REPORT

[소프트웨어 아키텍쳐]

〈통합 종교 검색 시스템〉

■ 과 목 : 소프트웨어 아키텍쳐

■ 담당교수 : 최은미 교수님

■ 학 과 : 소프트웨어학부

■ 학 번 : 20152857

■ 이 름 : 정준권

■ 제출날짜 : 2020.05.26

내용

1. 목표

1.1 Vision

사람들이 종교를 갖고, 자신의 신앙을 키워 가기 위해서 종교시설을 찾게 된다. 하지만, 가려는 종교시설이 좋은지, 나쁜지 알 수 없다. 심지어 해당 종교시설이 사이비일 가능성도 존재한다. 일반적인 사람들은 종교시설의 좋고 나쁨을 혼자서 가늠하기 어렵다. 이를 해결하기 위하여 통합 종교 검색 시스템을 설계하여 각 종교시설에 대한 평을 듣고, 사람들은 더욱 쉽고, 안전하게 시설을 선택할 수 있게 된다.

1.2 Scope

일반적인 중개 플랫폼의 형태를 따른다. 시설을 등록하는 유저와 등록된 시설을 검색하는 유저가 존재한다. 시스템은 등록한 시설을 데이터베이스에 저장하고, 해당 데이터를 검색에 따라서 보여주는 역할을 한다.

1.3 AS-IS

종교 중 하나인 기독교는 교회를 검색할 수 있는 사이트가 존재한다.

1.4 TO-BE

기존의 교회 검색 사이트는 대한예수장로교, 감리교등 특정 교리에 대해서만 검색이 된다. 이 시스템은 교리 등과 같은 기존의 규칙에서 벗어나 등록된 어떠한 종교시설도 검색이 가능하게 한다. 또한, 해당 종교시설에 대한 평가 즉, 리뷰를 남길 수 있도록 해주는 시스템이다. 또한, 등록한 종교시설에 대한 설명과 영상 샘플 등을 추가할 수 있다.

2. 특성 및 개요

2.1 개요 및 특성 리스트

2.1.1 개요

사람들은 누구나 죽음에 대한 불안감을 간직하고 있다. 죽음에 대한 어떠한 정보도 없기 때문이다. 사람들은 이를 해소하기 위하여 삶의 형태를 만들어 나간다.

죽음에 대한 두려움을 해소하는 가장 큰 방법은 신을 믿는 것이고, 이는 다양한 종교로 나타난다. 또한, 해당 종교의 신앙심을 더욱 굳건히 하기 위하여 종교시설에 모이게 된다.

하지만, 이를 악용하여 이득을 취하는 종교시설도 존재한다. 이에 사람들이 종교시설에 가기 전에 그 종교시설이 어떤 지 알 수 있다면, 더 안전하고, 유익하게 신앙심을 키워나갈 수 있지 않을까 하는 생각에서 해당 시스템은 시작되었다.

2.1.2 특성

통합 종교 검색 시스템은 사용자가 원하는 종교의 카테고리를 선택하여 검색할 수 있다.

통합 종교 검색 시스템은 사용자의 위치에 기반하여 근처의 종교시설을 검색할 수 있다.

통합 종교 검색 시스템은 종교시설에 대한 평가를 사용자가 입력할 수 있다.

