

# **뚝딱! 일주일만에 웹 서비스 구현하기**

**아주대학교  
사이버보안학과  
이진근**

# 목차

- 필요한 것들
- 웹서버
- 프론트엔드와 백엔드
- 도메인 라우팅
- 만들었던 서비스
- Q&A

# 필요한 것들

- 아이디어
- 깃
- 도메인
- 배포할 서버 등

# 웹서버?



**APACHE**

HTTP SERVER PROJECT



**NGINX**



# APACHE

HTTP SERVER PROJECT

- **프로세스 기반**

- ✓ 클라이언트 요청 1개 당 스레드가 생성
- ✓ 사용자가 많아지면 그만큼 많은 스레드가 만들어짐
- ✓ 따라서 다중 요청에서는 부하가 심해짐

- **멀티 프로세싱**

- ✓ 실행 중인 프로세스를 복제
- ✓ 응답 프로세스를 생성하고 클라이언트 요청 시 자식 프로세스가 반응하게 되는 방식

- **단점**

- Apache 서버의 프로세스가 블록킹이 되면 요청을 처리하지 못하고, 처리가 완료될 때까지 계속 대기



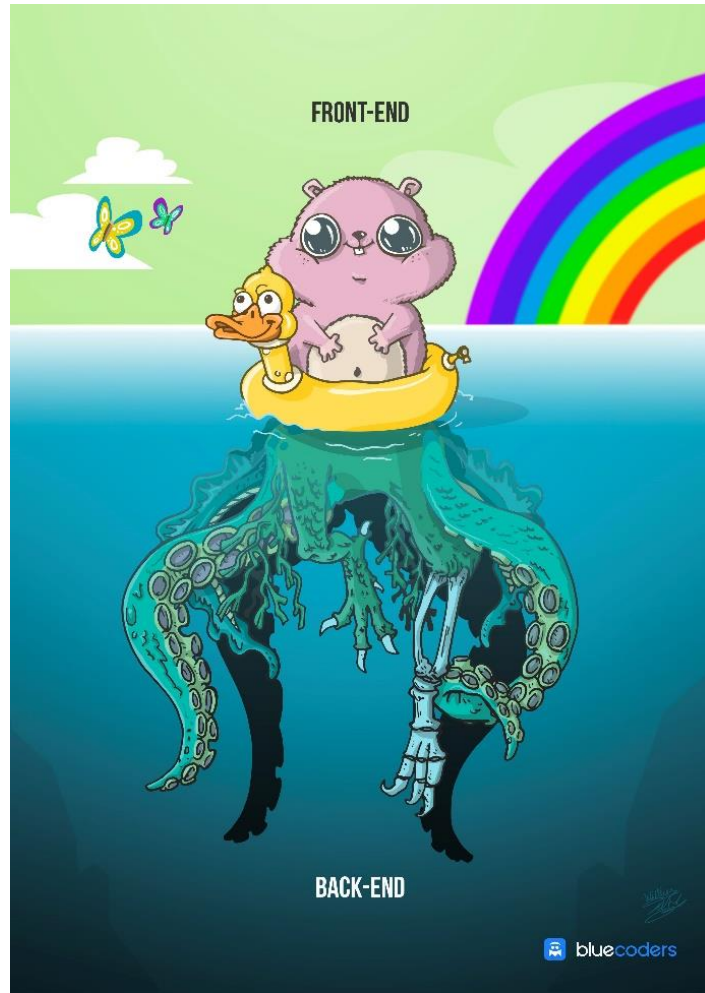
- **Event-Driven 처리 기반**

- 여러 개의 커넥션을 Event-Handler를 통해 비동기로 처리해 먼저 처리되는 것부터 로직이 진행
- 스레드를 많이 사용하지 않기 때문에 전체적인 코스트가 적다
- 적은 수의 스레드로 효율적으로 일 처리하며, 스레드에 메모리가 적게 할당되는 구조
- CPU와 관계없이 모든 IO들을 전부 Event Listener로 미루기 때문에 응답이 빠르게 진행이 되어 단일 프로세스로 빠른 작업이 가능

