

FastAPI



소프트웨어융합대학원
진혜진

■ API와 REST

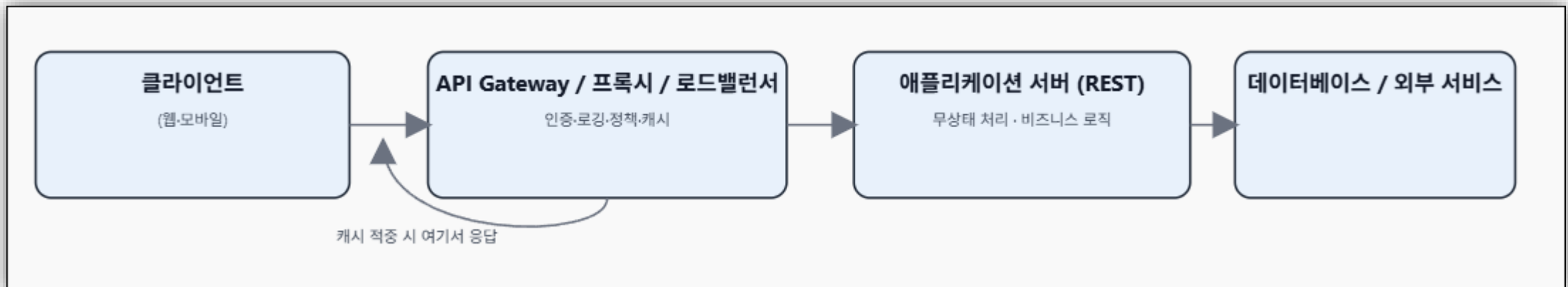
- API (Application Programming Interface)는 시스템 간 상호 작용을 위한 규칙
- REST는 자원 기반 설계, 무상태성, 표준 HTTP를 활용하는 아키텍처 스타일
- RESTful API는 일관된 URI와 HTTP 메서드를 사용해 자원을 조작

```
{  
  "city": "Seoul",  
  "tempC": 23.4,  
  "conditions": ["cloudy", "wind"]  
}
```

메서드	설명	역등성
GET	자원 조회	✓
POST	새 자원 생성	X
PUT	전체 자원 업데이트	✓
PATCH	부분 업데이트	X
DELETE	자원 삭제	✓