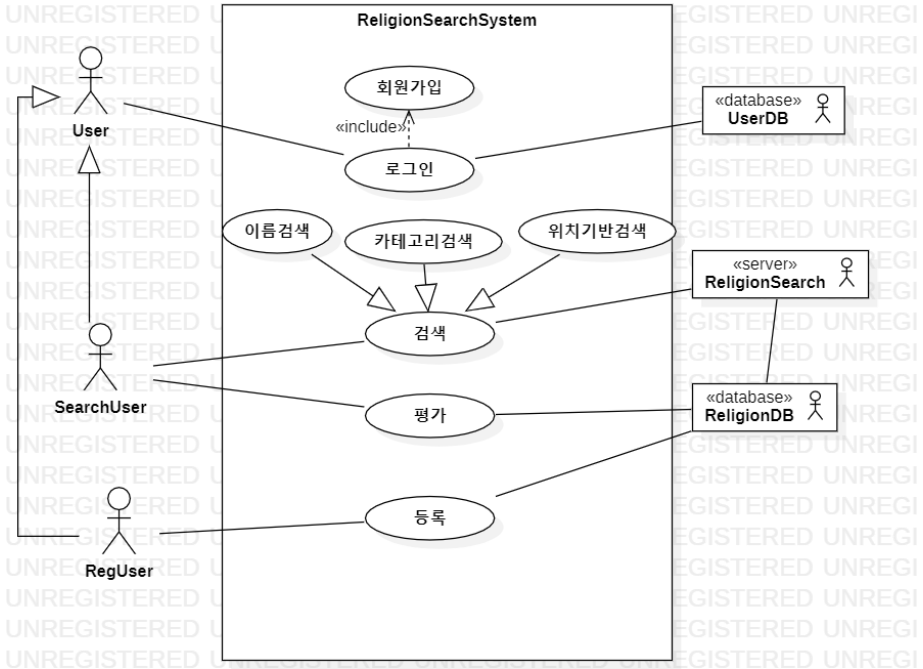
다만, 이 경우, 해당 평가를 악용하는 사례가 발생할 수 있기 때문에, 실명제를 사용한다.

통합 종교 검색 시스템은 사용자가 종교시설을 등록할 수 있다.

통합 종교 검색 시스템은 사용자가 등록한 종교시설에 한해 관리할 수 있다.

다만, 이 경우, 사용자의 무분별한 등록이 발생할 수 있기 때문에, 해당 종교시설에 대한 정보를 입력하도록 한다.

2.2 Use Case Diagram



2.3 기능적인 요구사항

번호	유스케이스명	요구사항	중요도
FR1	로그인	사용자가 아이디와 비밀번호를 입력한다. 이를 서버의 데이터베이스에 존재하는 지 확인 후, 사용자에게 접속 권한의 여부를 알려준다.	3
FR2	회원가입	사용자가 시스템에서 자신의 아이디 및 비밀번호 등 개인 정보를 입력한다. 이를 서버의 데이터베이스에 저장한다.	3
FR3	검색	사용자는 찾고자 하는 종교를 검색한다. 이는 총 3가지 방식으로 진행된다.	5
FR4	위치기반검색	사용자는 자신이 검색한 위치 반경내에 존재하는 종교시설을 검색한다. 서버의 데이터베이스에서 일치하는 종교시설을 보여준다.	5
FR5	카테고리검색	사용자는 검색할 종교의 분류를 입력한다. 서버의 데이터베이스에서 분류에 일치하는 종교시설을 보여준다.	5
FR6	이름검색	사용자는 검색할 종교시설의 이름을 입력한다. 서버의 데이터베이스에서 이름과 일치하는 종교시설을 보여준다.	5
FR7	평가	사용자는 종교시설의 평가를 남길 수 있다. 평가는 실명으로 이루어지며, 이는 서버의 데이터베이스에 저장된다.	2
FR8	등록	사용자는 종교시설을 등록할 수 있다. 종교시설의 분류, 이름, 위치 등을 입력할 수 있고, 무분별한 등록을 방지하기 위하여 신뢰할 수 있는 증명서를 확인한다. 이를 서버의 데이터베이스에 저장한다.	5