# Nginx와 apache

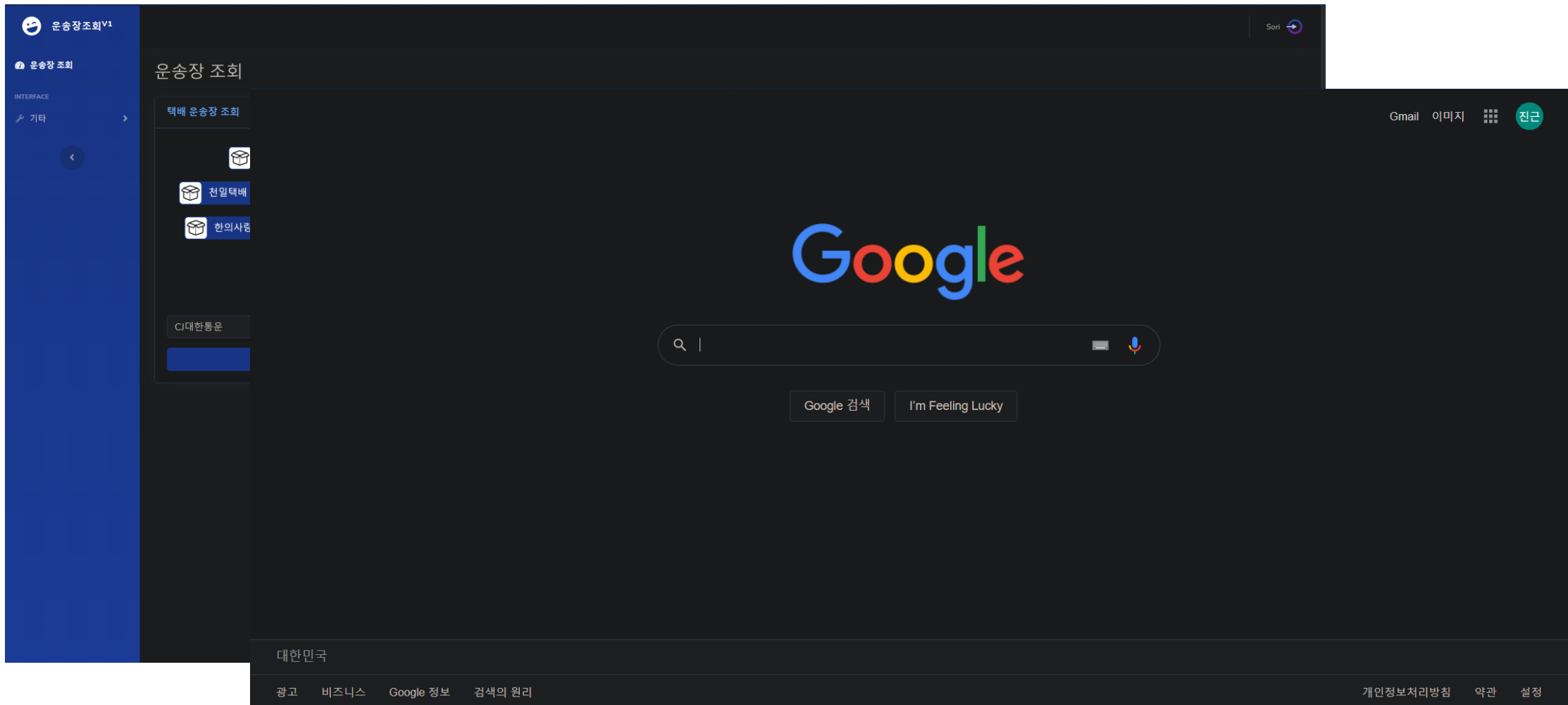
- 장/단점이 서로 있기에 어디서 무얼 쓰는 건 백엔드 개발자의 취향
- 단일 서비스나 SI 쪽에서는 apache를 많이 선호
- 스타트업이나 트래픽이 많거나, 다중 서비스를 운영한다면 nginx 선호
- 본인은 개인적으로 nginx를 좋아함.
  - SSL, TLS 적용이 편리함.
  - Web-app과 api 서버를 라우팅할 때 편하기 때문
- 따라서 여기서는 nginx를 씁니다.

