 정보를 제공

사용기술 목록 (환경/기술/도구)

개발 환경

- Windows 10 [windows10]
- ubuntu Linux [ubuntu 기반 Linux(20.04.3)]
- Java [JDK 1.8.0_361]
- Apache Tomcat [Apache Tomcat9.0]
- ORACLE [Oracle 11.2.0.2.0]

개발 기술

- JSP & Servlet [JSP 2.3 / Servlet 4.0]
- HTML, CSS, JS [HTML5, CSS3, modern js]
- R [R-studio 4.1.1]
- spring [Spring framework 4.3.9]
- spring boot [Spring boot 2.7.14]
- python [python 3.10.11]

개발 도구

- eclipse [eclipse 2019-09]
- Spring Tools 4 [Spring Tool Suits 4.7.0]
- 표준프레임워크 포털 eGovFrame [전자정부 표준 프레임워크 3.9.0]
- Lombok [Lombok 1.18.28]

사용기술 경험 (1)

Java

- ✓ 미니프로젝트 (주소록관리) 경험
  : 컬렉션프레임 (List, Map, Set)을 이용하여 입력, 저장, 삭제, 목록 수행
- ✓ Stream을 이용하여 파일시스템 read, write 경험
- ✓ Multithreading을 위한 클래스 개발 경험
- ✓ 소켓프로그래밍을 이용한 채팅시스템 경험
- ✓ JDBC 사용 경험
- ✓ JSP/Servlet 프로그램 개발 경험

Spring Framework

- ✓ web.xml, servlet-context.xml, application-config.xml 환경설정 및 maven을 이용한 라이브러리 DI 경험
- ✓ xml과 annotation을 이용한 bean 생성 및 DI 경험
- ✓ AOP를 이용하여 각 pointcut에 대한 로깅 및 예외 발생 시 예외 처리 advice 작성 경험
- ✓ JDBC template 사용 경험 (RowMapper 인터페이스를 구현하여 각 행에 대한 정보를 원하는 VO에 매핑)
- ✓ war 파일 배포 경험

MVC Pattern

- ✓ 서비스 별 controller, repository, VO 생성
- ✓ Command객체를 이용한 데이터 수신
- ✓ ModelAndView를 이용하여 데이터 송신 및 view 지정
- ✓ @ResponseBody사용하여 JSON 형태로 데이터 리턴
- ✓ Lombok을 이용하여 VO의 getter/setter 등 메소드 자동 생성
- ✓ session, request 객체를 이용하여 데이터 전달

Spring Boot

- ✓ JSP를 사용하기 위해 Build.gradle에 관련 라이브러리 의존성 추가
- ✓ Web MVC 디자인 패턴을 이용한 프로그래밍
- ✓ war 파일 배포 경험

Database

- ✓ 데이터 모델 및 데이터베이스 구조 학습
- ✓ SQL (DDL, DML,DCL,TCL) 사용 및 테이블 관리 경험
- ✓ sub-query 사용하여 sql 작성
- ✓ 자주 사용하는 query문의 결과 테이블에 대한 view생성 경험
- ✓ ER diagram 작성 경험

Modern JavaScript

- ✓ ajax를 활용한 비동기식 데이터 요청 처리
- ✓ 회원관리 시 유효성 검사
- ✓ chart js를 활용한 그래프 처리
- ✓ 새 탭으로 열기 및 팝업차단 해제 요청 알림
- ✓ createDOM을 이용하여 테이블 생성

사용기술 경험 (2)

JSP

- ✓ 공통 화면을 모듈화하여 include지시어로 각 화면에 삽입
- ✓ EL을 이용하여 해당 세션의 id값의 null 여부 판단
- ✓ JSTL의 반복문을 이용하여 JSP의 option 요소를 동적으로 생성
- ✓ form 태그를 이용하여 get/post 방식으로 데이터 전달

HTML5/CSS3/Bootstrap

- ✓ 그리드 시스템 (row)을 사용하여 페이지 내 행을 단위로 요소를 그룹화
- ✓ CSS를 활용하여 메인 화면 애니메이션 적용 경험
- ✓ 미디어쿼리, 부트스트랩으로 반응형 페이지 제작경험

python

- ✓ 기본 문법 (조건문, 반복문) 학습
- ✓ 파일시스템 데이터 read , write
- ✓ numPy 라이브러리를 이용한 데이터분석 실습 및 sklearn 라이브러리를 이용한 머신러닝 분석 실습
- ✓ 공공 API와 연동 실습

React

- ✓ class-based component 및 function-based component 학습
- ✓ props를 이용하여 element간 데이터 전달 실습
- ✓ component function
- ✓ class-based component의 state변경 시 re-rendering 실행 및 생명주기 관리 실습
- ✓ function-based component에 hook을 이용하여 변수 값 변경 시 re-rendering 실행 및 생명주기 관리 실습

R

- ✓ 공공 API와 R 연동 실습
- ✓ 단순/다중선형회귀 및 로지스틱 회귀분석 실습
- ✓ 산점도 등 데이터 시각화 실습

Git / GitHub

- ✓ Git bash를 이용하여 기본적인 git 명령어 사용 경험
- ✓ GitHub를 이용하여 팀원과 협업경험

Linux

- ✓ Ubuntu 기반 Linux 개발환경 구축 경험

Big data & UI

업무 카테고리	이름	번호	세부항목	시작일	종료일	작업기간
기획	윤희선, 조민정, 이준형	1	아이디어 구상	2023-05-23	2023-05-25	3
		2	기획서 초안작성	2023-05-25	2023-05-28	4
		3	기획서 중간발표	2023-05-30	2023-05-30	1
	윤희선	4	기획서/요구사항 정의서 작성	2023-05-31	2023-06-03	4
	윤희선	5	와이어프레임, 화면설계, 명세서 작성	2023-06-05	2023-06-09	5
	공통	6	1차 프로젝트 결과 발표	2023-06-23	2023-06-23	1
빅데이터분석	공통	7	데이터수집	2023-05-29	2023-06-03	6
	윤희선	8	데이터분석 및 전처리	2023-06-07	2023-06-15	9
	윤희선	9	데이터 시각화	2023-06-15	2023-06-17	3
UI	윤희선	10	메인화면 개발	2023-06-18	2023-06-20	3
		11	메인서비스(나의 만족도 예측서비스) 화면 개발	2023-06-21	2023-06-24	4
		12	메인서비스(MZ세대 만족도 비교 서비스) 화면 개발	2023-06-25	2023-06-27	3
		13	문의하기 게시판 화면 (목록/게시글 작성/열람) 개발	2023-06-28	2023-06-29	2
		14	로그인, 회원가입(회원가입/ 회원가입완료) 화면 개발	2023-06-30	2023-07-01	2
		15	회원관리(아이디찾기/결과 및 비밀번호 찾기/변경/완료) 화면 개발	2023-07-02	2023-07-02	1
		16	설문지(설문지/설문제출) 화면 개발	2023-07-03	2023-07-03	1

Servlet & JSP / Spring framework

업무 카테고리	이름	번호	세부항목	시작일	종료일	작업기간
백엔드 (Servlet)	윤희선	17	UML제작(class diagram, Usecase diagram, sequence diagram)	2023-07-17	2023-07-18	2
		18	HTML파일을 JSP로 변환	2023-07-19	2023-07-19	1
		19	나의 만족도 예측 서비스- 입력한 개인정보 테이블 및 각 항목에 대한 백분율 데이터 시각화(레이더차트) 개발	2023-07-19	2023-07-20	2
		20	나의 만족도 예측 서비스- 개선항목 추천 및 항목 변경 시 만족도 증가율 보여주기 기능 개발	2023-07-20	2023-07-20	1
		21	나의 만족도 예측서비스 - 만족도 증가율 높은 순으로 데이터 시각화 (막대그래프) 기능 개발	2023-07-21	2023-07-22	2
		22	나의 만족도 예측서비스 - 항목 변경시 증감율, 레이더차트, 막대그래프를 ajax로 비동기 요청 처리	2023-07-23	2023-07-25	3
		23	MZ세대 만족도 비교 서비스 - 카테고리 조건 중복 선택기능 개발	2023-07-26	2023-07-26	1
		24	MZ세대 만족도 비교 서비스 - 각 항목 별 비율 데이터시각화 (도넛그래프)	2023-07-27	2023-07-27	1
		25	MZ세대 만족도 비교 서비스 - 카테고리 조건 변경 시 막대그래프를 ajax로 비동기 요청 처리	2023-07-28	2023-07-30	3
		26	로그인, 회원가입기능 개발	2023-07-30	2023-07-31	2
		27	설문지 입력 기능 개발	2023-07-31	2023-07-31	1
		28	회원관리(아이디찾기, 비밀번호 찾기, 비밀번호 변경) 기능 개발	2023-08-01	2023-08-01	1
		29	게시글 목록, 게시글 작성, 게시글 삭제기능 개발	2023-08-02	2023-08-03	2
		30	예외처리	2023-08-04	2023-08-04	1
백엔드 (Spring Framework)	윤희선	31	xml 파일 환경설정	2023-08-27	2023-08-27	1
		32	빈생성 및 의존성 주입	2023-08-28	2023-08-28	1
		33	나의 만족도 예측 서비스 개발	2023-08-29	2023-08-30	2
		34	MZ세대 만족도 비교 서비스 개발	2023-08-30	2023-08-31	2
		35	로그인, 회원가입 기능 개발	2023-09-01	2023-09-01	1
		36	게시판 기능개발	2023-09-02	2023-09-03	2