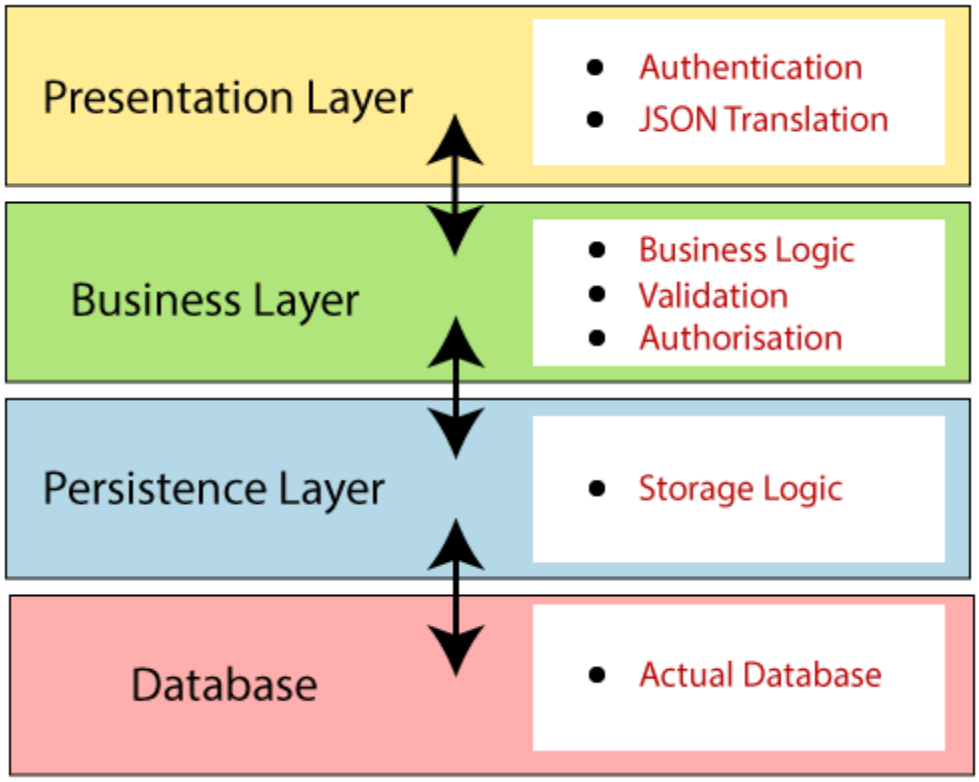
2.4 비기능적인 요구사항

번호	퀄리티명	요구사항	중요도
NFR1	성능	사용자의 트랜잭션을 0.1초 내로 처리할 수 있어야 한다.	4
NFR2	확장가능성	시스템의 기능이 다양해질 필요성이 존재하기 때문에, 시스템의 구조는 유연해야 하고, 확장에 용이해야 한다.	4
NFR3	규모확장	사용자의 증가에 따라 시스템의 규모가 확장되어야 하기 때문에, 규모확장에 용이해야 한다.	2
NFR4	편리성	사용자가 기능적인 요구사항을 쉽게 이행할 수 있도록 UI측면에서 개선해야 한다.	3