# 프론트 엔드와 백엔드





# Front-end



# Front-end

**HTML**



*Markup Language*  
**Content**

**CSS**



*Style sheet Language*  
**Presentation**

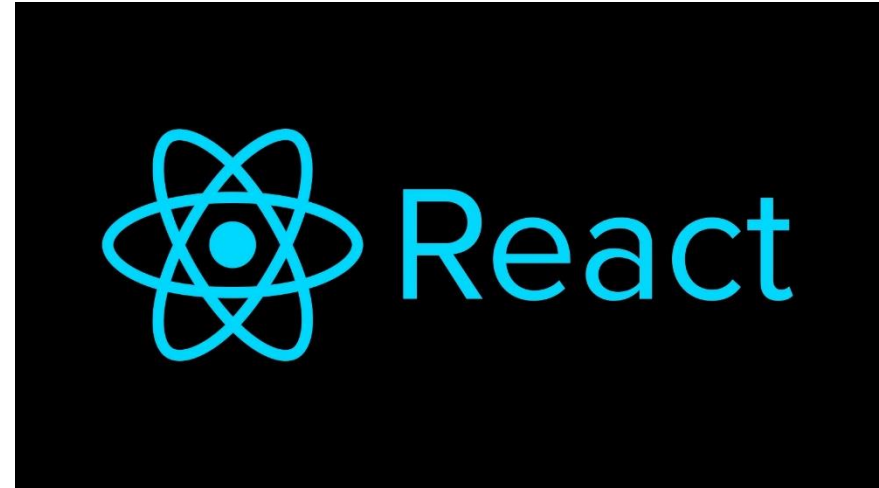
**JS**



*Programming Language*  
**Behavior**



Vue.js



# Back-end

```
mysql> mysql> select * from savePostInfo where user_idx=1;
+-----+-----+-----+-----+-----+
| idx | user_idx | postNum | postItem | carriers |
+-----+-----+-----+-----+-----+
| 8 | 1 | 356503801791 | 67iU660o7Yis7IqkI0ydt0yWt02PsA== | kr.cjlogistics |
| 10 | 1 | 94157469604 | 7Jqw7JiB7J206rCAI0uzt0uCu0qxsA== | kr.logen |
| 11 | 1 | 95404102463 | 7ZiE7KSA7J20I0y7tA== | kr.logen |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