요구사항정의서 (1)

▼ 요구사항 정의서 중 [나의 만족도 예측 서비스] 부분 발췌

요구사항 정의서

프로젝트명: MyZeneration

(MZ 세대 만족도 분석서비스)

2023.05.23 ~ 2023.09.05

성명: 윤희선

문서번호: ver1.0

업데이트일자	version	업데이트 내용
23.06.03	1.0	요구사항 정의서 업로드

- 메인 화면의 배너는 #ee7752, #863ce7, #2361d5, #23d5b 색상의 그라데이션으로 구성되어 있어야 하고 백그라운드 애니메이션은 #ee7752→ #863ce7→ #2361d5→ #23d5b→#2361d5→ #863ce7→ #ee7752 을 한 사이클로 색상전환이 이뤄져야 한다.
- 15 초당 한 사이클이 실행되어야 한다.

2.3 외/내부 링크

- '한국노동패널조사 사이트로 이동' 버튼에 https://www.kli.re.kr/klips#secondPage 링크를 걸어주며, 클릭 시 해당 웹 사이트는 새 탭으로 열려야 한다.
- 서비스 이용하기 칸 안에 '로그인'과 '회원가입' 버튼이 있어야 하며 각각 클릭 시 '로그인 화면'과 '회원가입화면'으로 이동할 수 있어야 한다.

2.4 요소 애니메이션 효과

- 화면에 나타나는 요소들은 아래에서 위로 나타나는 애니메이션이 적용되며 나타나야 한다.

3 나의 만족도 예측 서비스 기능

3.1 로그인 여부 검증

- 나의 만족도 예측 서비스 버튼 클릭 시 사용자의 로그인 여부를 확인해야 한다.

3.1.1로그인 사용자 (회원)의 경우

- 나의 만족도 예측 서비스 화면으로 이동한다.

3.1.2로그인 하지 않은 사용자 (비회원)의 경우

7

- alert 창으로 '로그인 후 이용이 가능합니다.'를 화면에 표시하고 사용자가 확인 버튼을 누르면 로그인 화면으로 이동한다.

3.2 '내가 입력한 정보' 테이블

- 회원가입 시 설문을 바탕으로 수집했던 사용자의 정보가 테이블 형태로 표기되어야 한다.
- 성별과 나이는 사용자가 입력한 값을 가져야 하고 변경할 수 없어야 한다.
- 최종학력, 전공분류, 거주지역, 연 임금, 가구원 수, 거주형태, 근로시간, 결혼 여부 항목에는 옵션이 있으며 각 옵션은 하기 3.2.1 항과 같다.
- '나의 만족도 예측 서비스'에 맨 처음 접속했을 때 회원가입 시 사용자가 입력한 데이터와 동일한 값이 테이블에 표기되어야 한다.

3.2.1테이블 항목의 옵션

- 최종학력 - 대학원졸업, 대학교졸업, 고등학교졸업
- 전공분류 - 이학/공학, 인문/사회, 예체능
- 거주지역 - 서울특별시, 부산광역시, 광주광역시
- 연 임금 - 6000 만원, 5000 만원, 4000 만원, 3000 만원, 2000 만원, 1000 만원
- 가구원 수 - 5 인, 4 인, 3 인, 2 인, 1 인
- 거주형태 - 자가, 부모님과동거, 전세, 월세
- 근로시간 - 40 시간미만, 40~49 시간, 50~59 시간, 60 시간 이상
- 결혼여부 - 기혼, 미혼

3.3 만족도 분석 및 예측

8

요구사항정의서 (2)

▼ 요구사항 정의서 중 [MZ세대 만족도 비교 서비스] 부분 발췌

- 사용자가 입력한 만족도와 동일 조건의 사람들이 갖는 평균 만족도는 RGB(13,102,221) 색상으로 강조되어야 한다.
- 각 항목에 대한 백분율 정보는 레이더 차트 형태로 표기되어야 한다.
- 테이블에서 선택한 옵션에 따라 레이더 차트가 변해야 한다.
- 백분율이 100% 미만인 항목은 그래프 아래 텍스트로 표기되어야 한다.

3.4 각 항목 별 개선점에 따른 예상 만족도

- 추천 개선항목을 적용했을 때 만족도 증가율이 높은 순으로 3 개의 항목을 테이블 형태로 정보제공 해야 하며 각 컬럼에 순위, 추천하는 변경사항, 예상 만족도 증가율이 표기되어야 한다.
- 추천 개선 항목을 적용했을 때 만족도 증가율이 높은 순으로 모든 항목에 대한 예상 만족도 점수를 막대그래프 형태로 표기해야 한다.

4 MZ세대 만족도 비교 화면 기능

4.1 로그인 여부 검증

- MZ 세대 만족도 비교분석 버튼 클릭 시 사용자의 로그인 여부를 확인해야 한다.

4.1.1로그인 사용자 (회원)의 경우

- MZ세대 만족도 비교분석 화면으로 이동한다.

4.1.2로그인 하지 않은 사용자 (비회원)의 경우

- alert 창으로 '로그인 후 이용이 가능합니다.'를 화면에 표시하고 사용자가 확인 버튼을 누르면 로그인 화면으로 이동한다.

4.2 카테고리 선택기능

9

- 화면 상단에 카테고리 선택 메뉴가 있어야 한다
- 사이드 바 내 총 3 개의 카테고리 (성별, 연령, 삶의 만족도 점수)로 나뉘어 표기되어야 한다.
- 선택 시 버튼 색상은 #863ce7, 글자는 #fff 으로 변환되어야 한다.

4.2.1성별 카테고리의 경우

- 항목은 2 개로 '남성'과 '여성'이 있으며 중복 선택이 가능해야 한다.

4.2.2연령 카테고리의 경우

- 항목은 6 개로 '20 대 초 (20~22)', '20 대 중 (23~26)', '20 대 후 (27~29)', '30 대 초(30~32)', '30 대 중 (33~36)', '30 대 후 (37~39)'가 있으며 중복선택이 가능해야 한다.

4.2.3삶의 만족도 점수 카테고리의 경우

- 항목은 총 9개로 '10점대', '20점대', '30점대', '40점대', '50점대', '60 점대', '70 점대', '80 점대', '90 점대'가 있으며 중복 선택이 가능해야 한다.

4.3 카테고리 선택 메뉴바에서 선택한 조건에 따른 정보제공 기능

- 맨 처음 MZ 세대 만족도 비교화면에 접근 시 default 값으로 성별엔 남성, 여성이, 연령엔 30 대 초, 삶의 만족도 점수는 80 점대가 선택되어있어야 한다.
- 카테고리 선택 메뉴에서 바로 항목을 선택하면 비동기적으로 그래프의 형태가 바뀌어야 한다.

10

- 그래프는 총 5 개이며, 학력별, 전공분류별, 연봉별, 가구원 수별, 거주지역별 순서로 표시되어야 한다.
- 그래프는 도넛형태의 그래프이며, 그래프 아래 각 조건별 구성인원의 비율이 표기되어야 한다.
- 그래프의 색상은 비율이 높은 순으로 'RGB(7,77,129)', 'RGB(32,203,194)', 'RGB(64,178,230)','RGB(204,153,255)','RGB(134,60,231)' 색상이어야 한다.

5 문의하기 기능

5.1 게시글 목록 화면 기능

5.1.1로그인 여부 검증

- 문의하기 버튼 클릭 시 사용자의 로그인 여부를 확인해야 한다.

5.1.1.1 로그인 사용자 (회원)의 경우

- 게시글 목록 화면으로 이동한다.

5.1.1.2 로그인 하지 않은 사용자 (비회원)의 경우

- alert 창으로 '로그인 후 이용이 가능합니다.'를 화면에 표시하고 사용자가 확인 버튼을 누르면 로그인 화면으로 이동한다.

5.1.2게시글 목록 기능

- 각 게시글의 번호, 제목, 작성자의 ID, 작성 일자, 조회 수가 표시되어야 한다.
- 게시글의 제목을 클릭하면 해당 게시글로 이동할 수 있어야 한다.