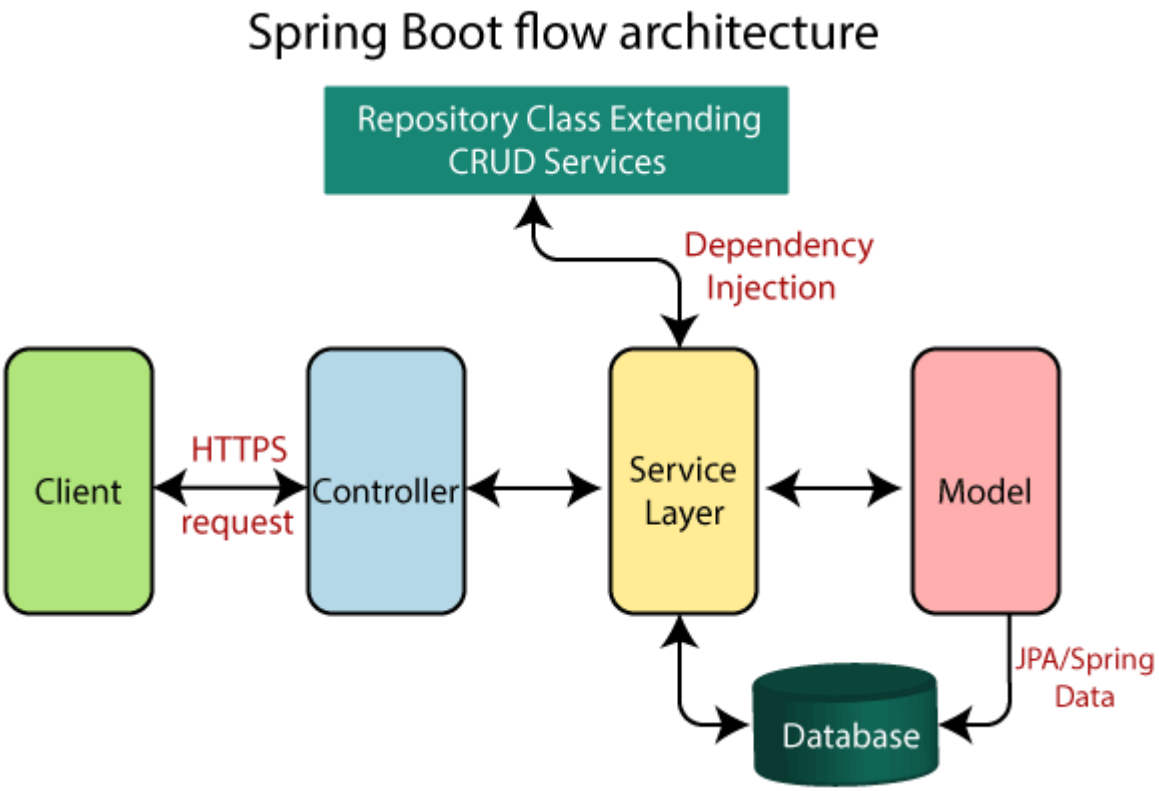
3. Framework 및 기술

3.1 사용하려는 Framework

3.1.1 Java Spring Boot



- 스프링 부트는 총 4단계의 레이어로 구성되어 있다.
  - Presentation layer: 현재 레이어는 HTTP 요청을 처리하고, JSON 매개변수를 객체로 변환하고, 요청을 인증하고, 이를 비즈니스 레이어로 보내는 역할을 한다. 해당 레이어는 프론드엔드 부분에 해당한다.
  - Business layer: 비즈니스 레이어는 모든 비즈니스 로직을 처리한다. 서비스 클래스들로 구성되어 있으며, 데이터 엑세스 레이어에서 제공받은 서비스들을 이용한다. 또한, 인증과 판별을 수행한다.
  - Persistence layer: 해당 레이어는 모든 저장 로직을 포함하고 잇고, 비즈니스 객체를 데이터베이스에 저장할 수 있는 데이터 형태로 변환하는 역할을 한다.
  - Database layer: 데이터베이스 레이어는 CRUD 기능을 수행한다.



- 스프링 부트는 Spring MVC, Spring Data 등과 같은 스프링 관련 모듈을 사용한다. 다만 스프링 부트는 DAO 관련된 클래스가 필요하지 않다.
- 데이터 엑세스 레이어를 생성하고, CRUD 기능을 수행한다.
- 클라이언트는 HTTP 요청을 보내게 된다.
- 해당 요청은 컨트롤러로 가고, 컨트롤러는 요청을 매핑하고, 이를 처리한다. 이후, 서비스 레이어에 콜을 보낸다.
- 서비스 레이어에서는 모든 비즈니스 로직을 처리한다. 데이터를 JPA로 매핑하는 작업을 수행한다.

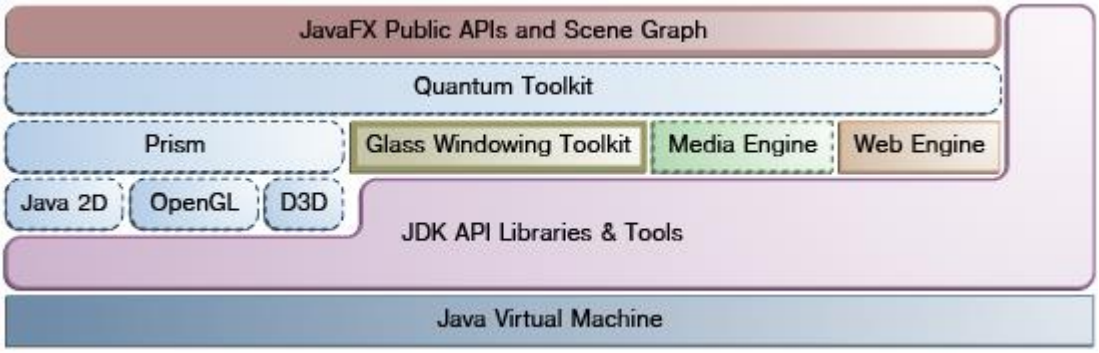
- 스프링부트는 단독 실행되는, 실행하기만 하면 되는 상용화 가능한 수준의 스프링 기반 어플리케이션을 쉽게 만들어낼 수 있다. 최소한의 설정으로 스프링 플랫폼과 서드파티 라이브러리들을 사용할 수 있도록 하고 있다.
- 스프링부트의 기능
  - 단독실행가능한 스프링애플리케이션을 생성한다.
  - 내장형 톰캣, 제티 혹은 언더토우를 내장하고 있다.
  - 기본설정되어 있는 'starter' 컴포넌트들을 쉽게 추가할 수 있다.

