# 5. System Architecture – Backend

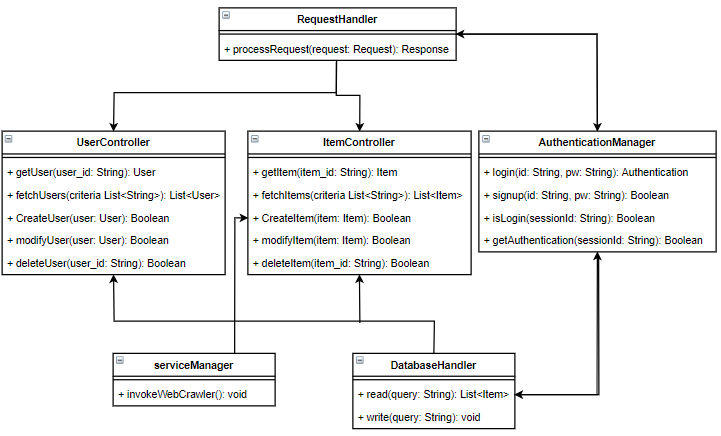
## 5.1. Objectives

System Architecture – Backend에서는 사용자가 직접 사용하는 부분인 frontend를 뺀 Backend의 웹서버에 대한 시스템 및 하위 시스템의 구조를 소개한다.

## 5.2. Overall Architecture

## 5.3. Subcomponents

### A. Application Server



#### 1) RequestHandler: 프론트 엔드로부터의 요청을 처리하는 객체

1-1) processRequest(request: Request): 요청을 각 컨트롤러와 매니저에 알맞게 분배하고 그에 대한 응답을 반환하는 메서드

#### 2) Controllers

2-1) getEntity(entity\_id): 엔티티를 가져오는 메서드

2-2) fetchEntities(criteria: List<String>) 검색 조건에 맞는 엔티티의 리스트를 가져오는 메서드

2-3) createEntity(entity: Entity): 엔티티 생성 메서드

2-4) modifyEntity(entity: Entity): 엔티티 수정 메서드

2-5) deleteEntity(entity\_id): 엔티티 삭제 메서드

#### 3) Service Manager

3-1) invokeWebCrawler(): 웹 크롤러를 실행하는 메서드

#### 4) Authentication Manager

4-1) login(id: String, pw: String): ID/PW로 로그인하고 Authentication을 반환하는 메서드

4-2) signup(id: String, pw: String): 사용자가 사용할 계정의 ID/PW를 DB에 저장하는 메서드

4-3) isLogin(token: String): 사용자가 Authentication을 확보한 상태인지 확인하는 메서드

4-4) getAuthentication(sessionId: String): 로그인 토큰을 바탕으로 현재 접속한 사용자의 정보를 불러오는 메서드

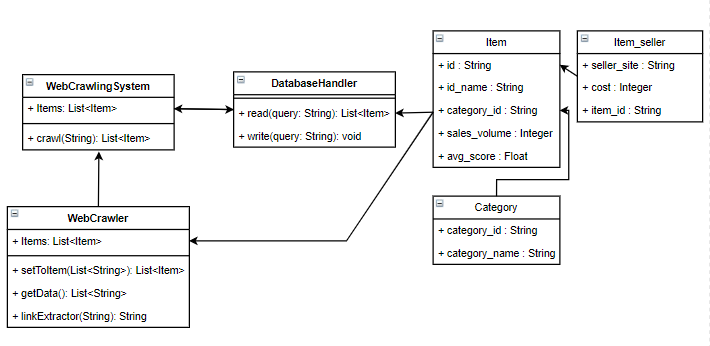
#### 5) Database Handler

5-1) read(query: String): 데이터베이스 조회 쿼리를 데이터베이스에 질의하는 메서드

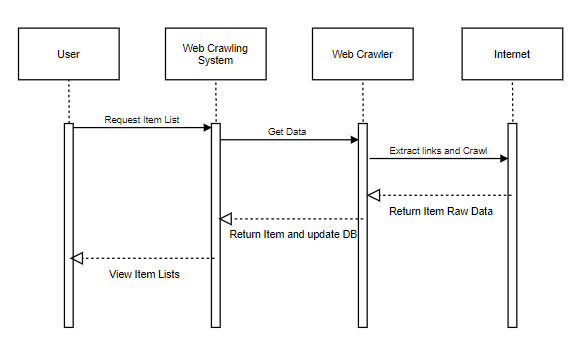
5-2) write(query: String): 데이터베이스 입력 쿼리를 데이터베이스에 질의하는 메서드