11

화면설계서 주요서비스 (1)

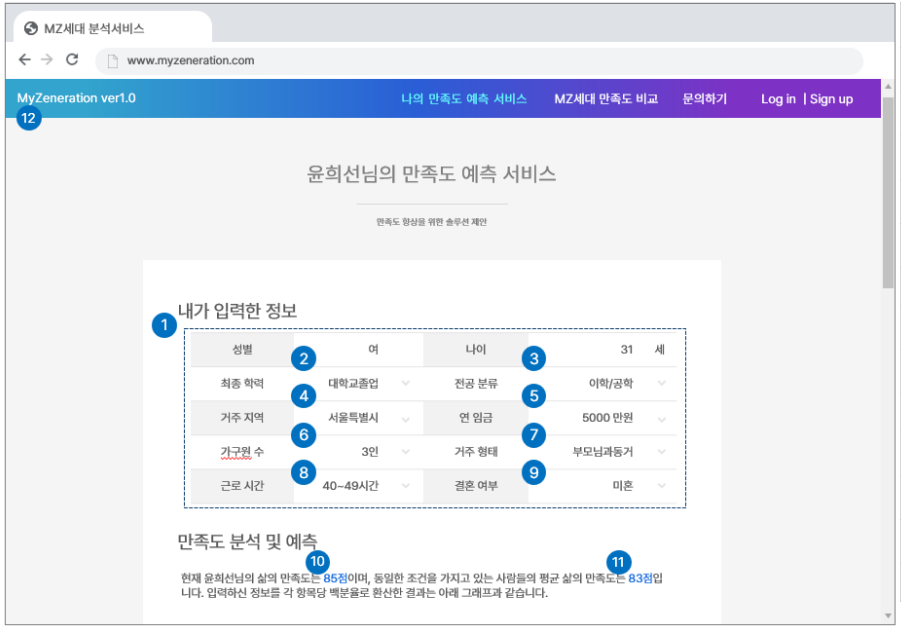
▼ 화면 목록

페이지#	카테고리	설명	화면 명
1	메인	메인 화면	index
2	서비스1	나의 만족도 예측서비스 화면	analysisMySat
3	서비스2	MZ세대 만족도 비교 화면	analysisMz
4	문의하기 게시판	게시글 목록 화면	bList
5		게시글 작성 화면	bWriteView
6		게시글 열람 화면	bContentView
7	회원관리	회원가입 화면	joinView
8		회원가입완료 화면	joinOk
9		설문지 화면	survey
10		설문완료 화면	surveyComplete
11		로그인 화면	loginView
12		아이디 찾기 화면	idFindView
13		아이디 찾기 결과 화면	idFindResult
14		비밀번호 찾기 화면	pwFindView
15		비밀번호 변경 화면	pwChangeView
16		비밀번호 변경완료 화면	pwChangeOk

주요서비스

▼ 주요서비스: 나의 만족도 예측서비스 화면

프로젝트명	MZ세대 분석서비스 "나는 어떤 MZ일까?"	설명	나의 만족도 예측 서비스	페이지 #	2	페이지명	analysisMySat
화면경로	메인 화면 → 나의 만족도 예측 서비스						

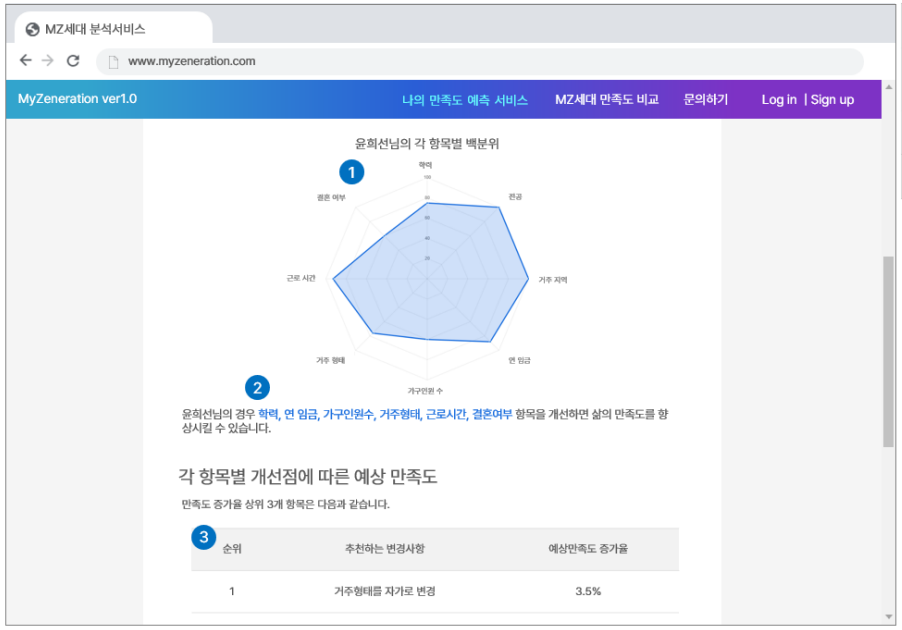


No.	Description
1	'나의 만족도 예측 서비스' 클릭 시, 회원가입 시 입력했던 개인정보가 테이블 형식으로 나타남
2	최종학력란의 option항목은 대학원졸업, 대학교졸업, 고등학교 졸업
3	전공분류란의 option항목은 이학/공학, 인문/사회, 예체능
4	거주지역란의 option항목은 서울특별시, 부산광역시, 광주광역시
5	연 임금란의 option항목은 6000만원, 5000만원, 4000만원/3000만원, 2000만원, 1000만원
6	가구원수란의 option항목은 1인, 2인, 3인, 4인, 5인
7	거주형태란의 option항목은 자가, 부모님과동거, 전세, 월세
8	근로시간란의 option항목은 40시간미만, 40~49시간, 50~59시간, 60시간 이상
9	결혼여부란의 option항목은 미혼, 기혼
10	회원가입 시 사용자가 입력한 삶의 만족도가 나타남 (글자색상: rgb(13, 102, 221))
11	회원가입 시 사용자가 입력한 조건을 가진 사람들의 평균 만족도가 나타남 (글자색상: rgb(13, 102, 221))
12	페이지 스크롤을 아래로 이동 시 내비게이션 바와 버튼 색상 전환
* No.2~9의 경우 각 란의 화살표 클릭 시 option은 아래로 내려옴	

화면설계서 주요서비스 (2)

▼ 주요서비스: 나의 만족도 예측서비스 화면

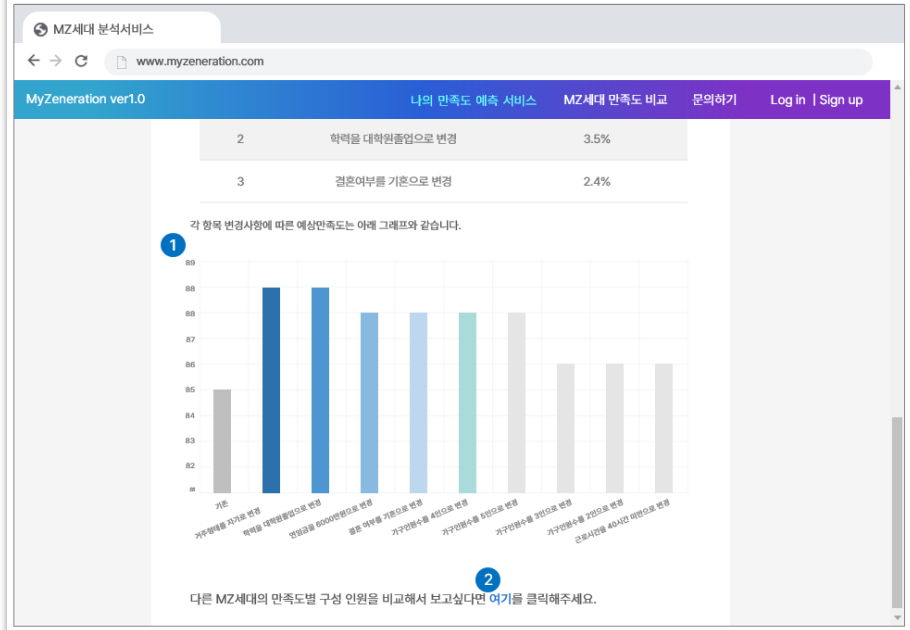
프로젝트명	MZ세대 분석서비스 “나는 어떤 MZ일까?”	설명	나의 만족도 예측 서비스 (스크롤 아래로)	페이지 #	2	페이지명	analysisMySat
화면경로	메인 화면 → 나의 만족도 예측 서비스						