- 가능한 자동 설정되어 있다.
- 상용화에 필요한 통계, 상태 점검 및 외부 설정을 제공한다.
- 설정을 위한 XML 코드를 생성하거나 요구하지 않음

- 스프링 부트를 이용하여 서버를 배포하고, 비즈니스 로직을 처리한다. 이후, REST API 를 이용하여 클라이언트와 서버와의 통신을 용이하게 하는 것을 목표로 한다.

3.1.2 Java FX

- 자바 FX 는 개발자들이 플랫폼 간 일관성 있게 작동하는 풍부한 클라이언트 애플리케이션을 만들고, 배치할 수 있도록 설계된 Java 라이브러리 모음이다.



- 해당 그림은 JavaFX 플랫폼의 아키텍처 구성요소를 보여준다. JavaFX 공용 API 아래에는 JavaFX 를 실행하는 엔진이 있다. 이것은 프리즘이라 불리는 새로운 JavaFX 고성능 그래픽 엔진, 글래스라고 불리는 작고 효율적인 윈도우 시스템, 미디어 엔진, 그리고 웹 엔지을 포함하는 하위 컴포넌트로 구성되어 있다.

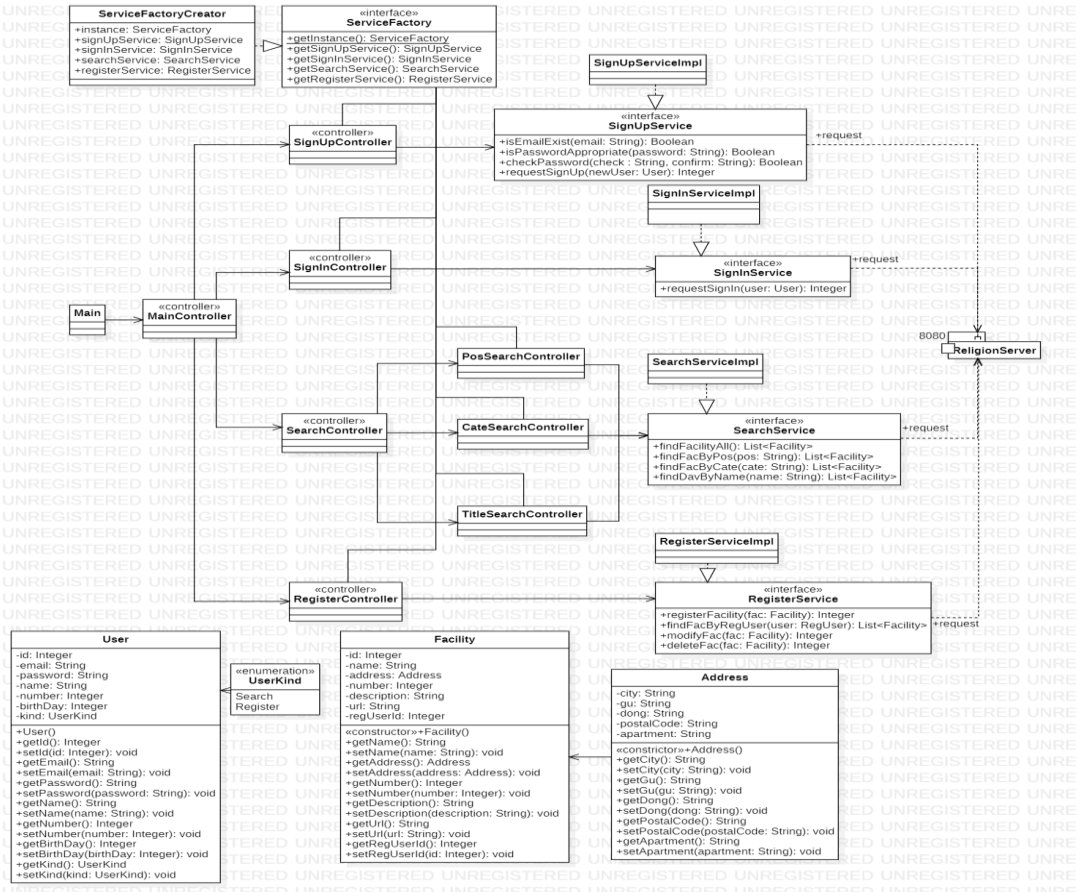
- Java public APIs
  - 제네릭, 주석 및 멀티스레딩과 같은 강력한 Jvava 기능의 사용을 허락한다.
  - 웹 개발자가 Groovy 및 JavaScript 와 같은 다른 JVM 기반 동적 언어의 JavaFx 를 더 쉽게 사용할 수 있다.
  - Java 개발자들이 크거나 복잡한 JavaFX 응용프로그램을 쓰기 위해 Groovy 와 같은 다른 시스템 언어를 사용하도록 한다.
  - 여러 방식의 바인딩을 허용한다. 대체언어는 해당 바인딩 라이브러리를 사용하여 JavaFX 스크립트와 유사한 바인딩 구문을 사용할 수 있다.
  - 데이터 모델에 사용자 인터페이스를 연결하고, 해당 모델의 변경사항을 관찰하며, 그에 따라 해당 UI 제어를 업데이트할 수 있다.

