



Web Service Development

What we learned



Likelion

What We Learned

- HTML
- CSS
- Chrome developer tools
- jQuery
- JavaScript
- AJAX
- Session
- Flask
- GET/POST
- Client/Server
- Database
- ORM
- MVC
- Python
- Git
- Crawler
- Jinja
- Bootstrap
- Pusher
- BeautifulSoup
- Debugging
- Python data types
- DB data types
- JSON
- Functions
- Variables
- If-else
- While loop
- For loop
- jQuery plugins
- Python modules
- API

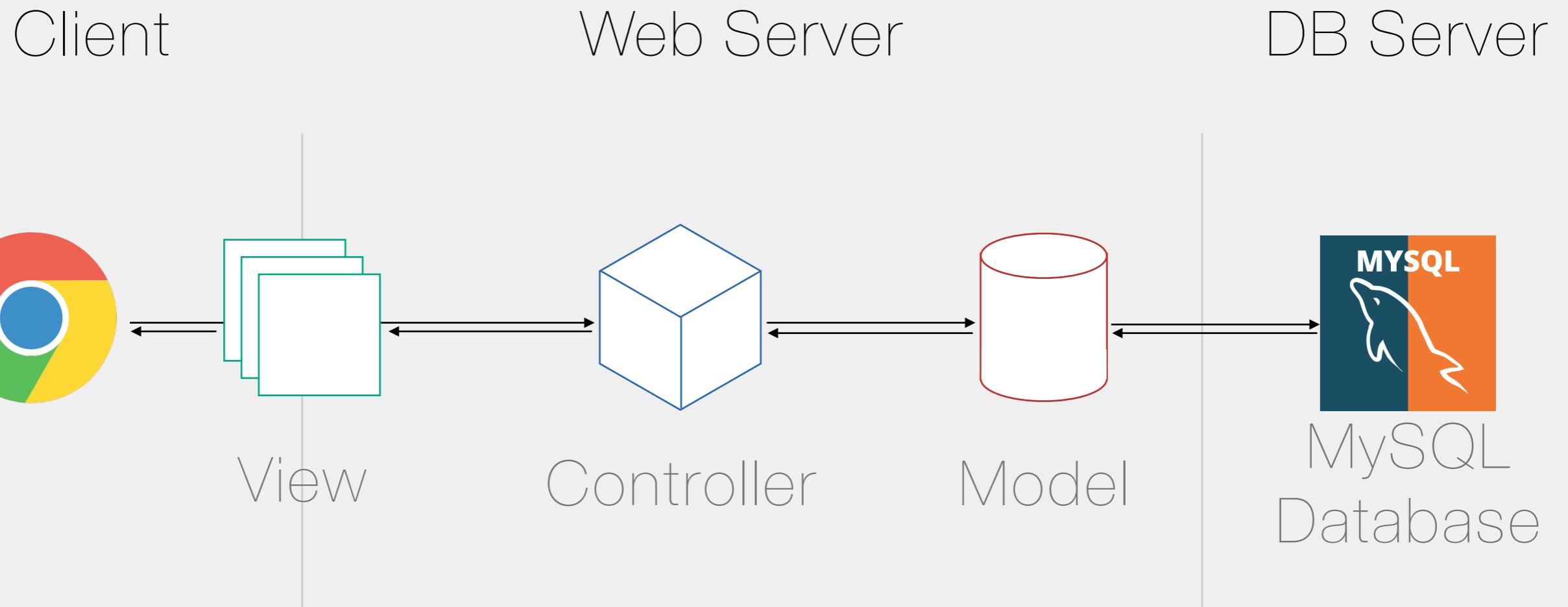


Web Service Development

- Client - Server
- Request
- HTML, CSS, JavaScript(jQuery)
- Python
- HTML Form
- Session
- AJAX
- JSON
- MVC
- Database - ORM