No.	Description
1	학력, 전공, 거주지역, 연임금, 가구인원수, 거주형태, 근로시간, 결혼여부 8가지 항목 중 각 항목에 대한 점수를 백분율로 표기하며 이를 레이더차트로 나타냄
2	레이더차트(No.1)에 나타낸 항목 중 백분율이 100%인 항목에 대해 나열함 (글자색상: rgb(13, 102, 221))
3	2번 설명에서 나열한 항목 중 개선사항 적용 시 삶의 만족도 증가율이 높은 순으로 3개 항목을 테이블에 나타냄

▼ 주요서비스: 나의 만족도 예측서비스 화면

프로젝트명	MZ세대 분석서비스 “나는 어떤 MZ일까?”	설명	나의 만족도 예측 서비스 (스크롤 아래로)	페이지 #	2	페이지명	analysisMySat
화면경로	메인 화면 → 나의 만족도 예측 서비스						

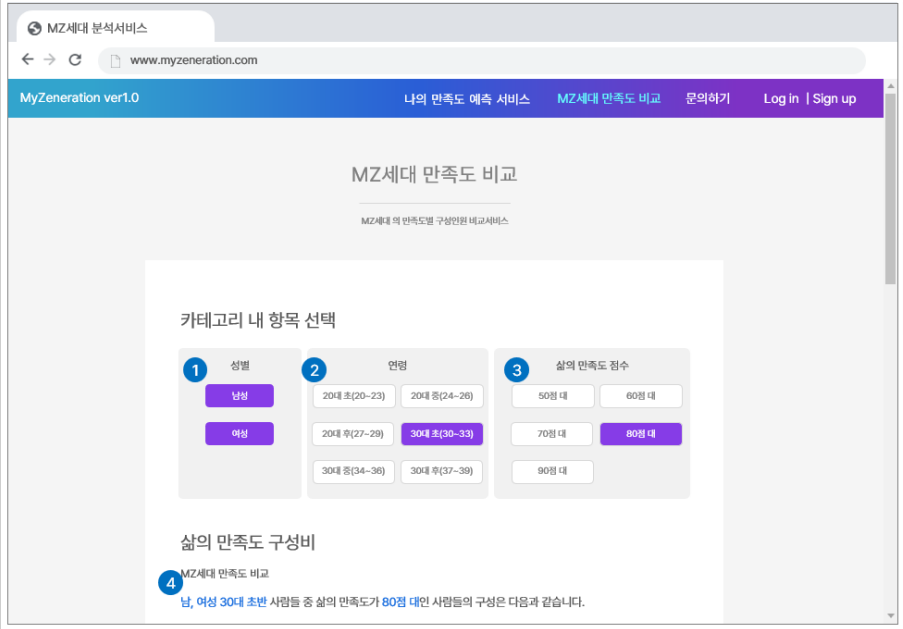


No.	Description
1	개선사항 적용 시 삶의 만족도 증가율이 높은 순으로 막대그래프에 나타내며 상위 5개 항목에 대해 막대의 색상을 ‘BFBFBF’, ' #2C71AC’, ' #4E96D2’, ‘#7EB3DE’, ' #BCD7EE’, ' #A9DBDA' 순으로 적용함
2	' 여기' 버튼 : 클릭 시 “MZ세대 만족도 비교” 화면으로 이동

화면설계서 주요서비스 (3)

▼ 주요서비스: MZ세대 만족도 비교 화면

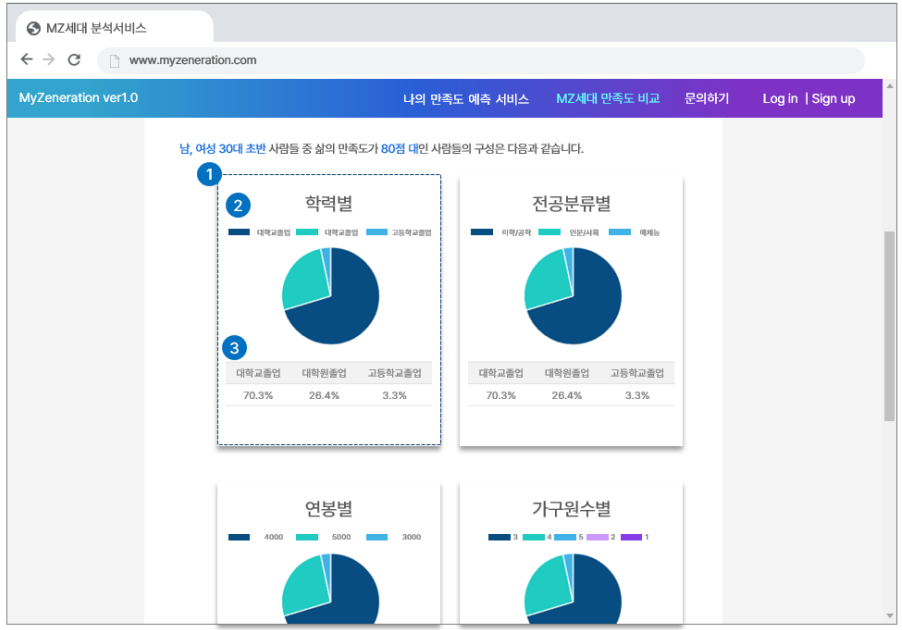
프로젝트명	MZ세대 분석서비스 “나는 어떤 MZ일까?”	설명	MZ세대 만족도 비교 서비스	페이지 #	3	페이지명	analysisMz
화면경로	메인 화면 → MZ세대 만족도 비교 서비스						



No.	Description
1	성별 카테고리 : 남성, 여성 항목이 있으며 중복 선택이 가능함
2	연령 카테고리 : 20대 초(20~23), 20대 중(24~26), 20대 후(27~29), 30대 초(30~33), 30대 중(34~36), 30대 후(37~39) 항목이 있으며 중복 선택이 가능함
3	삶의 만족도 카테고리 : 50점 대, 60점 대, 70점 대, 80점 대, 90점 대 항목이 있으며 중복 선택이 가능함
4	1~3번에서 선택한 항목이 텍스트 형태로 나타남 *선택한 항목은 글자 색상을 rgb(13, 102, 221)로 적용 *항목을 다르게 선택할 때마다 텍스트가 변환됨
* 1~3번에서 클릭된 버튼은 #863ce7 색상으로 변환	