- JavaFX API 를 통해 이용할 수 있는 JavaFX UI 컨트롤은 씬 그래프의 노드를 이용해 구축한다. 이는 시각적으로 풍부한 기능을 충분히 이용할 수 있고, 다른 플랫폼에서 이동 가능하다. 해당 그림은 UI 컨트롤의 일부이다.
- JavaFX 를 통하여 클라이언트의 인터페이스를 구현하는 것을 목표로 한다.

4. 시스템 구성도 및 설명

4.1 Class Diagram

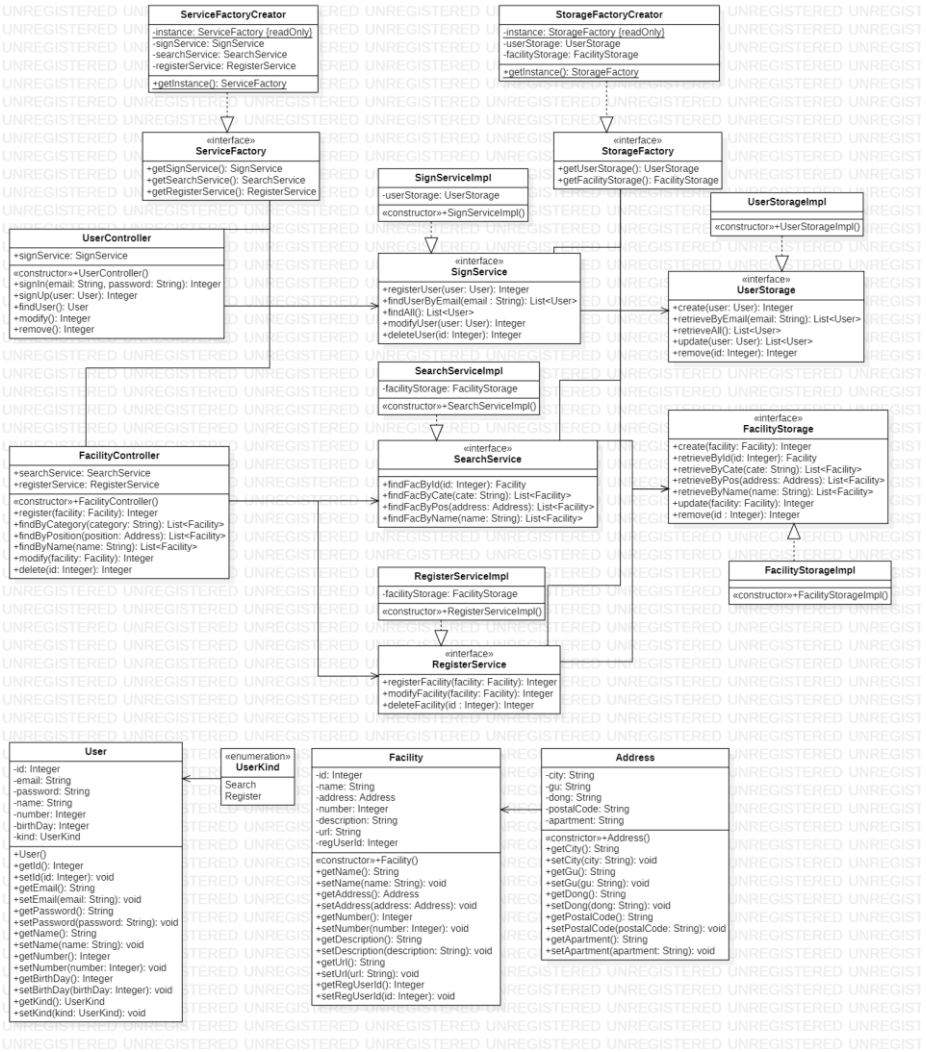
4.1.1 Religion Client



-

- Religion System은 메인에서 실행된다. 시스템은 직접적으로 데이터를 가지고 있지 않으며, 컨트롤러에서 데이터를 객체 형태로 변환한 뒤, server에 데이터를 보내고, 요청받는다. 요청시의 작업은 비동기로 처리한다.
- 컨트롤러는 Scene Builder를 이용하여 구축한다. 이에 따라 클래스 다이아그램에서는 직접 표현하지 않는다. 이후 수정을 거쳐 추가하도록 한다.
- 팩토리 패턴을 이용하여, 서비스 객체를 생성하는 생성 클래스를 따로 두었고, 인터페이스를 사용하여 호출함으로써, 낮은 의존도를 가진다.

