VuewithDRF

작성 일시	@2022년 11월 14일 오전 10:15
강의 날짜	@2022/11/14
편집 일시	@2022년 11월 14일 오후 5:30
분야	Vue
공부 유형	강의
복습	☐
태그	CORS DRF Auth System DRF Auth with Vue DRF-spectacular Vue with DRF

Vue with DRF

개요

- Server와 Client의 통신 방법 이해하기
- CORS 이슈 이해하고 해결하기
- DRF Auth System 이해하기
- Vue와 API server 통신하기

Server & Client

Server

- 서버란?
  - 클라이언트에게 **정보**와 **서비스**를 제공하는 컴퓨터 시스템
  - 서비스 전체를 제공 == Django Web Service
  - 정보를 제공 == DRF API Service

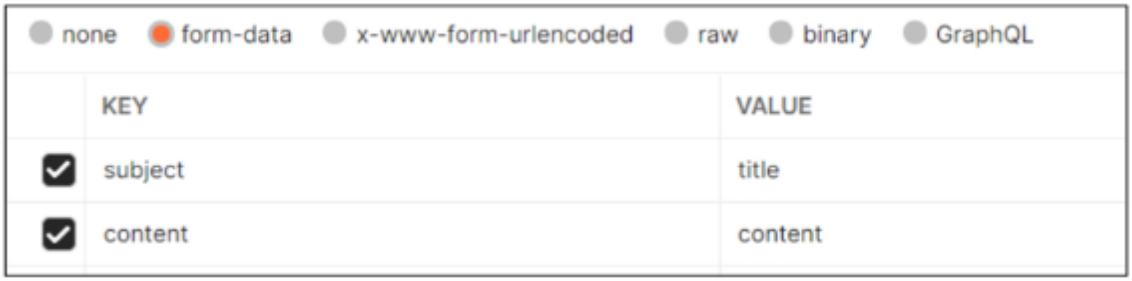
- 서비스 전체를 제공 == Django Web Service
  - Django를 통해 전달받은 HTML에는 하나의 웹 페이지를 구성할 수 있는 모든 데이터가 포함
  - 즉, 서버에서 모든 내용을 렌더링 → 하나의 HTML 파일로 제공
  - 정보를 포함한 web 서비스를 구성하는 모든 내용을 서버 측에서 제공
- 정보를 제공 == DRF API Service
  - Django를 통해 관리하는 정보만을 클라이언틍게 제공
  - DRF를 사용하여 JSON으로 변환

Client

- 클라이언트란?
  - **Server가 제공하는 서비스에 적절한 요청**을 통해 **Server로부터 반환 받은 응답을 사용자에게 표현**하는 기능을 가진 프로그램 혹은 시스템
- Server가 제공하는 서비스에 적절한 요청
  - Server가 정의한 방식대로 요청 인자를 넘겨 요청
  - Server는 정상적인 요청에 적합한 응답 제공
- 잘못된 요청 예
  - 아래와 같은 Model이 정의되어 있다면

```python
class Article(models.Model):
    title = models.CharField(max_length=100)
    content = models.TextField()
```

  - 잘못된 field 명으로 요청을 보낼 경우 처리할 수 없음



- Server로부터 반환 받은 응답을 사용자에게 표현
    - 사용자의 요청에 적합합 data를 server에 요청하여 응답받은 결과로 **적절한 화면을 구성**

정리

- Server는 정보와 서비스를 제공
    - DB와 통신하며 데이터를 생성, 조회, 수정, 삭제를 담당
    - 요청을 보낸 Client에게 정상적인 요청이었다면 처리한 결과를 으답
- Client는 사용자의 정보 요청을 처리, server에게 응답 받은 정보를 표현
    - Server에게 정보(데이터)를 요청
    - 응답 받은 정보를 가공하여 화면에 표현

CORS

What Happened?

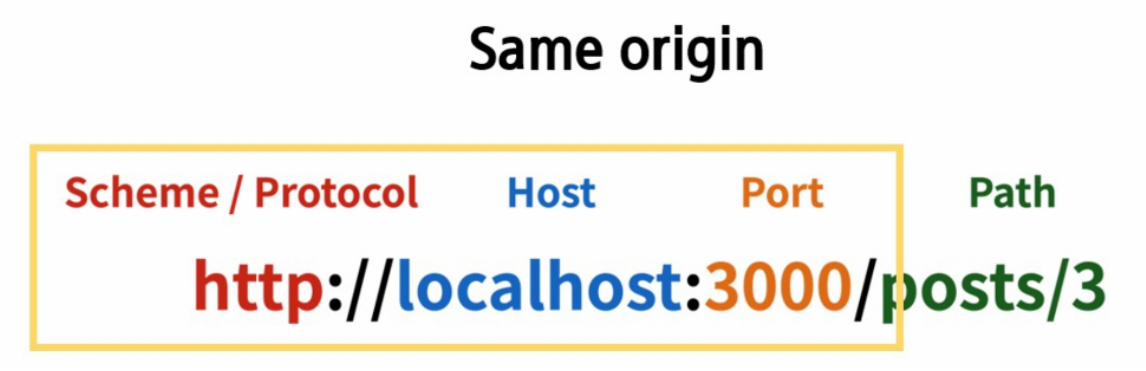
- 브라우저가 요청을 보내고 서버의 응답이 브라우저에 도착
    - Server의 log는 200(정상) 반환
    - 즉 Server는 정상저긍로 응답했지만 브라우저가 막은 것
- 보안상의 이유로 브라우저는 **동일 출처 정책(SOP)**에 의해 다른 출처의 리소스와 상호작용 하는 것을 제한함