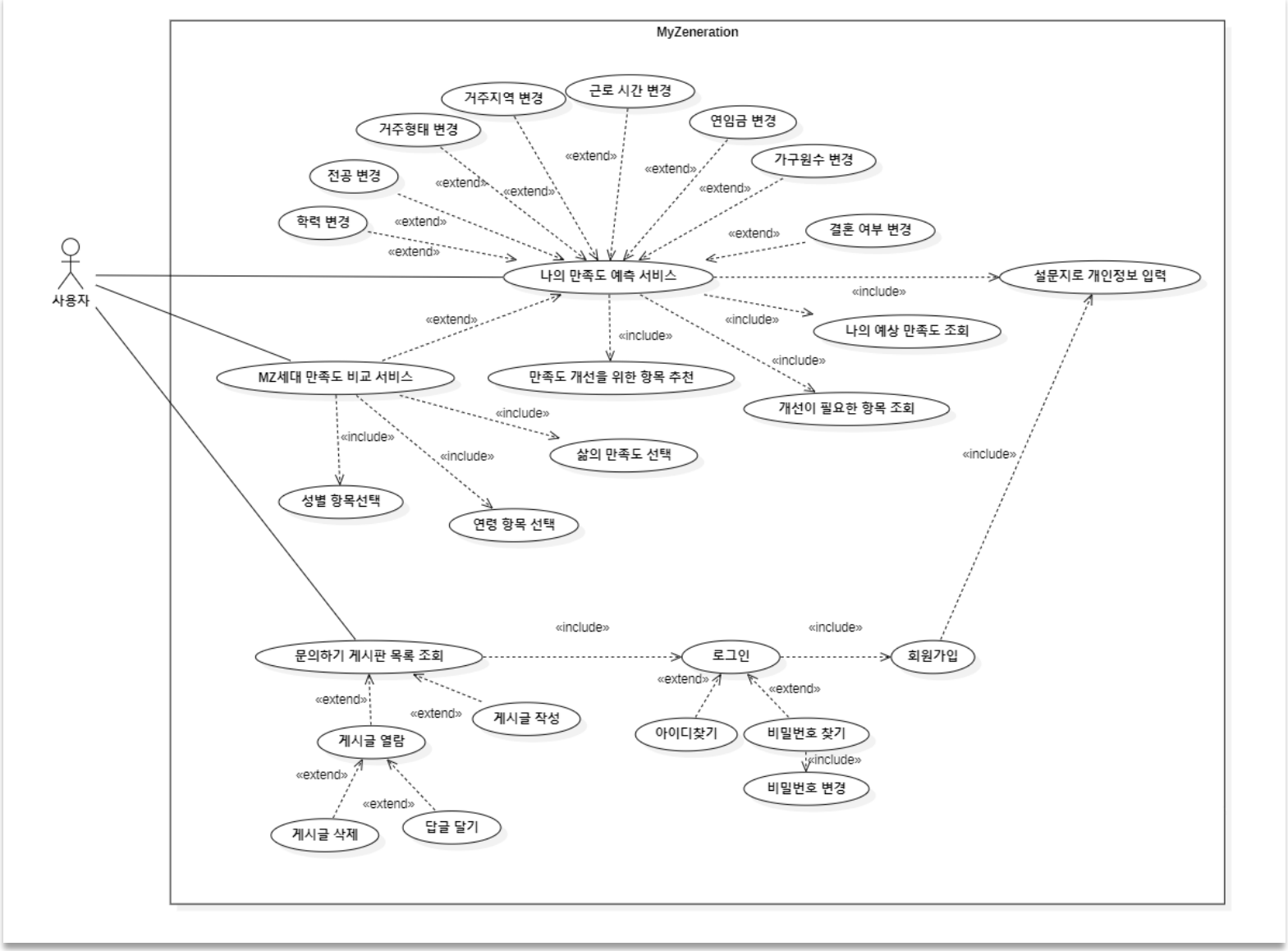
▼ 주요서비스: MZ세대 만족도 비교 화면

프로젝트명	MZ세대 분석서비스 “나는 어떤 MZ일까?”	설명	MZ세대 만족도 비교 서비스 (스크롤 아래로)	페이지 #	3	페이지명	analysisMz
화면경로	메인 화면 → MZ세대 만족도 비교 서비스						



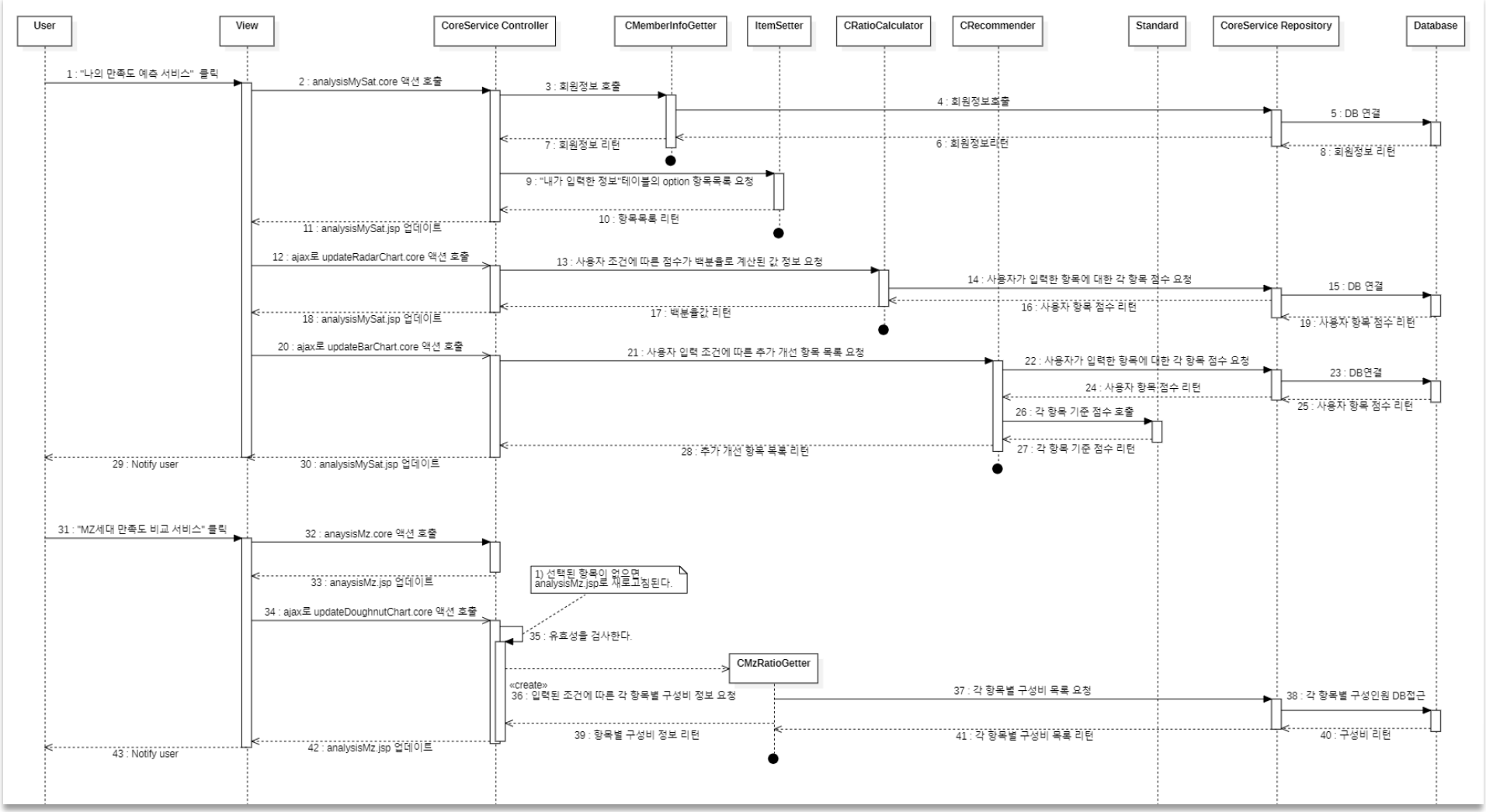
No.	Description
1	선택한 항목에 따라 학력별, 전공분류별, 연봉별, 가구원 수별, 거주지역별로 통계정보를 나타냄
2	통계정보는 도넛그래프 형태로 나타내며, 레이블은 그래프 상단에 위치함 통계 데이터 중 비율이 높은 순서대로 도넛 파이의 색상을 rgb(7,77,129)', 'rgb(32,203,194)', 'rgb(64,178,230) 로 적용함
3	비율에 대한 정보를 테이블 형태로 나타냄
*2,3번은 카테고리에서 항목을 다르게 선택할 때마다 해당 데이터로 테이블이 변환	