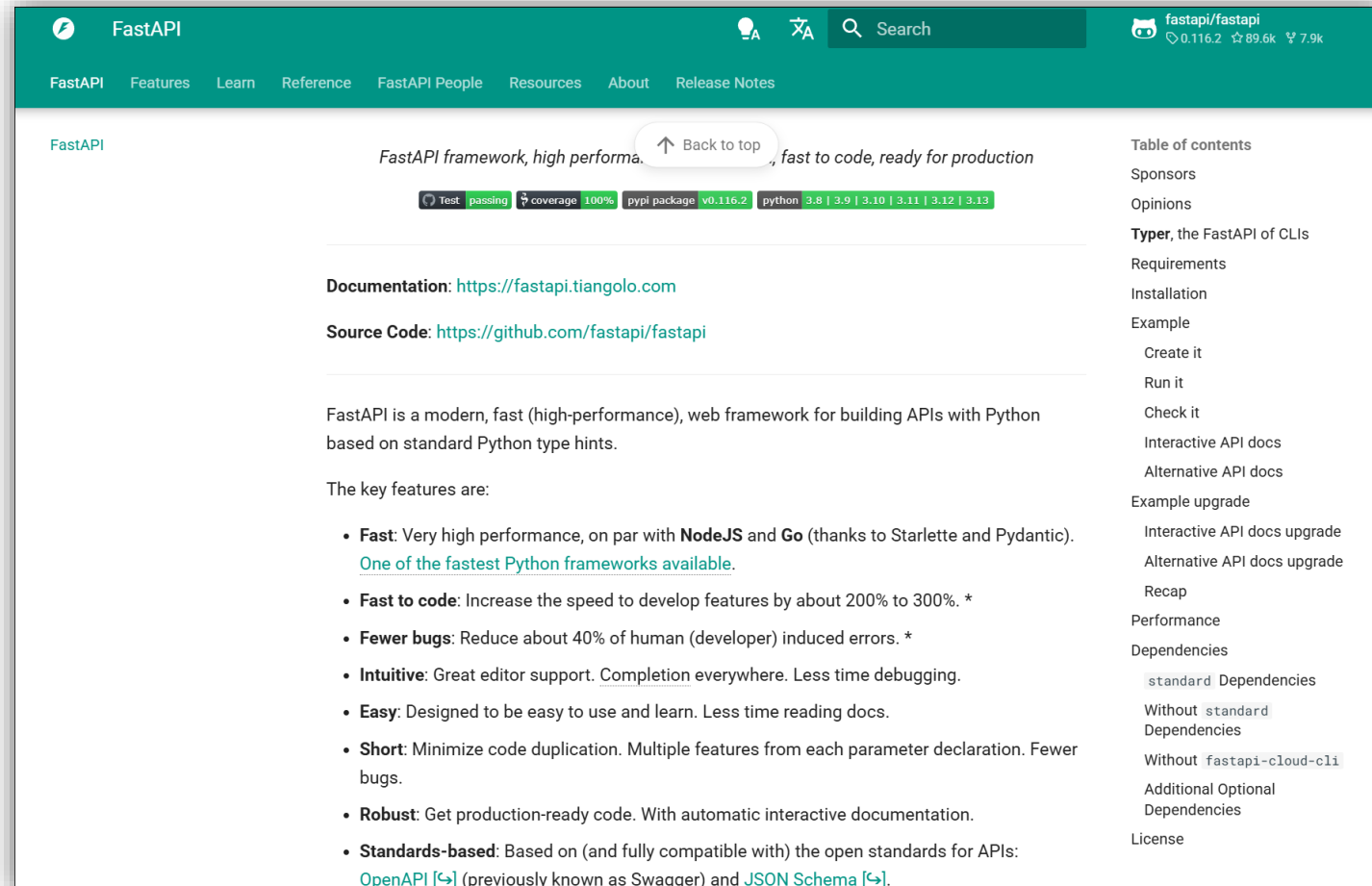
■ REST 설계 원칙

- 리소스는 명사형 URI로 표현 (/users, /products).
- 클라이언트-서버 분리 및 무상태성(State-less).
- JSON, XML 등 표현 계층을 통한 전송.
- 일관된 인터페이스: HTTP 메서드와 상태 코드 활용.



■ FastAPI

- 표준 Python 유형 힌트를 기반으로 Python으로 API를 구축하기 위한 현대적이고 빠른(고성능) 웹 프레임워크



Opinions

*"[...] I'm using **FastAPI** a ton these days. [...] I'm actually planning to use it for all of my team's **ML services at Microsoft**. Some of them are getting integrated into the core **Windows** product and some **Office** products."*

Kabir Khan - **Microsoft** [\(ref\)](#)

*"We adopted the **FastAPI** library to spawn a **REST** server that can be queried to obtain **predictions**. [for Ludwig]"*

Piero Molino, Yaroslav Dudin, and Sai Sumanth Miryala - **Uber** [\(ref\)](#)

*"**Netflix** is pleased to announce the open-source release of our **crisis management** orchestration framework: **Dispatch!** [built with **FastAPI**]"*

Kevin Glisson, Marc Vilanova, Forest Monsen - **Netflix** [\(ref\)](#)

*"I'm over the moon excited about **FastAPI**. It's so fun!"*

Brian Okken - **Python Bytes** podcast host [\(ref\)](#)

*"Honestly, what you've built looks super solid and polished. In many ways, it's what I wanted **Hug** to be - it's really inspiring to see someone build that."*

Timothy Crosley - **Hug** creator [\(ref\)](#)