```
0|toy | GET /delivery/api/users/ 200 6ms - 53b
0|toy | GET /delivery/api/users/postNum 200 13ms - 401b
0|toy | GET /delivery/img/logo/box.jpg 304 1ms
0|toy | GET /delivery/api/carriers/kr.cjlogistics/tracks/356503801791 200 171ms - 1.52kb
```

POST ▾

sori.no.e.systems/clean/api/users/login

Params

Send ▾

Body

Cookies

Headers (6)


Test Results

Pretty

Raw

Preview

JSON ▾



1 ▾

2 {

3 ▾

4 "status": "ok",

5 "data": {

6 "student\_id": 201920663,

7 "name": "이진근",

8 "dept": "정보통신대학",

9 "school": "사이버보안학과"

10 }

11 }

# Back-end



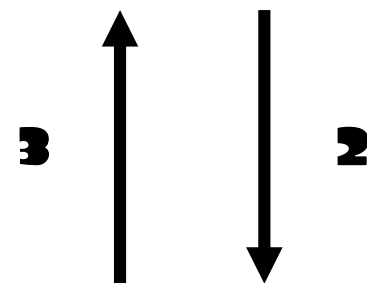
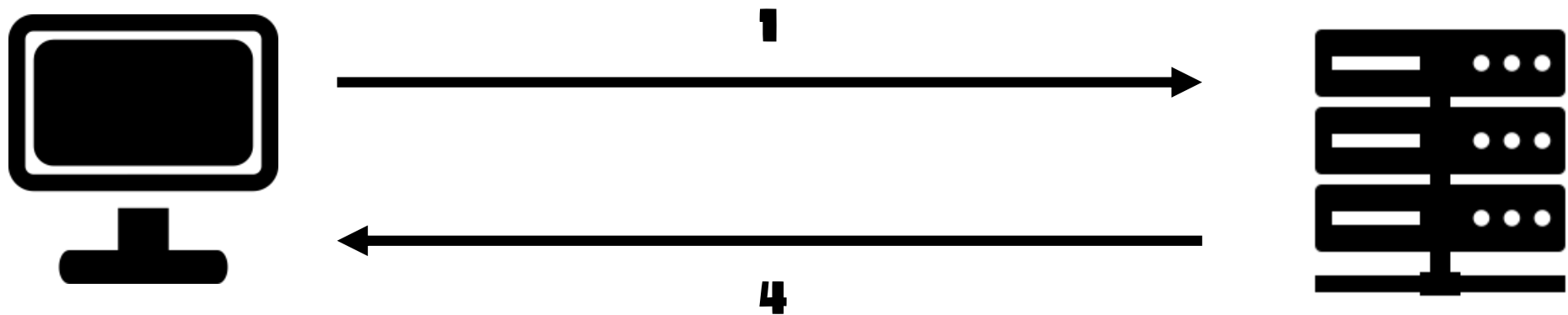


- 비동기 지원
- 생산성 증대
- 성능과 안전성
  - [PayPal](#)에서는 요청 처리 속도와 응답 시간에서 Node.js가 Java보다 좋은 퍼포먼스를 보임
- Npm 등 다양한 모듈과 오픈 소스 제공
- Node.js 커뮤니티의 발달

# 만들었던 서비스

- `//sori.noe.systems/delivery/`
- 원 페이지 서비스
- 송장번호로 택배를 추적해주는 서비스
- Express 이용

# 서비스가 굴러가는 구조



 **LOGEN** 로젠택배

# 서비스가 굴러가는 구조

- 클라이언트에서 데이터 요청
- 서버에서 운송장 조회 데이터 리스폰스 요청
  - GET <https://www.ilogen.com/web/personal/trace/>:운송장번호
- 데이터 가공
- API respons를 JSON 형태로 반환
- 클라이언트에서 도착한 리스폰스를 기반으로 랜더링



# 서비스가 굴러가는 구조

ilogen.com/web/personal/trace/94157469604

🏠

개인택배

▼

배송조회

▼

배송조회

운송장 번호로 조회

고객정보로 조회

예약번호로 조회

물품정보

송장번호	941-5746-9604	상품명	M*****
집하일자	2019-07-16	배송지점	영도
집하지점	북서대문	수량	1
보내시는 분	벨**	받으시는 분	강**
주소	부산 영도구 동삼동		

배송정보

01방문예정 > 02물품수거 > 03이동중 > 04배송중 > 05배송완료

# 서비스가 굴러가는 구조

**Request URL:** `https://www.ilogen.com/web/personal/trace/94157469604`

**Request Method:** GET

**Status Code:** 🟢 200 200

**Remote Address:** 203.247.141.65:443

**Referrer Policy:** no-referrer-when-downgrade

```
let parse = await new Promise((resolve, reject) => {
  request.get(
    {
      url: "https://www.ilogen.com/web/personal/trace/" + trackId,
      headers: {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.97 Safari/537.36",
        "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3",
        "Accept-Language": "ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7,und;q=0.6"
      },
    },
    (err, response, html) => err ? console.log(err) : resolve(html)
  )
})
console.log(parse)
```

# 서비스가 굴러가는 구조