Usecase diagram



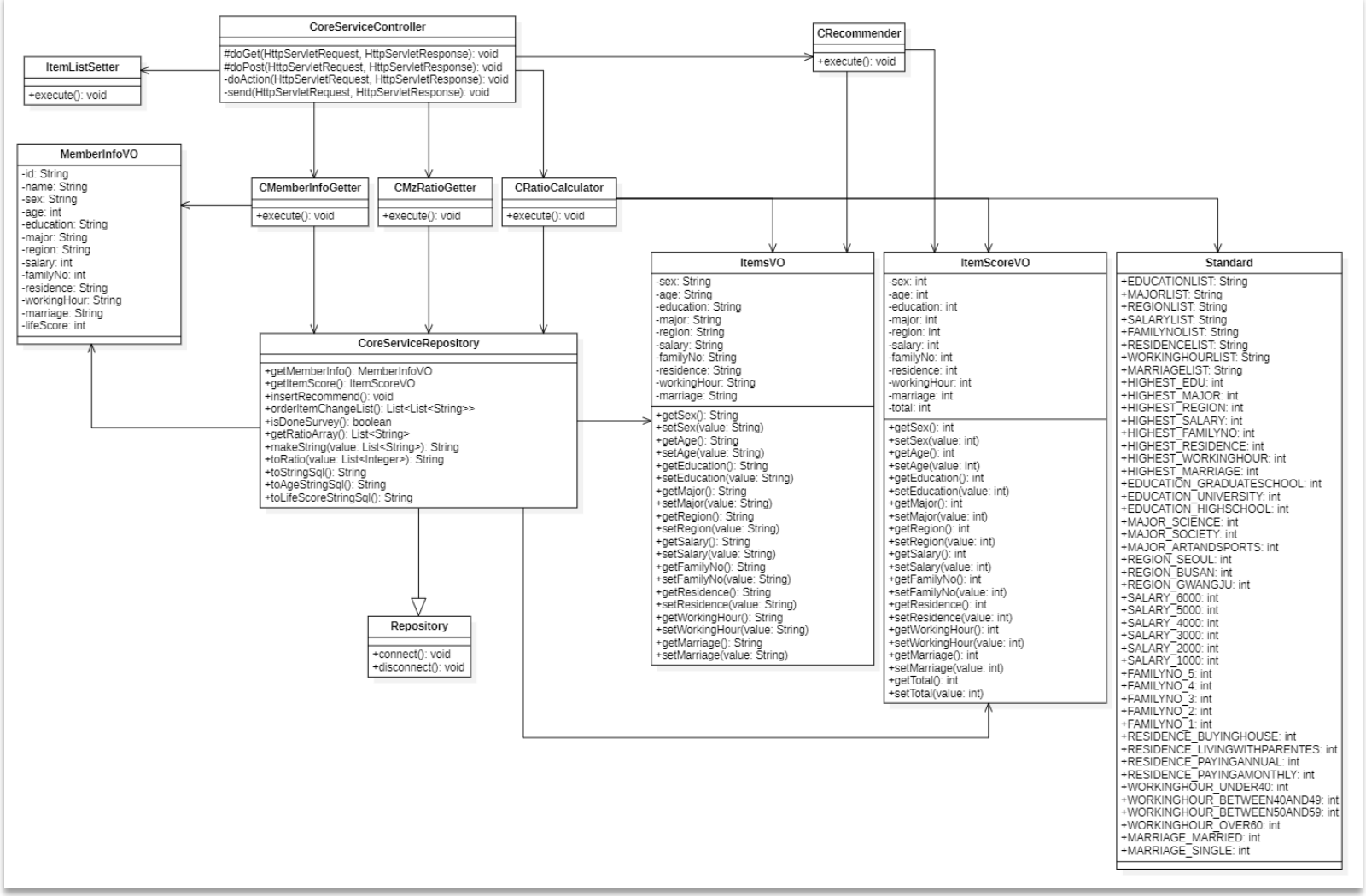
Sequence diagram

▼ [나의 만족도 예측 서비스], [MZ세대 만족도 분석 서비스] 시퀀스 다이어그램



Class diagram

▼ [나의 만족도 예측 서비스], [MZ세대 만족도 분석 서비스] 클래스 다이어그램



서비스 기능 (1)

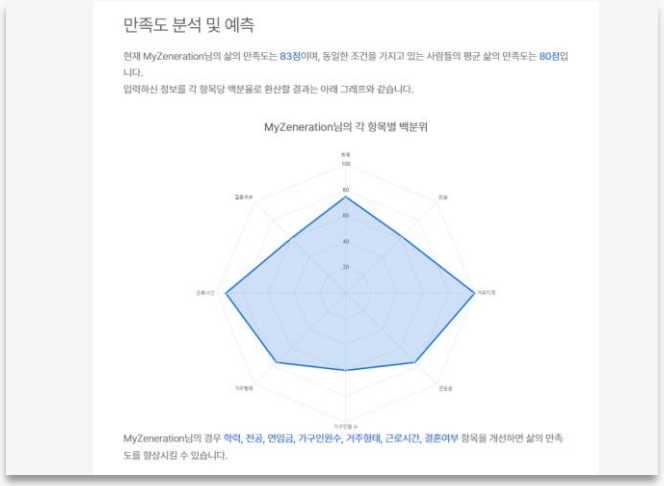
[1] 나의 만족도 예측 서비스

1. 만족도 분석 및 예측 → 레이더차트로 각 항목에 대한 백분율 표기, 개선이 필요한 항목 제시
2. 개선항목 중 만족도 증가율이 높은 순 3개 항목을 추천
3. 각 개선항목에 대한 만족도를 산출

** 테이블 내 조건은 변경 가능하며 변경 시 그래프와 표가 비동기적으로 변경되어야 함

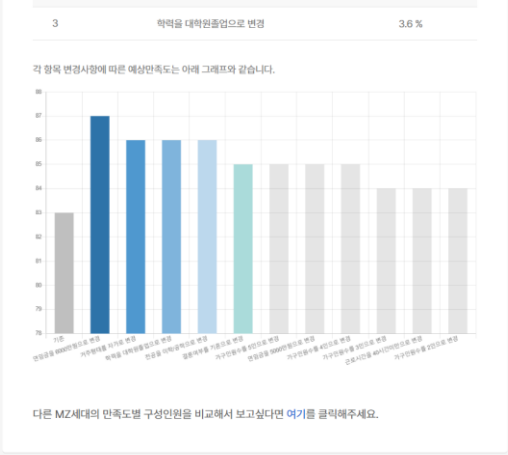
▼ [나의 만족도 예측서비스] 클릭 시

❶ 만족도 분석 및 예측 서비스



❷ 개선항목 상위 3개 추천

❸ 개선 항목에 대한 만족도 산출

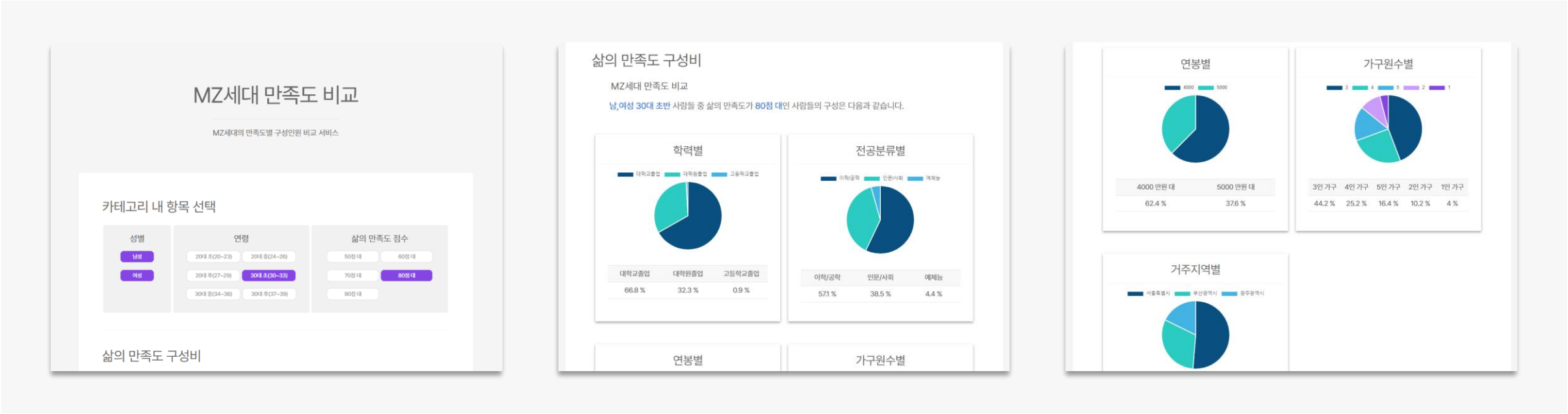


서비스 기능 (2)

[2] MZ세대 만족도 비교 서비스

1. 카테고리 내 성별, 연령, 삶의 만족도 점수에 따라 해당 조건에 부합하는 구성인원의 비율 정보 제공
2. 중복 선택 가능

** 카테고리 선택항목 변경 시 그래프와 표가 비동기적으로 변경되어야 함



소스 디렉토리

❶ Controller, Model, Service

```
MyZeneration_spring
  src/main/java
    com.project.MyZeneration_spring.controller
      BoardController.java
      CoreServiceController.java
      HomeController.java
      MemberController.java
    com.project.MyZeneration_spring.repository
      BoardRepository.java
      Connector.java
      CoreServiceRepository.java
      ICoreServiceRepository.java
      MemberRepository.java
    com.project.MyZeneration_spring.service
      BoardService.java
      CoreService.java   주요 서비스
      IBoardService.java
      ICoreService.java
      IMemberService.java
      MemberService.java
      Standard.java
    com.project.MyZeneration_spring.vo
      BoardInsertVO.java
      ItemScoreVO.java
      ItemsVO.java
      MemberInfoVO.java
      MemberJoinVO.java
      NestedSelectionVO.java
      SurveyVO.java
```

Controller

Repository

Service

VO

❷ View, resources

```
src/main/webapp/WEB-INF/views
  common
    footer.jsp
    footerLogOut.jsp
    head.jsp
    header.jsp
    headerLogOut.jsp
    indexHeader.jsp
    indexHeaderLogOut.jsp
  analysisMySat.jsp
  analysisMz.jsp
  bContentView.jsp
  bList.jsp
  bWriteView.jsp
  idFindResult.jsp
  idFindView.jsp
  index.jsp
  joinOk.jsp
  joinView.jsp
  loginView.jsp
  pwChangeOk.jsp
  pwChangeView.jsp
  pwFindView.jsp
  survey.jsp
  surveyComplete.jsp
src
  main
    webapp
      resources
        css
          font
          login.css
          main.css
          surveyForm.css
        img
          navbar-toggle-b.png
          navbar-toggle.png
        js
          analysisMySat.js
          analysisMz.js
          common.js
          jquery.counterup.min.js
          main.js
          needRequest.js
```

JSP

resources

소스코드 (1)