Web Service Structure



Client-Server



Likelion

Client-Server

in Web Services

Web Client

Client Computer + Client Program

Web Server

Server Computer + Server Program



Client-Server

in Web Services

Web Client



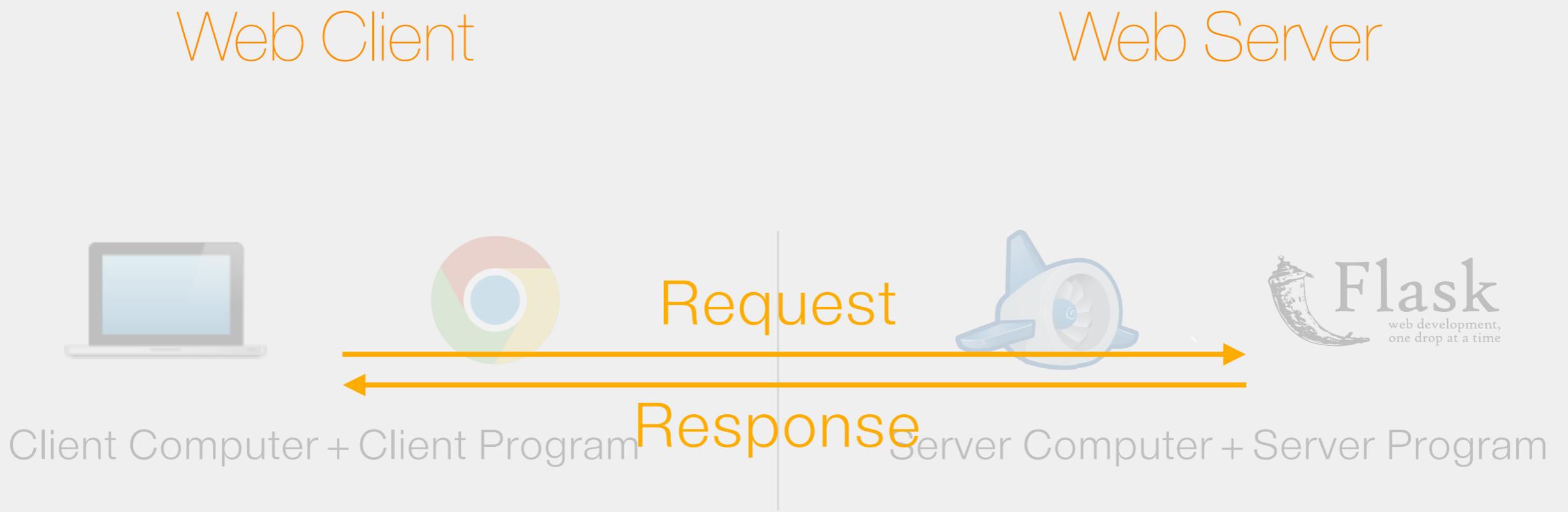
Client Computer + Client Program

Web Server



Server Computer + Server Program

Client-Server in Web Services



Client-Server

in Web Services

- Client : 서버로 요청(request) 전송
- Server : 클라이언트의 요청을 받아서 처리한 후 응답(response) 돌려줌



Client-Server

Client

- Web Browser (Chrome)

Server

- Web Server (Flask)



