백엔드 직무부트캠프 2차 과제

오규혁

1. HTTP 통신이란?

- HTTP 통신이란 HyperText Transfer Protocol의 약자로 HTML 파일을 전송하는 프로토콜이라는 의미를 가진다. 웹브라우저에서 통신이 일어나며, 초기에는 HTML 파일을 전송하려는 목적으로 만들어졌으나 현재는 JSON, Image 파일 등 또한 전송한다.

HTTP 통신은 클라이언트에서 서버로 요청ㅇ을 보내고 서버가 응답하는 방식으로 통신이 이루어진다. 응답은 클라이언트의 요청에 따른 결과를 반환한다. 이로 인해 클라이언트의 요청이 있을 때 서버가 응답하는 방식이 단방향 방식으로 이루어진다.

서버의 응답에는 응답 코드가 같이 전송되며, 사용자는 응답 코드와 메시지 응답으로부터 오는 메시지 바디를 통해 요청 값을 전달받는다.

초기에는 서버는 응답한 후 클라이언트의 Connection을 곧바로 끊어버렸으나, 최근에는 성능상의 이유로 Keep Alive 옵션을 통해 일정 기간 동안 클라이언트와 Connection을 유지하는 방식으로 통신이 가능해졌다.

추가적으로 HTTP가 단방향 통신이라면, 양방향 통신이 가능한 통신 방식이 있는데 이를 소켓 통신이라 일컫는다. 즉 소켓 통신이란 서버와 클라이언트의 양방향 연결이 이루어지는 통신으로, 클라이언트도 서버로 요청을 보낼 수 있고 서버도 클라이언트로 요청을 보낼 수 있는 통신으로 양방향 통신의 방식을 따른다.

보통 스트리밍이나 실시간 채팅 등 실시간으로 데이터를 주고 받아야 하는 경우 Connection을 자주 맺고 끊는 HTTP 통신보다 소켓 통신이 적합하다. 소켓 통신은 계속해서 Connection을 들고 있기 때문에 HTPP 통신에 비해 많은 리소스가 소모된다.

2. 웹 브라우저에 URL을 입력하고 서버에서 응답하는 과정까지

- 웹 브라우저에서 <https://www.naver.com>과 같은 URL을 입력하면 브라우저는 인터넷에서 사이트를 호스팅하는 서버를 파악하는데, [www.naver.com](http://www.naver.com)과 같은 도메인을 검색해서 주소를 찾게 된다. 이 도메인 또한 IP 주소를 가지고 있으며, 사람이 IP 주소를 기억하기에는 어려움이 있어 기억하고 알아보기 쉽게 도메인으로 표현해주는 것이다. 이 도메인을 DNS가 인터넷에서 서버를 찾는데 도움을 준다. DNS 조회를 수행하여 도메인 이름을 기반으로 서버의 IP주소를 찾을 수 있다.

이제 좀 더 자세히 살펴보자. URL의 https는 통신 프로토콜이다. 이 스키마는 브라우저에 전송 계층 보안을 사용하여 서버에 연결하도록 지시한다. https를 사용하면 암호와 같이 중요한 정보들이 브라우저와 서버가 데이터를 교환할 때 암호화된다.

브라우저는 URL을 가지고 인터넷에서 연결할 서버를 파악하기 위해 입력한 도메인을 사용하여 웹 사이트를 호스팅하는 서버의 IP주소를 알아낸다. DNS가 이 작업을 수행한다.

DNS는 복잡하고 매우 빠르게 동작해야 하기 때문에 캐시에 저장되게 된다. 웹 브라우저는 고유한 캐시 -> 운영 체제 캐시 -> 라우터의 로컬 네트워크 캐시 -> 회사 네트워크 또는 인터넷 서비스 제공업체의 DNS 캐시를 확인한다.

만약 웹 브라우저가 캐시 계층에서 IP주소를 찾을 수 없는 경우 회사 네트워크 또는 ISP의 DNS 서버가 재귀적 DNS 조회를 수행한다.

재귀적으로 인터넷에 있는 여러 DNS 서버를 요청하며, 검색될 때까지 많은 DNS 서버에 요청한다. 웹 브라우저가 IP 주소로 DNS 레코드를 가져오면 인터넷에서 서버를 찾아 연결을 설정한다.

인터넷에 연결된 웹 브라우저 요청 패킷은 일반적으로 TCP/IP 전송 제어 프로토콜을 사용하여 라우터 장비, 인터넷 서비스 제공회사 교환기를 통해 이동되어 통신 회사간 경로인 라우팅 테이블을 따라서 연결할 IP 주소가 있는 웹 서버를 찾는다.

웹 브라우저가 인터넷에서 서버를 찾으면 웹 서버와 TCP 연결을 설정하고, HTTP를 통해 통신을 시작한다. 하지만 요즘은 대부분 HTTPS를 사용하고, HTTPS는 주고받는 데이터의 암호화를 위한 TLS 핸드셰이크 과정을 수행한다. 이는 암호화를 할 대상을 확인하는 것이다.