▼ 주요서비스 [나의 만족도 예측 서비스],[MZ세대 만족도 분석 서비스] 컨트롤러 (CoreServiceController)

```java
package com.project.MyZeneration_spring.controller;

import java.util.List;...

@Controller
@RequestMapping(value = "/core")
public class CoreServiceController {

  @Autowired
  CoreService coreService;

  Standard standard;

  @GetMapping("/analysisMySat")
  public String analysisMySat(@RequestParam(value = "memberId") String memberId, Model model, HttpSession session) {
    System.out.println("[Notice] CoreServiceController (analysisMySat)");
    model.addAttribute("memberInfo", coreService.getMemberInfo(memberId, session));
    model.addAttribute("educationList", Standard.EDUCATIONLIST);
    model.addAttribute("majorList", Standard.MAJORLIST);
    model.addAttribute("regionList", Standard.REGIONLIST);
    model.addAttribute("salaryList", Standard.SALARYLIST);
    model.addAttribute("familyNoList", Standard.FAMILYNOLIST);
    model.addAttribute("residenceList", Standard.RESIDENCELIST);
    model.addAttribute("workingHourList", Standard.WORKINGHOURLIST);
    model.addAttribute("marriageList", Standard.MARRIAGELIST);
    session.setAttribute("lifeScore", coreService.getMemberInfo(memberId, session).getLifeScore());
    return "analysisMySat";
  }

  @PostMapping("/updateRadarChart")
  public @ResponseBody List<List<String>> updateRadarChart(@RequestBody ItemsVO items) {
    System.out.println("[Notice] CoreServiceController (updateRadarChart)");
    return coreService.calculateRatio(items);
  }
```

CoreServiceController.java

```java
  @PostMapping("/updateBarChart")
  public @ResponseBody List<List<String>> updateBarChart(@RequestBody ItemsVO items, HttpSession session) {
    System.out.println("[Notice] CoreServiceController (updateBarChart)");
    return coreService.recommend(items, session);
  }

  @GetMapping("/analysisMz")
  public String analysisMz() {
    System.out.println("[Notice] CoreServiceController (analysisMz)");
    return "analysisMz";
  }

  @PostMapping("/updateDoughnutChart")
  public @ResponseBody List<List<String>> updateDoughnutChart(@RequestBody NestedSelectionVO lists) {
    System.out.println("[Notice] CoreServiceController (updateDoughnutChart)");
    return coreService.getMzRatio(lists);
  }

  @PostMapping("/updateMySatScore")
  public @ResponseBody int updateMySatScore(@RequestBody ItemsVO items, HttpSession session) {
    System.out.println("[Notice] CoreServiceController (updateMySatScore)");
    int result = coreService.updateMySatScore(items, (String) session.getAttribute("memberId"), session);
    return result;
  }
}
```

CoreServiceController.java

소스코드 (2)

▼ [나의 만족도 예측 서비스] 중 ajax를 이용한 비동기식 레이더 차트 (chart js) 데이터 요청 처리

```
function sendRadarAjaxRequest(){
  var myRadarChart;

  let selectedSexValue =sexSelect.value;
  let selectedAgeValue =ageSelect.value;
  let selectedEducationValue =educationSelect.value;
  let selectedMajorValue =majorSelect.value;
  let selectedRegionValue =regionSelect.value;
  let selectedSalaryValue =salarySelect.value;
  let selectedFamilyNoValue =familyNoSelect.value;
  let selectedResidenceValue =residenceSelect.value;
  let selectedWorkingHourValue =workingHourSelect.value;
  let selectedMarriageValue =marriageSelect.value;

  if(myRadarChart){
    myRadarChart.destroy();
  }

  $.ajax({
    type:'post',
    url:'./updateRadarChart',
    contentType:"application/json;charset=UTF-8",
    data: JSON.stringify({
      'sex':selectedSexValue,
      'age':selectedAgeValue,
      'education':selectedEducationValue,
      'major':selectedMajorValue,
      'region':selectedRegionValue,
      'salary':selectedSalaryValue,
      'familyNo':selectedFamilyNoValue,
      'residence':selectedResidenceValue,
      'workingHour':selectedWorkingHourValue,
      'marriage':selectedMarriageValue
    }),
```

❶ Post 방식으로 데이터 처리 요청

analysisMySat.js

```
@PostMapping("/updateRadarChart")
public @ResponseBody List<List<String>> updateRadarChart(@RequestBody itemsVO items) {
  System.out.println("[Notice] CoreServiceController (updateRadarChart)");
  return coreService.calculateRatio(items);
}
```

❷ Response

```
    success: function(response){
      console.log("AJAX 요청 성공");
      var radarRatioResult = response[0].map(function(item) {
        return parseInt(item, 10);
      });
      var badItemList = response[1].map(function(item) {
        return String(item);
      });
      var totalScore = response[2].map(function(item) {
       return String(item);
      });
      itemList.textContent = badItemList;
      averageScore.textContent=totalScore+'점';

      var itemRatioData={
        labels:['학력','전공','거주지역','연임금','가구인원 수','거주형태','근로시간','결혼여부'],
        datasets:[{
          data:radarRatioResult,
          fill: true,
          backgroundColor: 'rgba(13, 102, 225, 0.2)',
          borderColor: 'rgb(13, 102, 221)',
          pointBackgroundColor: 'rgb(54, 162, 235, 0.2)',
          pointBorderColor: '#fff',
          pointHoverBackgroundColor: '#fff',
          pointHoverBorderColor: 'rgb(54, 162, 235)'
        }]
      };
```

❸ 서버로부터 수신한 response 데이터

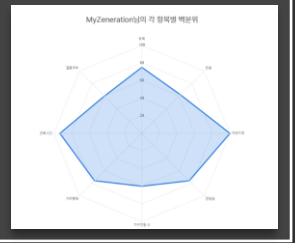
❹ Chart js

analysisMySat.js

```
    var radarChart = document.getElementById('radar').getContext('2d');
    myRadarChart = new Chart(radarChart, {
      type: 'radar',
      data: itemRatioData,
      options: {
        legend:{
          display:false
        },
        scale: {
          ticks: {
            beginAtZero: true,
            min: 0,
            max: 100,
            stepSize: 20,
            suggestedMin: 0,
            suggestedMax: 100,
          }
        }
      }
    });},
  error:function(data){
    console.log(data);
    console.log("AJAX 요청 실패");
  }
});
}// sendRadarAjaxRequest() 끝

sendRadarAjaxRequest();

document.addEventListener('DOMContentLoaded', function() {
  elements.forEach(element => {
    const el = document.getElementById(element.id);
    el.addEventListener('change', sendRadarAjaxRequest);
    el.addEventListener('change', sendBarAjaxRequest);
  });
});
```

❺ Chart js

❻ 결과



*카테고리 항목 값이 변경될 때마다 함수 호출

analysisMySat.js

소스코드 (3)

▼ [MZ세대 만족도 비교 서비스] 중 카테고리 내 중복선택 처리

```jsp
<form action="/core/analysisMz" method="post" name="category">
  <div class="choice">
    <div class="select width-narrow">
      <h4>성별</h4>
      <input type="checkbox" id="sexMale" name="sex" value="남"><label for="sexMale">남성</label><br>
      <input type="checkbox" id="sexFemale" name="sex" value="여"><label for="sexFemale">여성</label>
    </div>
    <div class="select">
      <h4>연령</h4>
      <input type="checkbox" id="age20_1" name="age" value="20대 초반"><label for="age20_1">20대 초(20~23)</label>
      <input type="checkbox" id="age20_2" name="age" value="20대 중반"><label for="age20_2">20대 중(24~26)</label><
      <input type="checkbox" id="age20_3" name="age" value="20대 후반"><label for="age20_3">20대 후(27~29)</label>
      <input type="checkbox" id="age30_1" name="age" value="30대 초반"><label for="age30_1">30대 초(30~33)</label><
      <input type="checkbox" id="age30_2" name="age" value="30대 중반"><label for="age30_2">30대 중(34~36)</label>
      <input type="checkbox" id="age30_3" name="age" value="30대 후반"><label for="age30_3">30대 후(37~39)</label><
    </div>
    <div class="select wider">
      <h4>삶의 만족도 점수</h4>
      <input type="checkbox" id="score5" name="lifeScore" value="50"><label for="score5">50점 대</label>
      <input type="checkbox" id="score6" name="lifeScore" value="60"><label for="score6">60점 대</label>
      <input type="checkbox" id="score7" name="lifeScore" value="70"><label for="score7">70점 대</label>
      <input type="checkbox" id="score8" name="lifeScore" value="80"><label for="score8">80점 대</label>
      <input type="checkbox" id="score9" name="lifeScore" value="90"><label for="score9">90점 대</label>
      <input type="checkbox"><label style="border:none;"></label>
    </div>
  </div>
</form>
```

❷CoreService의 getMzRatio 메소드 호출

analysisMz.jsp

```java
@GetMapping("/analysisMz")
public String analysisMz() {
  System.out.println("[Notice] CoreServiceController (analysisMz)");
  return "analysisMz";
}

@PostMapping("/updateDoughnutChart")
public @ResponseBody List<List<String>> updateDoughnutChart(@RequestBody NestedSelectionVO lists) {
  System.out.println("[Notice] CoreServiceController (updateDoughnutChart)");
  return coreService.getMzRatio(lists);
}
```