Request

HTTP Request

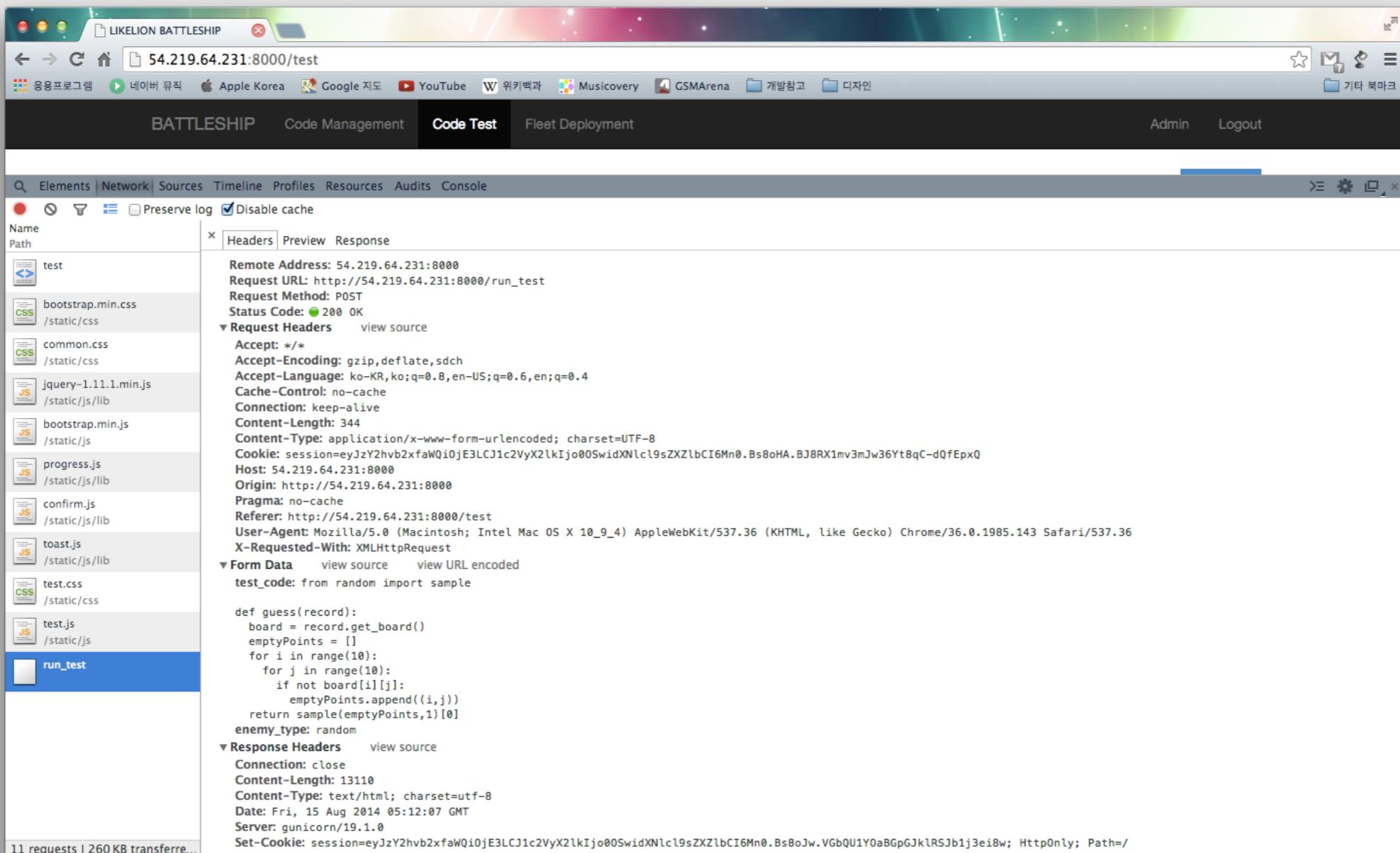


Request

HTTP Request : URL + Data



Request



Likelion

Request

The screenshot shows a web browser window titled "LIKELION BATTLESHIP" with the URL "54.219.64.231:8000/test". The browser's address bar also displays "54.219.64.231:8000/run_test". A yellow callout box highlights this URL. The browser interface includes a toolbar with various icons and a menu bar with "Admin" and "Logout". Below the toolbar, there are tabs for "Elements", "Network", "Sources", "Timeline", and "Profiles". The "Network" tab is active, showing a list of files and a detailed view of a request for "run_test". The detailed view shows the following information:

Request URL: http://54.219.64.231:8000/run_test

Request Headers:

```
Accept: */*
Accept-Encoding: gzip,deflate,sdch
Accept-Language: ko-KR,ko;q=0.8,en-US;q=0.6,en;q=0.4
Cache-Control: no-cache
Connection: keep-alive
Content-Length: 344
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Cookie: session=eyJzY2hb2xfawQi0jE3LCJ1c2VyX2lkIjo0SwidXNlcl9sZXZlbCI6Mn0.Bs8oHA.BJ8RX1mv3mJw36Yt8qC-dQfEpxQ
Host: 54.219.64.231:8000
Origin: http://54.219.64.231:8000
Pragma: no-cache
Referer: http://54.219.64.231:8000/test
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/36.0.1985.143 Safari/537.36
X-Requested-With: XMLHttpRequest
```