웹 브라우저가 서버에 연결되면(TCP), HTTPS 프로토콜에 대한 통신 규칙을 따른다. 웹 브라우저가 페이지의 콘텐츠를 요청하기 위해 HTTP 요청을 전송하는데, HTTP 요청에는 요청 라인, 헤더(또는 요청에 대한 메타데이터) 및 본문이 포함된다 요청 라인에는 클라이언트(브라우저)가 수행하려는 작업을 서버가 확인하는 데 사용하는 정보가 포함되어 있다.

<요청 라인>  
GET /tables/user HTTP/1.1  
- GET, POST, PUT, DELETE 등 요청 메소드 : GET  
- 요청된 리소스를 가리키는 경로 : /tables/user  
- 통신할 HTTP 버전 : HTTP/1.1

요청 헤더(Request Header)는 요청을 라우팅하는데 도움이 되는 추가 정보를 전달하고, 어떤 유형의 클라이언트(사용자 에이전트)가 요청을 수행했는지 나타내며, 블루/그린 배포나 카나리 배포를 지원하는데 사용할 수 있다.

웹 서버는 요청을 받고 요청 라인, 헤더 및 본문의 정보를 기반으로 요청을 처리한다. 콘텐츠를 가져오고 응답을 생성하여 클라이언트로 다시 전송을 하며 응답에는 다음이 포함된다.

- 클라이언트에게 요청 상태를 알려주는 상태 라인 (200, 404 등)

- 브라우저에 응답 처리 방법을 알려주는 응답 헤더 (text/html, application/json 등)

- 해당 경로에서 요청된 리소스 (이미지 파일과 같은 콘텐츠 또는 데이터, HTML, CSS, JS)

마지막으로 웹 브라우저가 응답을 받으면 콘텐츠를 렌더링하는 작업을 수행한다.

3. API 명세서 작성

활용률(접속자 수, 부서별 접속자 수, 로그인 요청 수, 게시 글 작성 수)에 필요한 데이터가 무엇일 지 고민해보고 해당 내 용에 맞게 문서를 작성(요청 파라미터 생각, 응답 데이터 포맷 고려) 합니다.

1. 접속자 수 API

* GET /rest/{login}/{yearMonth}
* Requst Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Mandatory | Note |
| RequestName | String |  | O | Login |
| yearMonth | String |  | O | 20230901 |

* Response Body

|  |
| --- |
| {  “totcnt”: 10,  “yearMonth”: “20230901”,  “requestslog”: “L”,  “is\_success”: true  } |

1. 부서별 접속자 수 API

* GET /rest/{login}/{yearMonth}/{department}
* Requst Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Mandatory | Note |
| RequestName | String |  | O | login |
| yearMonth | String |  | O | 20230901 |
| Department | String |  | O | A |

* Response Body

|  |
| --- |
| {  “totcnt”: 5,  “yearMonth”: “20230901”,  “Department”: “A”,  “requestslog”: “L”,  “is\_success”: true  } |

1. 로그인 요청 수 API

* GET /rest/{login}/{yearMonth}
* Requst Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Mandatory | Note |
| RequestName | String |  | O | login |
| yearMonth | String |  | O | 20230901 |

* Response Body

|  |
| --- |
| {  “totcnt”: 10,  “yearMonth”: “20230901”,  “requestslog”: “L”,  “is\_success”: true  } |

1. 유저별 로그인 요청 수 API

* GET /rest/{login}/{yearMonth}/{userID}
* Requst Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Mandatory | Note |
| RequestName | String |  | O | login |
| yearMonth | String |  | O | 20230901 |
| userID | String |  | O | 12345 |

* Response Body

|  |
| --- |
| {  “totcnt”: 5,  “yearMonth”: “20230901”,  “userID”: “12345”,  “requestslog”: “L”,  “is\_success”: true  } |

1. 게시글 작성 수 API

* GET /rest/{login}/{yearMonth}
* Requst Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Mandatory | Note |
| RequestName | String |  | O | login |
| yearMonth | String |  | O | 20230901 |
| boardID | String |  | O | community |

* Response Body

|  |
| --- |
| {  “totcnt”: 10,  “yearMonth”: “20230901”,  “boatdID”: “community”,  “requestslog”: “L”,  “is\_success”: true  } |

1. 요청 성공시(HTTP response 200)

|  |
| --- |
| {  "status": "success",  "message": "요청이 성공적으로 처리되었습니다.",  "data": {  "total\_users": 100,  "active\_users": 50  }  } |

1. 요청이 잘못됨(HTTP 400)

|  |
| --- |
| {  "status": "error",  "message": "잘못된 요청입니다. 필수 요청 파라미터가 누락되었습니다."  } |

1. 권한이 없는 경우(HTTP 403)

|  |
| --- |
| {  "status": "error",  "message": "접근 권한이 없습니다. 해당 리소스에 접근할 권한이 없습니다."  } |

1. 백엔드 서버 오류(HTTP 500)

|  |
| --- |
| {  "status": "error",  "message": "내부 서버 오류로 인해 요청을 처리할 수 없습니다."  } |