```
<div class="subTit pdInfoTit mgt100">물 품 정 보 </div>
<table class="horizon pdInfo">
  <caption>배 송 조 회 물 품 정 보 </caption>
  <colgroup>
    <col style="width:200px;">
    <col style="width:407px;">
    <col style="width:200px;">
    <col style="width:407px;">
  </colgroup>
  <tbody>
    <tr>
      <td class="tit">송 장 번 호 </td>
      <td>941-5746-9604</td>
      <td class="tit">상 품 명 </td>
      <td>M*****</td>
    </tr>
    <tr>
      <td class="tit">집 하 일 자 </td>
      <td>2019-07-16</td>
      <td class="tit">배 송 지 점 </td>
      <td>영 도 </td>
    </tr>
    <tr>
      <td class="tit">집 하 지 점 </td>
      <td>북 서 대 문 </td>
      <td class="tit">수 량 </td>
      <td>1</td>
    </tr>
    <tr>
      <td class="tit">보 내 시 는 분 </td>
      <td>벨 **</td>
      <td class="tit">받 으 시 는 분 </td>
      <td>강 **</td>
    </tr>
    <tr>
      <td class="tit">주 소 </td>
      <td colspan="3">부 산 영 도 구 동 삼 동 </td>
    </tr>
  </tbody>
</table>
```

# 서비스가 굴러가는 구조

## cheerio

1.0.0-rc.3 • Public • Published 10 months ago

[Readme](#)[Explore](#) BETA[6 Dependencies](#)[10,249 Dependents](#)[60 Versions](#)

## cheerio

Fast, flexible & lean implementation of core jQuery designed specifically for the server.

build passing coverage 99% gitter join chat backers 14 sponsors 29

```
const cheerio = require('cheerio')
const $ = cheerio.load('<h2 class="title">Hello world</h2>')

$('h2.title').text('Hello there!')
$('h2').addClass('welcome')

$.html()


//> <html><head></head><body><h2 class="title welcome">Hello there!</h2></body></html>
```

### Install

```
> npm i cheerio
```

### Weekly Downloads

4,119,502



Version	License
1.0.0-rc.3	MIT

Unpacked Size	Total Files
112 kB	15

# 서비스가 굴러가는 구조

```
let parse = await new Promise((resolve, reject) => {
  request.get(
    {
      url: "https://www.ilogen.com/web/personal/trace/" + trackId,
      headers: {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.97 Safari/537.36",
        "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3",
        "Accept-Language": "ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7,und;q=0.6"
      },
    }, (err, response, html) => err ? console.log(err) : resolve(html))
  })
var $ = cheerio.load(parse)
data.from.name = $("body > div.contents.personal.tkSearch > section > div > div.tab_container > div > table.horizon.pdInfo > tbody > tr:nth-child(4) > td:nth-child(2)").text()
data.from.time = $("body > div.contents.personal.tkSearch > section > div > div.tab_container > div > table.data.tkInfo > tbody > tr:nth-child(1) > td:nth-child(1)").text()
data.to.name = $("body > div.contents.personal.tkSearch > section > div > div.tab_container > div > table.horizon.pdInfo > tbody > tr:nth-child(4) > td:nth-child(4)").text()
```

- Cheerio를 이용하여 데이터 가공
- JQuery에서 이용하는 방식으로 가공 가능