Form Data:

```
test_code: from random import sample

def guess(record):
    board = record.get_board()
    emptyPoints = []
    for i in range(10):
        for j in range(10):
            if not board[i][j]:
                emptyPoints.append((i,j))
    return sample(emptyPoints,1)[0]
enemy_type: random
```

Response Headers:

```
Connection: close
Content-Length: 13110
Content-Type: text/html; charset=utf-8
Date: Fri, 15 Aug 2014 05:12:07 GMT
Server: gunicorn/19.1.0
Set-Cookie: session=eyJzY2hb2xfawQi0jE3LCJ1c2VyX2lkIjo0SwidXNlcl9sZXZlbCI6Mn0.Bs8oJw.VGbQU1Y0aBGpGJklRSJb1j3ei8w; HttpOnly; Path=/
```

At the bottom of the Network tab, it says "11 requests | 260 KB transferred...".



Likelion

Request

LIELION BATTLESHIP

54.219.64.231:8000/test

BATTLESHIP Code Management Code Test Fleet Deployment Admin Logout

Elements Network Sources Timeline Profiles Resources Audits Console

Name Path

test

bootstrap.min.css /static/css

common.css /static/css

jquery-1.11.1.min.js /static/js/lib

bootstrap.min.js /static/js

progress.js /static/js/lib

confirm.js /static/js/lib

toast.js /static/js/lib

test.css /static/css

test.js /static/js

run_test

Preserve log Disable cache

Headers Preview Resp

Remote Address: 54.219.64.231
Request URL: http://54.219.64.231:8000/test
Request Method: POST
Status Code: 200 OK

Request Headers

Accept: */*
Accept-Encoding: gzip, deflate
Accept-Language: ko
Cache-Control: no-cache
Connection: keep-alive
Content-Length: 344
Content-Type: application/x-www-form-urlencoded
Cookie: session=eyJzY2hv...
Host: 54.219.64.231
Origin: http://54.219.64.231
Pragma: no-cache
Referer: http://54.219.64.231:8000/test
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/39.0.2171.95 Safari/537.36

Form Data

test_code: from random import sample

```
def guess(record):
    board = record.get_board()
    emptyPoints = []
    for i in range(10):
        for j in range(10):
            if not board[i][j]:
                emptyPoints.append((i,j))
    return sample(emptyPoints,1)[0]
```

enemy_type: random

Response Headers

Connection: close
Content-Length: 13110
Content-Type: text/html; charset=utf-8
Date: Fri, 15 Aug 2014 05:12:07 GMT
Server: gunicorn/19.1.0
Set-Cookie: session=eyJzY2hv...; HttpOnly; Path=/

11 requests | 260 KB transferred...



Likelion

Request

HTTP Request : URL + Data



Request > URL

URL : 요청을 보낼 위치



Request > URL

http://54.219.64.231:8000/test

서버 주소



Request > URL

http://54.219.64.231:8000/test

서버 내 위치



Request > URL

```
# controllers/test.py

@app.route('/test')
def test():
    return render_template('test.html')
```



Request > URL

```
# controllers.py  
@app.route('/test')  
def test():  
    return render_template('test.html')
```

이 위치로 요청을 받으면



Request > URL

```
# controllers/test.py  
@app.route('/test')  
def test():  
    return render_template('test.html')
```