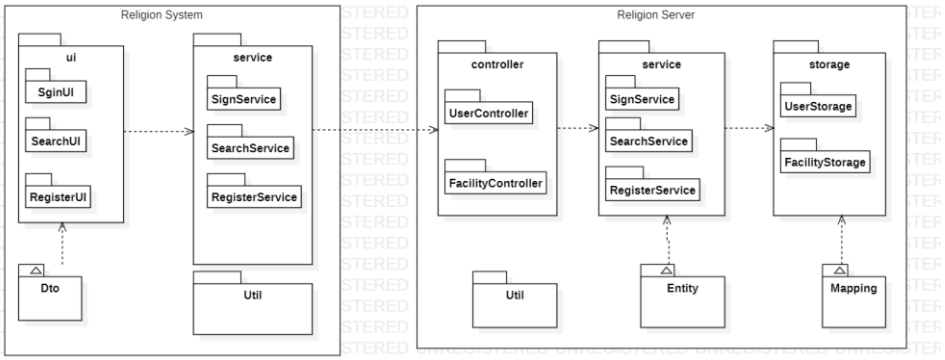
4.1.2 Religion Server



-

- 서버는 클라이언트에서 보내는 요청을 처리한다. 클라이언트가 보낸 요청은 컨트롤러의 매개변수로 들어오게 되며, 해당 매개변수로부터 데이터베이스에 저장하는 일련의 비즈니스 로직을 담당한다.
- 팩토리 패턴을 통하여 서비스, 스토리지 객체를 생성하는 클래스를 생성하였다.
- 인터페이스를 통하여 작업을 함으로써, 낮은 의존도를 유지하였다.

4.2 Deployment View



- 클라이언트와 서버를 구분하였다.
- 클라이언트는 UI와 Service 2개의 레이어로 구분하였고, 작업을 보완하는 Util 패키지와 UI에서 사용할 Dto 객체 패키지가 존재한다.
- 서버는 컨트롤러를 통해서 클라이언트의 요청을 처리한다. 서비스 패키지는 모든 비즈니스 로직을 처리한다. 스토리지 패키지는 데이터베이스에 저장을 요청하는 로직을 담당한다. 서비스 패키지는 entity 패키지를 사용하여 객체를 모델링하고, 이를 데이터베이스에 저장하기 위한 매핑 패키지가 존재한다. 또한, JSON 파일을 객체 형태로 변환하는 Util 패키지가 존재한다.