```
{
  "to": {
    "name": "강**",
    "time": "2019.07.18 19:19"
  },
  "from": {
    "name": "벨**",
    "time": "2019.07.16 18:56"
  },
  "state": {
    "id": "delivered",
    "text": "배송 완료"
  },
  "progresses": [
    {
      "time": "2019.07.16 18:56",
      "status": {
        "text": "이동중",
        "id": "in_transit"
      },
      "location": {
        "name": "북서대문"
      }
    }
  ]
}
```

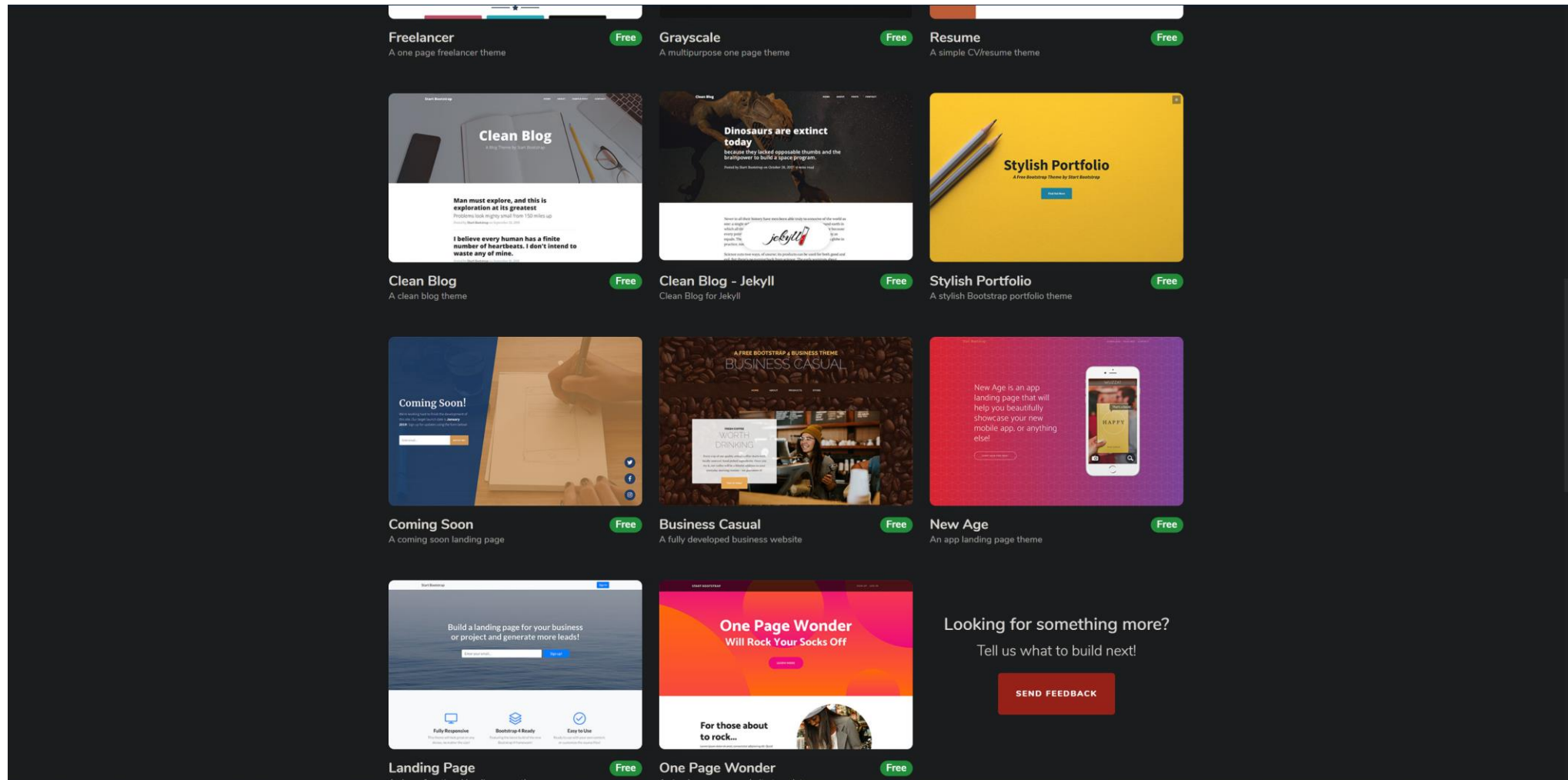
# cheerio

- Jquery에서 사용되는 메소드들을 모두 사용이 가능하다
  - 셀렉터에 기반한 DOM 조작
  - 노드 요소 및 노드 속성(아이디 및 클래스)을 셀렉터 생성을 위한 기준으로 사용
- 프론트 엔드 개발자도 편하게 개발이 가능.
  - 수인 \$에 Html 소스를 load해서 사용을 많이 함.
- End to end 벤치마크에서 JS dom 보다 8배 빠른 퍼포먼스를 보였음.
  - 페이지 데이터의 양이 많아도 빠른 속도를 보인다.

# Bootstrap

- **Front-end 프레임워크**
  - 필요한 것은 보통 다 구현되어 있다.
  - 가져다가 쓰기만 하면 된다.
  - 존재하는 템플릿들이 많다.
  - = 나같은 똥 손도 나름대로 예쁘게 만들 수 있다.
- **단점. 너무 양산형 디자인이다.**

# Bootstrap





# Bootstrap