이 함수를 호출



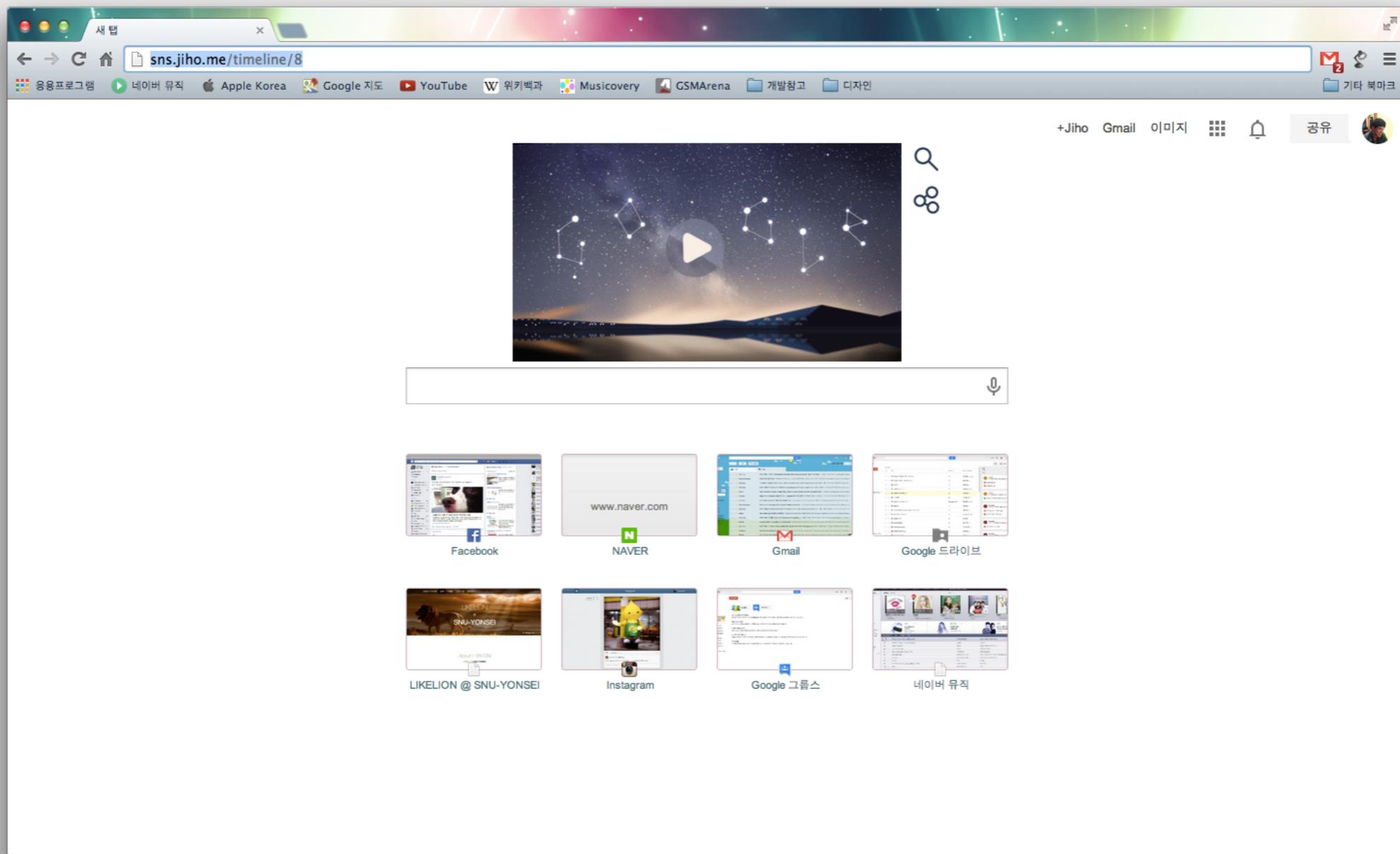
Request > URL

```
# controllers/test.py  
  
@app.route('/test')  
def test():  
    return render_template('test.html')
```

return 값 : Response



Request Handling



Likelion

Request Handling

```
# sns.jiho.me/timeline/8
@app.route('/', defaults={'wall_id':0})
@app.route('/timeline/<int:wall_id>')
def timeline(wall_id) :
    if not is_login():
        return redirect(url_for('login'))
    if wall_id == 0:
        wall_id = session['user_id']

    session['wall_id'] = wall_id
    session['wall_username'] = get_user(wall_id).username

    context = {
        'posts':get_wall_posts(wall_id,0),
        'BS':BS
    }
    postList = render_template('post_list.html',context=context)
    context = {
        'post_list':postList
    }
    return render_template('timeline.html', context=context)
```