❶ Controller의 analysisMz 메소드 호출 & updateDoughnutChart 메소드 호출 (페이지 로딩 시 호출하도록 개발)

CoreServiceController.java

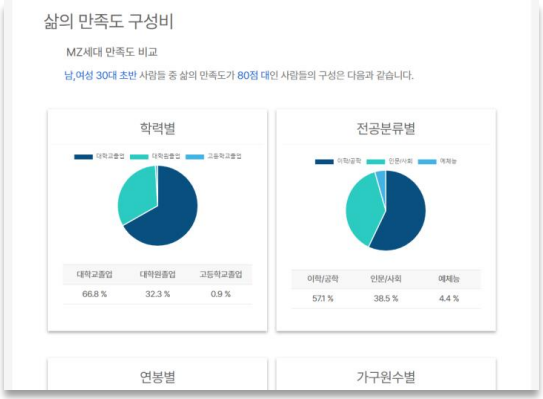
```java
public List<List<String>> getMzRatio(NestedSelectionVO lists){

  String[] sexList = lists.getSexList();
  String[] ageList = lists.getAgeList();
  String[] lifeScoreList = lists.getLifeScoreList();

  List<String> educationList = csRepository.getRatioArray(sexList, ageList, lifeScoreList, "education");
  List<String> majorList = csRepository.getRatioArray(sexList, ageList, lifeScoreList, "major");
  List<String> salaryList = csRepository.getRatioArray(sexList, ageList, lifeScoreList, "salary");
  List<String> familyNoList = csRepository.getRatioArray(sexList, ageList, lifeScoreList, "familyNo");
  List<String> regionList = csRepository.getRatioArray(sexList, ageList, lifeScoreList, "region");

  List<List<String>> itemLists = new ArrayList<List<String>>();
  itemLists.add(educationList);
  itemLists.add(majorList);
  itemLists.add(salaryList);
  itemLists.add(familyNoList);
  itemLists.add(regionList);

  return itemLists;
}
}
```

❸ coreSeriveRepositoy의 getRatioArray 메소드 호출

CoreService.java

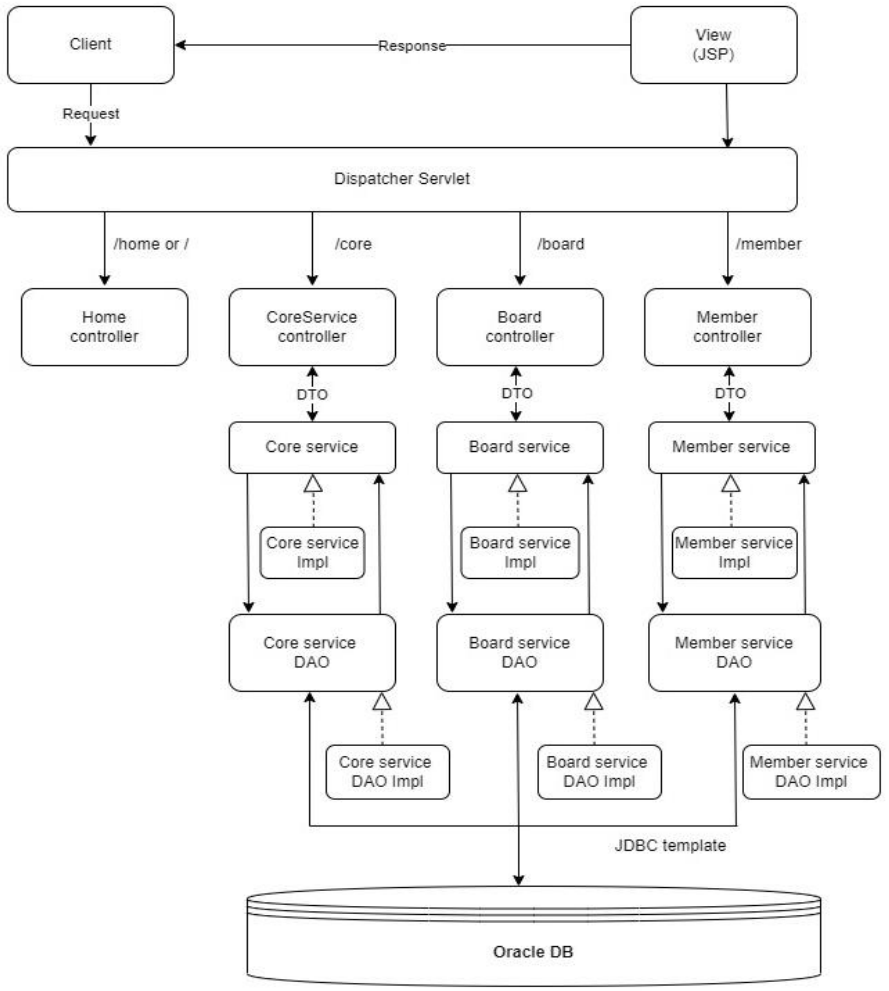
```java
public List<String> getRatioArray(String[] sexList, String[] ageList, String[] lifeScoreList, String category){
  String sql = "";
  String sex = toStringSql(sexList);
  String age = toAgeStringSql(ageList);
  String lifeScore = toLifeScoreStringSql(lifeScoreList);

  if (category.equals("education")) {
    sql = "select count(*) as count, education as category  from bigdatatotal where sex IN " + sex + " and age IN " + age
        + " and lifeScore IN " + lifeScore + " group by education order by count desc";
  } else if (category.equals("major")) {
    sql = "select count(*) as count, major as category from bigdatatotal where sex IN " + sex + " and age IN " + age
        + " and lifeScore IN " + lifeScore + " group by major order by count desc";
  } else if (category.equals("salary")) {
    sql = "select count(*) as count, salary as category from bigdatatotal where sex IN " + sex + " and age IN " + age
        + " and lifeScore IN " + lifeScore + " group by salary order by count desc";
  } else if (category.equals("familyNo")) {
    sql = "select count(*) as count, familyNo as category from bigdatatotal where sex IN " + sex + " and age IN " + age
        + " and lifeScore IN " + lifeScore + " group by familyNo order by count desc";
  } else if (category.equals("region")) {
    sql = "select count(*) as count, region as category from bigdatatotal where sex IN " + sex + " and age IN " + age
        + " and lifeScore IN " + lifeScore + " group by region order by count desc";
  }
  List<Integer> countList = jdbcTemplate.query(sql, new RowMapper<Integer>() {
    @Override
    public Integer mapRow(ResultSet rs, int rowNum) throws SQLException {
      return rs.getInt("count");
    }
  });
  List<String> itemList = jdbcTemplate.query(sql, new RowMapper<String>() {
    @Override
    public String mapRow(ResultSet rs, int rowNum) throws SQLException {
      return rs.getString("category");
    }
  });

  String ratios = toRatio(countList);
  String items = String.join(",", itemList);

  List<String> totalList = new ArrayList<String>();
  totalList.add(items);
  totalList.add(ratios);

  return totalList;
}
```

CoreServiceRepository.java



소프트웨어 아키텍처

[Spring Framework]



09

MyZeneration

프로그램 시연

향후계획

1) 회원가입 후 만족도 조사를 바로 실행하지 않았을 경우, 재로그인 시 만족도 조사 페이지로 이동할 수 있도록 개발 예정

2) 세대를 베이비붐 세대로 확장하여 삶의 만족도 분석서비스 제공

프로젝트 수행소감

(1) 네트워크에 대한 깊은 공부의 필요성을 느낌.

ajax로 비동기식 데이터 처리 시 송수신이 원활치 않아 개발에 어려움이 있었음. console.log로만 상태를 확인할 뿐 그 외에는 어디서 뭘 확인할 수 있는지 조차 몰랐음. 여러 번의 시도 끝에 드디어 서버에서 송신한 데이터가 브라우저 개발자도구의 네트워크 탭에서 데이터의 정상적인 송/수신이 확인됐음. 이 때 비로소 네트워크 공부에 대한 필요성을 확실히 느끼게 됨. 이에 네트워크 공부를 반드시 수행해야겠다는 다짐을 함

(2) 시스템 동작원리에 대한 공부 필요성을 느낌

에러가 발생했을 때 시스템에 대한 이해도가 부족하여 디버깅하는데 시간이 오래 걸리는 상황이 발생한 적이 있음. 시스템 동작원리를 더 깊이 알았다면 디버깅을 더 정확하고 빠르게 해결했을 텐데 하는 아쉬움이 크게 남음. 이에 시스템 동작원리에 대한 이해도를 높이기 위한 공부를 수행해야겠다는 다짐을 함.

감사합니다.