REST API

🚀 RESTful API

📌 RESTful API란?

RESTful API는 API의 다양한 형식들 중 가장 널리 사용되는 아키텍처 스타일입니다. 클라이언트(A)가 어떤 방식으로 요청하고 서버(B)가 어떤 방식으로 응답할지를 명확하게 정의한 규약 중 하나입니다.

✨ 왜 RESTful API가 좋을까?

RESTful API는 **어떤 자원(Resource)에 관한 것인지 명확하게 표현**하고, 가능한 한 그것만을 표기합니다.

📋 기본 구조 예시

```
https://api.example.com

/v1/books           POST    → 책 생성 (Create)
/v1/books           GET     → 모든 책 조회 (Read)
/v1/books/1         GET     → 특정 책 조회 (Read)
/v1/books/20        PUT     → 책 전체 수정 (Update)
/v1/books/7         PATCH   → 책 부분 수정 (Update)
/v1/books/123       DELETE  → 책 삭제 (Delete)
```

🎯 핵심 원칙

- **URI는 자원을 표현**: `v1/books`, `v1/books/1` 등
- **동사는 URI에 포함하지 않음**: 무엇을 할지는 HTTP Method로 표현
- **HTTP Method로 행위 구분**: GET, POST, PUT, PATCH, DELETE

이러한 명확한 구조 덕분에 개발자들은 다른 팀과 협업하거나 새로운 Public API를 사용할 때도 큰 어려움 없이 사용법을 익힐 수 있습니다.

📊 데이터 형식

RESTful API 요청과 응답에는 **구조화된 표현이 가능하면서 가벼운 JSON**을 주로 사용합니다.

🔍 필터링과 페이징

Query Parameter 활용

```
/v1/books?status=available&size=10
```

- `status=available` : 상태별 필터링
- `size=10` : 한 페이지에 10개씩 표시

고유 식별자 사용

```
/v1/books/1
```

- `1` : 책의 고유 ID(식별자)

🔗 HATEOAS 원칙

HATEOAS(Hypermedia As The Engine of Application State)는 RESTful API에서 권장되는 또 다른 중요한 원칙입니다.

💡 개념

각 요청의 응답에 **사용 가능한 다른 요청들의 정보를 포함**시킵니다.

📝 응답 예시

```
{
  "id": 1,
  "title": "Clean Code",
  "author": "Robert C. Martin",

  "links": [
    {
      "rel": "self",
      "href": "https://api.example.com/v1/books/1"
    },
    {
      "rel": "update",
      "href": "https://api.example.com/v1/books/1"
    },
    {
      "rel": "reviews",
      "href": "https://api.example.com/v1/books/1/reviews"
    }
  ]
}
```

이 `links` 정보를 통해 API 문서를 모두 찾아보지 않아도 다음에 어떤 요청을 보낼 수 있는지 쉽게 파악할 수 있습니다.

🧬 리소스 관계 표현

리소스들 간의 관계는 **중첩된 URI 구조**로 표현합니다.

📚 예시: 책과 리뷰의 관계

```
/v1/books/1/reviews
```

- 색인번호가 1인 책에 대한 모든 리뷰를 의미

```
┌──────────────┐
│     책       │
├──────────────┤
│  기타 정보   │
│     ...      │
├──────────────┤
│    리뷰      │ ← /v1/books/1/reviews
└──────────────┘
```

🧮 HTTP 상태 코드

RESTful API는 명확한 상태 코드를 사용하여 요청 결과를 알려줍니다.

상태 코드	의미	예시
2XX	성공	200 OK, 201 Created
4XX	클라이언트 오류	400 Bad Request, 404 Not Found
5XX	서버 오류	500 Internal Server Error

🔄 RESTful API의 핵심 특징

🚫 무상태성 (Stateless)

서버는 클라이언트에 대해 아무것도 기억하지 않아야 합니다. 각 요청은 독립적이어야 합니다.

🔁 멱등성 (Idempotent)

클라이언트가 같은 요청을 몇 번 보내도 동일한 결과를 얻어야 합니다.

HTTP Method	멱등성
GET	✅
PUT	✅
DELETE	✅
POST	❌
PATCH	❌

⚡ 캐싱 가능

특정 요청에 대한 응답을 캐시에 저장해두면, 데이터베이스 연결 없이도 빠르게 응답할 수 있습니다.

🎯 마무리

RESTful API는 명확하고 일관된 규칙을 통해 **직관적이고 사용하기 쉬운 API**를 만들 수 있게 해줍니다. 이러한 특징들 덕분에 현재 웹 개발에서 가장 널리 사용되는 API 아키텍처가 되었습니다.