Request Handling

```
# sns.jiho.me/timeline/8
@app.route('/', defaults={'wall_id':0}) 클라이언트의 요청을 받아서
@app.route('/timeline/<int:wall_id>')
def timeline(wall_id) :
    if not is_login():
        return redirect(url_for('login'))
    if wall_id == 0:
        wall_id = session['user_id']

    session['wall_id'] = wall_id
    session['wall_username'] = get_user(wall_id).username

    context = {
        'posts':get_wall_posts(wall_id,0),
        'BS':BS
    }
    postList = render_template('post_list.html',context=context)
    context = {
        'post_list':postList
    }
    return render_template('timeline.html', context=context)
```



Request Handling

```
# sns.jiho.me/timeline/8
@app.route('/', defaults={'wall_id':0})
@app.route('/timeline/<int:wall_id>')
def timeline(wall_id) :
    if not is_login():
        return redirect(url_for('login'))
    if wall_id == 0:
        wall_id = session['user_id']

    session['wall_id'] = wall_id
    session['wall_username'] = get_user(wall_id).username

    context = {
        'posts':get_wall_posts(wall_id,0),
        'BS':BS
    }
    postList = render_template('post_list.html',context=context)
    context = {
        'post_list':postList
    }
    return render_template('timeline.html', context=context)
```

처리한 후



Likelion

Request Handling

```
# sns.jiho.me/timeline/8
@app.route('/', defaults={'wall_id':0})
@app.route('/timeline/<int:wall_id>')
def timeline(wall_id) :
    if not is_login():
        return redirect(url_for('login'))
    if wall_id == 0:
        wall_id = session['user_id']

    session['wall_id'] = wall_id
    session['wall_username'] = get_user(wall_id).username

    context = {
        'posts':get_wall_posts(wall_id,0),
        'BS':BS
    }
    postList = render_template('post_list.html',context=context)
    context = {
        'post_list':postList
    }
    return render_template('timeline.html', context=context)
```

응답을
돌려줌



Request Handling

```
# sns.jiho.me/timeline/8
@app.route('/', defaults={'wall_id':0})
@app.route('/timeline/<int:wall_id>')
def timeline(wall_id) :
    if not is_login():
        return redirect(url_for('login'))
    if wall_id == 0:
        wall_id = session['user_id']

    session['wall_id'] = wall_id
    session['wall_username'] = get_user(wall_id).username

    context = {
        'posts':get_wall_posts(wall_id,0),
        'BS':BS
    }
    postList = render_template('post_list.html',context=context)
    context = {
        'post_list':postList
    }
    return render_template('timeline.html', context=context) Response
```