SOP (same - Origin Policy)

- 동일 출처 정책
- 불러온 문서나 스크립트가 다른 출처에서 가져온 리소스와 상호작용하는 것을 제한하는 보안 방식
- 잠재적으로 해로울 수 있는 문서를 분리함으로써 공격받을 수 있는 경로를 줄임

Origin - 출처

- **URL의 Protocol, Host, Port를 모두 포함**하여 출처라고 부름
- Same Origin 예시
    - 아래 세 영역이 일치하는 경우에만 동일 출처로 인정



URL	결과	이유
http://localhost:3000/posts/	성공	path만 다름
http://localhost:3000/comments/3/	성공	path만 다름
https://localhost:3000/posts/3/	실패	protocol이 다름
http://localhost:80/posts/3/	실패	port가 다름
https://domain:3000/posts/3/	실패	Host가 다름

CORS - 교차 출처 리소스 공유

- 추가 **HTTP Header**를 사용하여, 특정 출처에서 실행중인 웹 어플리케이션이 **다른 출처의 자원에 접근할 수 있는 권한**을 부여하도록 브라우저에 알려주는 체제
  - 어떤 출처에서 자신의 컨텐츠를 불러갈 수 있는지 **서버에 지정**할 수 있는 방법
- 리소스가 자신의 출처와 다를 때 교차 출처 HTTP 요청을 실행
  - 만약 다른 출처의 리소스를 가져오기 위해서는 이를 제공하는 서버가 브라우저에게 **다른 출처지만 접근해도 된다는 사실을 알려야 함**
  - **교차 출처 리소스 공유 정책 (CORS policy)**

CORS policy - 교차 출처 리소스 공유 정책

- 다른 출처에서 온 리소스를 공유하는 것에 대한 정책
- CORS policy에 위배되는 경우 브라우저에서 해당 응답 결과를 사용하지 않음
  - Server에서 응답을 주더라도 브라우저에서 거절
- 다른 출처의 리소스를 불러오려면 그 출처에서 **올바른 CORS header**를 포함한 응답을 반환해야함

How to set CORS

- CORS 표준에 의해 추가된 HTTP Response Header를 통해 이를 통제 가능
- HTTP Response Header 예시
  - `Access-Control-Allow-Origin`
- Access-Control-Allow-Origin
  - 단일 출처를 지정하여 브라우저가 해당 출처가 리소스에 접근하도록 허용

DRF Auth System

Authentication & Authorization

Authentication - 인증, 입증

- 자신이라고 주장하는 사용자가 누구인지 확인하는 행위
- 모든 보안 프로세스의 첫 번째 단계 (가장 기본 요소)
- 즉, 내가 누구인지를 확인하는 과정
- 401 Unauthorized
  - 비록 HTTP 표주에서는 미승인(unauthorized)을 명확히 하고 있지만, 의미상 이 응답은 비인증(unauthenticated)을 의미

Authorization - 권한 부여, 허가

- 사용자에게 특정 리소스 또는 기능에 대한 액세스 권한을 부여하는 과정 (절차)
- 보안 환경에서 권한 부여는 항상 인증이 먼저 필요함
  - 사용자는 조직에 대한 액세스 권한을 부여받기 전에 먼저 자신의 ID가 진짜인지 먼저 확인해야함
- 서류의 등급, 웹페이지에서 글을 조회 & 삭제 & 수정할 수 있는 방법, 제한 구역
  - 인증이 되었어도 모든 권한을 부여받는 것은 아님
- 403 Forbidden
  - 401과 다른 점은 서버는 클라이언트가 누구인지 알고 있음

Authentication and authorization work together

- 회원가입 후, 로그인 시 서비스를 이용할 수 있는 권한 생성
  - 인증 이후에 권한이 따라오는 경우가 많음

- 단, 모든 인증을 거쳐도 권한이 동일하게 부여되는 것은 아님
    - Django에서 로그인을 했더라도 다른 사람의 글까지 수정 / 삭제가 가능하진 않음
- 세션, 토큰, 제 3자를 활용하는 등의 다양한 인증 방식이 존재