FastAPI

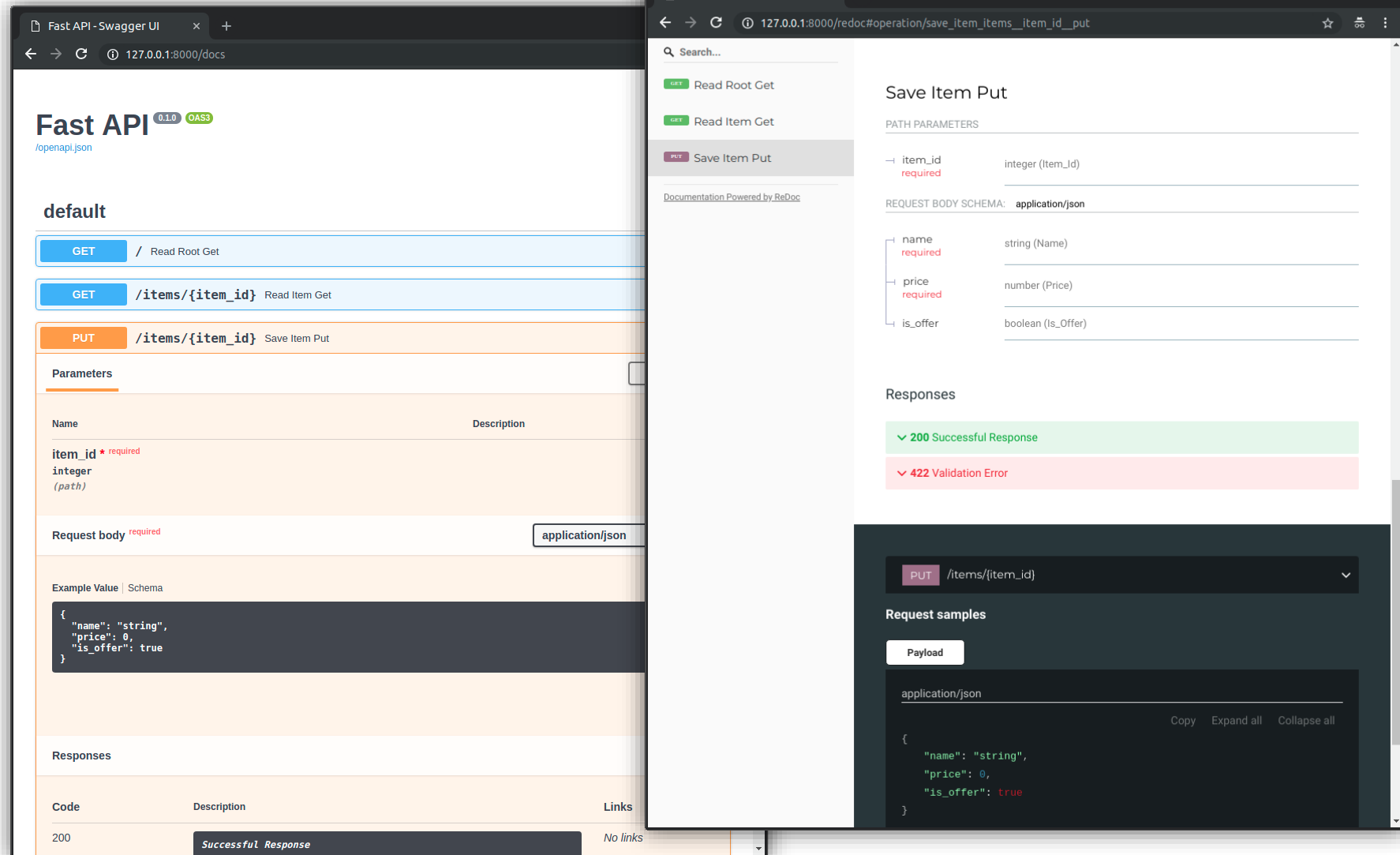
■ 개방형 표준 기반

■ 자동 문서

- 스웨거 UI
- 리닥

■ 스타렛

■ Pydantic



The image displays two overlapping browser windows showing API documentation for a FastAPI application.

The background window is the Swagger UI, titled "Fast API - Swagger UI", showing the "default" API version. It lists three endpoints:

- GET / Read Root Get
- GET /items/{item_id} Read Item Get
- PUT /items/{item_id} Save Item Put

The "Save Item Put" endpoint is selected, showing its details:

- Parameters:** item_id (integer, required, path)
- Request body:** application/json (required)
- Example Value:**

```
{
  "name": "string",
  "price": 0,
  "is_offer": true
}
```
- Responses:** 200 Successful Response

The foreground window is the ReDoc interface, titled "Fast API - ReDoc", showing the same "Save Item Put" endpoint with more detailed information:

- PATH PARAMETERS:** item_id (integer (Item_Id), required)
- REQUEST BODY SCHEMA:** application/json
 - name (string (Name), required)
 - price (number (Price), required)
 - is_offer (boolean (Is Offer))
- Responses:** 200 Successful Response, 422 Validation Error
- Request samples:** Payload:

```
application/json
{
  "name": "string",
  "price": 0,
  "is_offer": true
}
```

■ FastAPI

- Python 기반의 현대적 웹 프레임워크.
- Pydantic으로 입력/출력 데이터 유효성 검증 및 직렬화 지원.
- OpenAPI/Swagger 문서를 자동 생성하여 인터랙티브 문서 제공.
- Starlette 기반의 비동기 처리로 높은 성능을 보장.

■ FastAPI 장점

- Starlette와 Uvicorn 덕분에 속도가 빠르고 효율적
- 타입 힌트를 기반으로 IDE 자동 완성과 정적 분석이 용이
- Swagger/ReDoc 문서가 자동 생성되어 API 이해가 빠름
- 의존성 주입(Dependency Injection)으로 테스트와 모듈화가 쉬움

■ FastAPI의 제한점

- 비동기 프로그래밍에 익숙하지 않은 경우 학습 곡선이 존재
- 복잡한 프로젝트에서는 명확한 구조 설계가 필요
- 일부 고급 기능은 추가 라이브러리 학습을 요구할 수 있다.

■ 최신 동향 & FastAPI CLI

- Pydantic v2: PEP 695/696 제네릭 지원, 별칭 설정 API 개선, 성능 향상 및 실험적인 Free-Threading 지원
- FastAPI CLI: `fastapi dev` 로 개발 서버를 실행하고, `fastapi run` 으로 프로덕션 빌드 가능
- CLI 설치: `pip install "fastapi[standard]"` 시 `fastapi-cli` 패키지가 포함
- CLI는 애플리케이션 인스턴스를 자동으로 감지하고 Uvicorn을 사용