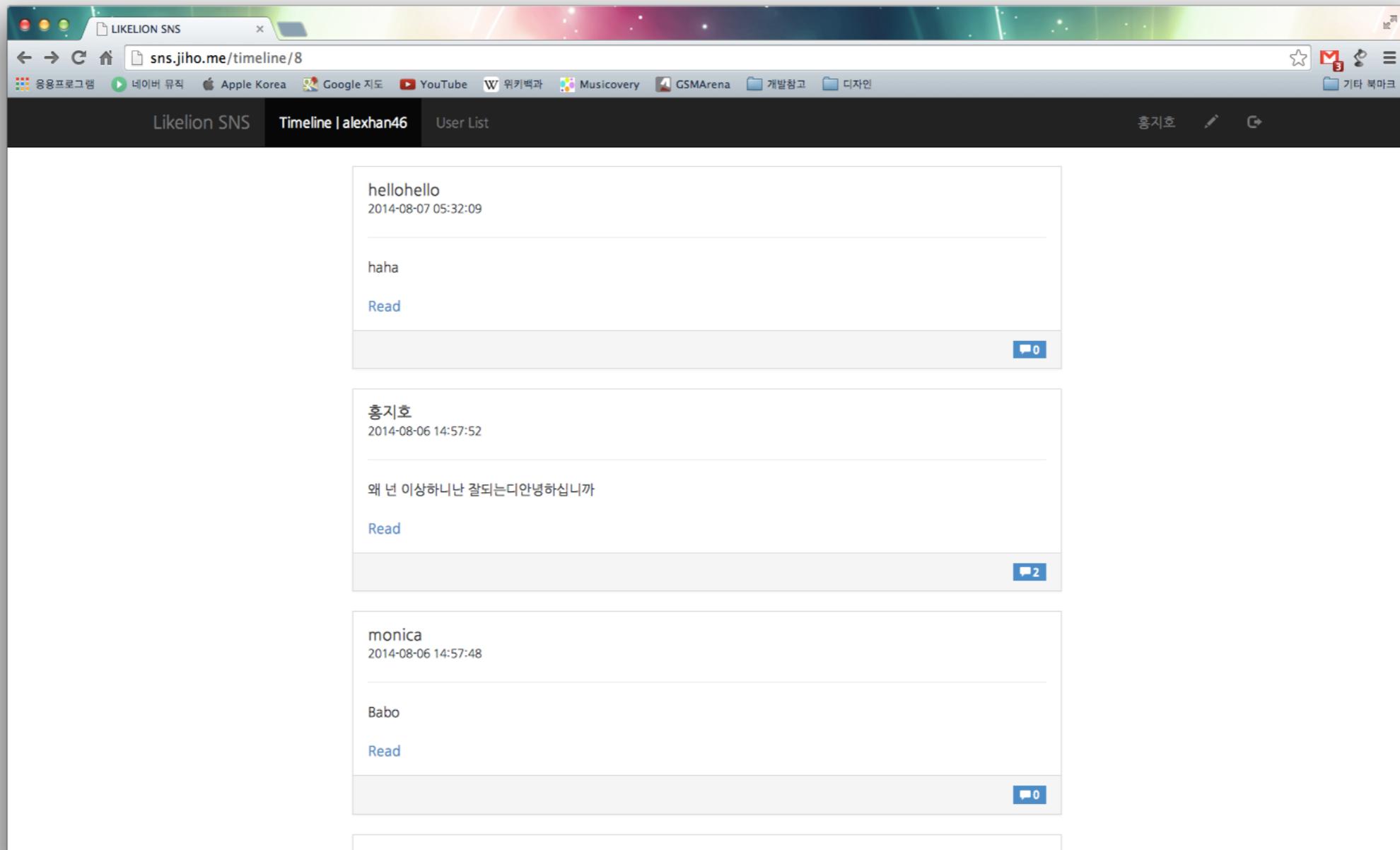
Request Handling

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1">
6   <title>LIKELION SNS</title>
7
8
9   <link rel="stylesheet" href="/static/css/bootstrap.min.css">
10
11  <link rel="stylesheet" href="/static/css/common.css">
12  <script src="/static/js/jquery-1.11.1.min.js"></script>
13  <script src="/static/js/bootstrap.min.js"></script>
14  <script src="/static/js/common.js"></script>
15
16
17
18 <link rel="stylesheet" href="/static/css/timeline.css">
19
20
21
22 </head>
23 <body>
24
25
26   <nav class="navbar navbar-default navbar-inverse navbar-fixed-top" role="navigation">
27     <div class="container">
28       <!-- Brand and toggle get grouped for better mobile display -->
29       <div class="navbar-header">
30         <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#bs-example-navbar-collapse-1">
31           <span class="sr-only">Toggle navigation</span>
32           <span class="icon-bar"></span>
33           <span class="icon-bar"></span>
34           <span class="icon-bar"></span>
35         </button>
36         <a class="navbar-brand" href="/">Likelion SNS</a>
37       </div>
38
39       <!-- Collect the nav links, forms, and other content for toggling -->
40
41       <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
42         <ul class="nav navbar-nav">
43           <li
44             class="active">
45             <a href="/timeline/8">Timeline | alexhan46</a></li>
46           <li><a href="/user_list">User List</a>
47           </li>
48
49         </ul>
50
51         <ul class="nav navbar-nav navbar-right">
52           <li><a href="/timeline/6">총지호</a></li>
```



Likelion

Request Handling



Likelion

Client-Side

Front-end



Likelion

HTML, CSS, JavaScript

Client-Side



HTML, CSS, JavaScript

Client-Side