```
$('#postNumSubmit').click(function () {  
    getPostInfo();  
});  
$.ajax({  
    url: "http://sori.noe.systems/delivery/api/carriers/" + $("#carriers").val().toString() + "/tracks/" + $("#numbers").val().toString(),  
    type: "GET",  
    dataType: "json",  
    beforeSend: function (xhr) {  
        xhr.setRequestHeader("Authorization", userData);  
    },  
    success: function (data) {  
        console.log(data)  
        $("#postResult").text("조회 결과 - " + data.state.text);  
        $("#fromInfo > strong").text("보내는 분. " + data.from.name);  
        $("#fromDate").text(moment(data.from.time).format("YYYY-MM-DD kk:mm:ss"));  
        $("#toInfo > strong").text("받는 분. " + data.to.name);  
        if (data.to.time == null) {  
            data.to.time = "배송 중"  
            $("#toDate").text(data.to.time);  
        }  
        else  
            $("#toDate").text(moment(data.to.time).format("YYYY-MM-DD kk:mm:ss"));  
        let tmp = "";  
  
        let dateCnt = moment(data.progresses[0].time).format("YYYY-MM-DD");  
        tmp += "<strong>" + dateCnt + "</strong><br>";  
        for (let i = 0; i < data.progresses.length; i++) {  
            if (dateCnt != moment(data.progresses[i].time).format("YYYY-MM-DD")) {  
                dateCnt = moment(data.progresses[i].time).format("YYYY-MM-DD")
```

# Bootstrap

- **jquery** 를 이용해서 **ajax**를 통한 서버 통신
- **Event listener**를 이용한 함수 실행
- 등등 위에서 말한 것들이 대부분 구현되었기 때문에 생산성이 향상된다.

# 배포

- **AWS(amazon web service)**
  - 학생은 에듀케이트 지원으로 100불 지원

만료 날짜	크레딧 이름	사용한 금액	남은 금액	적용 가능 제품
2020. 08. 31.	EDU_ENG_FY2019_IC-Q3_3_100USD	US\$85.78	US\$14.22	<a href="#">전체 목록 보기</a>
2022. 01. 31.	EDU_ENG_FY2020_IC_Q1_1_AWSEDUCATE_PROMO_25USD	US\$0.00	US\$25.00	<a href="#">전체 목록 보기</a>

- **Pm2 이용**
  - Nodemon과 많이 사용되는 매니지먼트 툴
  - 중간에 서비스가 죽어도 재시작 해준다.

# 배포

- **Nginx로 도메인 라우팅**

```
server {  
    listen 80;  
    server_name sori.noe.systems;  
    location / {  
        proxy_pass http://localhost:5051;  
    }  
}
```

service nginx reload // **중지하지 않고 설정만 적용**

service nginx restart // **중지하고 재시작**

**Q&A**

**감사합니다**