### B. Web Crawling system

1) Class diagram

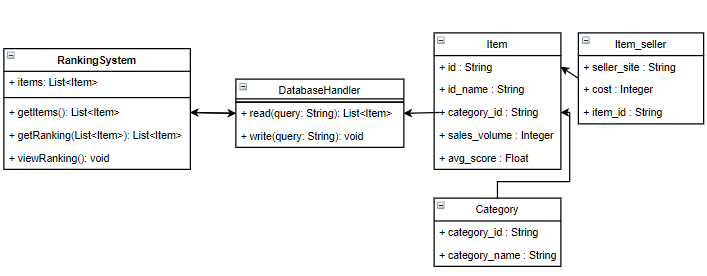


2) Sequence diagram

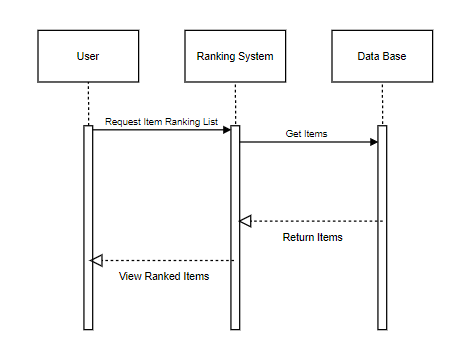


### C. Item Ranking System

1) Class diagram

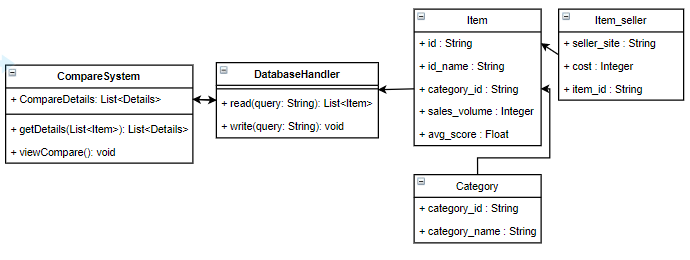


2) Sequence diagram

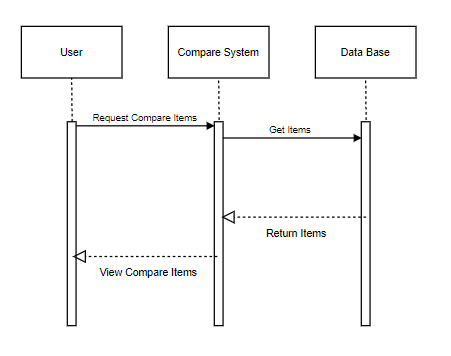


### D. Compare System

1) Class diagram



2) Sequence diagram

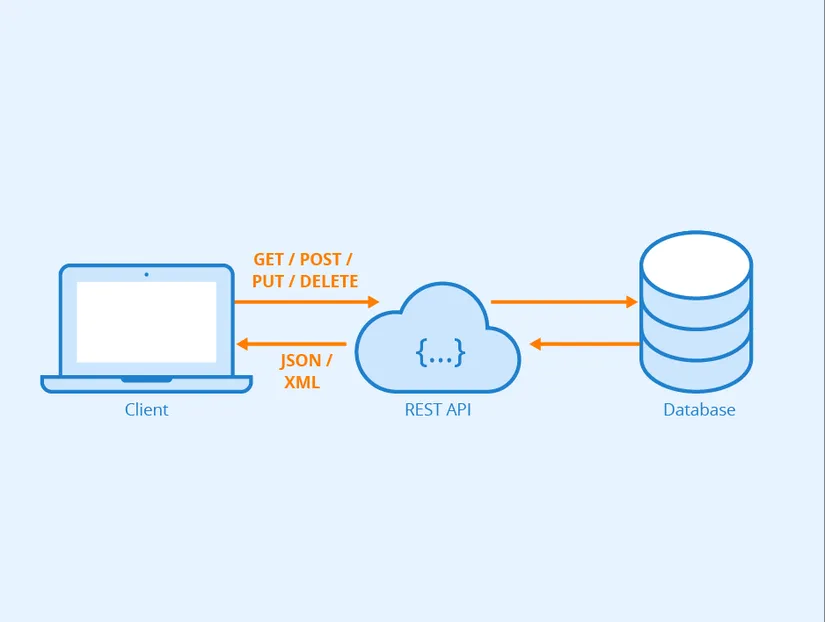


# 6. Protocol Design

## 6.1. Objectives

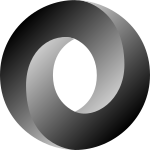
Protocol Design 에서는 프론트엔드와 백엔드에 해당하는 서브시스템들의 통신에 이용되는 프로토콜의 구조에 대한 내용을 설명하고, 각각의 인터페이스를 기술한다.

## 6.2. REST API



본 시스템에서의 프론트 엔드와 벡 엔드의 통신으로 HTTP를 이용하며, 형식은 REST API를 따른다. REST API는 Representational State Transfer” 의 약자로, 자원을 이름(자원의 표현)으로 구분하여 해당 자원의 상태(정보)를 주고 받는 모든 것을 의미한다. REST API는 HTTP URI를 통해 자원을 명시하고, POST, GET, PUT, DELETE와 같은 HTTP 메서드를 통해 해당 자원에 대한 CRUD Operation을 적용한다. REST API를 적용하면, HTTP 프로토콜의 인프라를 그대로 사용하므로 REST API 사용을 위한 별도의 인프라를 구출할 필요가 없으며, 서버와 클라이언트의 역할을 명확하게 분리한다.

## 6.3. JSON



JSON(JavaScript Object Notation)은 속성-값 쌍( attribute–value pairs and array data types (or any other serializable value)) 또는 "키-값 쌍" 으로 이루어진 데이터 오브젝트를 전달하기 위해 인간이 읽을 수 있는 텍스트를 사용하는 개방형 표준 포맷이다. 비동기 브라우저/서버 통신 (AJAX)을 위해, 넓게는 XML(AJAX가 사용)을 대체하는 주요 데이터 포맷이다. 특히, 인터넷에서 자료를 주고 받을 때 그 자료를 표현하는 방법으로 알려져 있다. 자료의 종류에 큰 제한은 없으며, 특히 컴퓨터 프로그램의 변수 값을 표현하는 데 적합하다.