5. 사용하고자 하는 Architecture Style

5.1 Layered Architecture

Java로 구현하는 Religion System은 UI와 Service를 각각의 패키지로 나누어 구현한다. UI 패키지는 시스템의 메인 화면, 로그인 화면, 회원가입 화면, 검색화면, 등록 화면 클래스를 포함하고 있으며, ui 패키지는 ui측면에 대해서만 구현한다.

Service 패키지는 로그인, 회원가입, 검색, 등록에 대한 비즈니스 로직 클래스를 포함한다. Service 패키지는 비즈니스 로직 및 서버와의 통신을 위한 내용으로만 구현한다.

시스템에 대한 전체적인 구조는 ui -> service -> server-service -> server-storage 구조로 이루어져 있다.

각 패키지는 interface를 통하여 호출이 가능하다.

5.2 Client-Server Architecture

통합 종교 검색 시스템은 사용자가 사용하는 Front-end(Client)와 사용자가 요청한 정보를 처리하는 back-end(Server) 구조로 이루어져 있다.

Client는 javaFX를 이용한 UI와 사용자의 요청을 Server에 보내는 비즈니스 로직으로 설계되었다. 서버로 보내는 요청은 REST API를 사용하여 요청을 보낸다.

Server는 받은 요청을 처리하는 비즈니스 로직과 이를 저장하는 데이터베이스 영역으로 구분되어 있다.

이는 동시에 two-tier Architecture이다.

6. 고려 사항

6.1 서버 구축

- 클라이언트가 자신에 대한 데이터와 종교시설에 대한 데이터를 받기 위해서는 저장소가 필요하다. 이는 파일 시스템으로 해결할 수 있지만, 문제는 독립적인 클라이언트가 데이터를 생성하는 것에 문제가 생긴다. 이를 해결하기 위하여 서버를 구축하고, 해당 서버에서 데이터베이스를 생성하는 것이 목표이다. 이를 해결하기 위한 방법으로 스프링 부트자체에서 서버를 배포할 수 있는 방법이 존재한다.

6.2 UI 개선 문제

- UI는 사용자의 편의를 개선하기 위하여 사용되어져야 한다. 이를 해결하기 위한 방법으로 JavaFX를 이용하여 UI구현을 실시한다.

6.3 REST API를 이용한 HTTP 통신

- 클라이언트와 서버에서 개발해야 할 내용을 명확하게 해야 하며, 서로에 대한 의존성을 감소시킴으로써 낮은 커플링을 유지해야 한다. REST API를 이용하여 이를 해결한다.
- REST API의 특징은 다음과 같다.
    - 유니폼 인터페이스: URI로 지정한 리소스에 대한 조작이 통일되고, 한정적인 인터페이스로 수행한다.
    - 무상태성: 작업을 위한 상태정보를 따로 저장하고 관리하지 않는다. 단순 API서버는 들어오는 요청만을 단순히 처리한다. 이는 서비스의 자유도가 높아지고, 서버에서 불필요한 정보를 관리하지 않음으로써 구현이 단순해진다.
    - 캐시 가능: 캐싱구현이 가능하다.
    - 자체 표현 구조: REST API 메시지만 보고도 쉽게 이해할 수 있는 자체 표현 구조로 되어 있다.
    - 클라이언트-서버 구조
    - 계층형 구조: 보안, 로드밸런싱, 암호화 계층을 추가해 구조상의 유연성을 둘 수 있고, PROXY, 게이트웨이와 같은 네트워크 기반의 중간매체를 사용할 수 있다.

7. 참고문헌

분류	위치
사이트(기독교검색사이트)	http://new.pck.or.kr/address.php
사이트(신천지뉴스)	http://www.hani.co.kr/arti/society/society_general/930633.html
사이트(SpringBoot)	https://gist.github.com/ihoneymon/8a905e1dd8393b6b9298
사이트(Spring boot)	https://www.javatpoint.com/spring-boot-architecture
사이트(javafx)	https://docs.oracle.com/javafx/2/architecture/jfxpub-architecture.htm
사이트(REST)	https://meetup.toast.com/posts/92