```
17 > <input type="password" class="form-control" id="input-password-confirm" placeholder="Confirm password" name="password-confirm" required>
18 </div>
19 <div class="form-group">
20   <label for="input-username">
21     Username</label>
22   <input type="text" class="form-control" id="input-username" name="username" placeholder="Username" required>
23 </div>
24
25 <div class="form-group">
26   <label class="radio-inline">
27     <input type="radio" name="gender" value="F" checked>
28       Female
29   </label>
30   <label class="radio-inline">
31     <input type="radio" name="gender" value="M"> Male
32   </label>
33 </div>
34 <div class="form-group">
35   <label for="input-mobile">Phone</label>
36   <input type="text" class="form-control" id="input-mobile" name="mobile" placeholder="Phone" required>
37 </div>
38
39 <div style="font-size: 10px; margin-top: 20px; text-align: right; background-color: #f2f2f2; padding: 5px; border-radius: 5px; width: fit-content; margin-left: auto; margin-right: 0; position: absolute; bottom: 100%; left: 50%; transform: translateX(-50%);>
40   <span>Copyright © 2023. All Rights Reserved. This application is for educational purposes only. It is not intended for commercial use or distribution. The source code is provided as-is without warranty or guarantee of any kind. The developer is not responsible for any damages or losses that may result from the use of this application. Use at your own risk.</span>
41 </div>
42
43 <script>
44   // Load Google Fonts
45   document.head.innerHTML += `@import url('https://fonts.googleapis.com/earlyaccess/nanumgothic.css');`;
46
47   // Show toast message
48   function show_toast(message, level, style) {
49     level = level || 'default';
50
51     // Set custom styles
52     if(style) {
53       for(s in defaultStyle) {
54         if(!style[s]) {
55           style[s] = defaultStyle[s];
56         }
57       }
58     } else {
59       style = defaultStyle;
60     }
61
62     style.backgroundColor = color[level];
63
64     if($('div#toast').length > 0) {
65       setTimeout('show_toast("'+message+'","'+level+'",'+JSON.stringify(style)+')', 300);
66     } else {
67       var toast = '<div id="toast" style="text-align:center; position:fixed;bottom:20%;width:100%;padding:0px;maring:0px;display:none;z-index:9999;"><span>'+message+'</span></div>';
68       document.body.innerHTML += toast;
69       document.getElementById('toast').style.display = 'block';
70     }
71   }
72
73   // Hide toast message
74   function hide_toast() {
75     if($('div#toast').length > 0) {
76       setTimeout('document.getElementById("toast").style.display = "none";', 300);
77     }
78   }
79
80   // Check if user has accepted terms and conditions
81   function check_terms() {
82     if($('input#checkbox').is(':checked')) {
83       localStorage.setItem('accepted', true);
84     } else {
85       localStorage.removeItem('accepted');
86     }
87   }
88
89   // Get accepted status from local storage
90   const accepted = localStorage.getItem('accepted');
91
92   if(accepted === null) {
93     // User has not accepted terms and conditions
94     // Show terms and conditions page
95     window.location.href = 'terms.html';
96   } else if(accepted === 'true') {
97     // User has accepted terms and conditions
98     // Show main application page
99     window.location.href = 'index.html';
100   } else {
101     // User has not accepted terms and conditions
102     // Show terms and conditions page
103     window.location.href = 'terms.html';
104   }
105
106   // Hide loading indicator
107   document.getElementById('loading').style.display = 'none';
108
109   // Show main application page
110   window.location.href = 'index.html';
111
112 </script>
```



HTML, CSS, JavaScript

Client-Side



HTML, CSS, JavaScript

Client-Side



구조

HTML, CSS, JavaScript

Client-Side



구조



표현



HTML, CSS, JavaScript

Client-Side



구조



표현



동작

HTML, CSS, JavaScript

Client-Side



HTML

구조



CSS

표현



JS

동작



jQuery :
JS 라이브러리



Likelion

HTML

Likelion SNS

sns.jiho.me

Toggle navigation Likelion SNS

- Timeline | 흥지호
- User List
- 흥지호
- ...

흥지호

2014-08-11 03:18:14

...

Read
0

흥지호

2014-08-07 16:48:00

하하ㅏㅎ라라

Read
1

흥지호

2014-08-07 03:07:29

ㅁㅁㅁㅁㅁ

Read
0

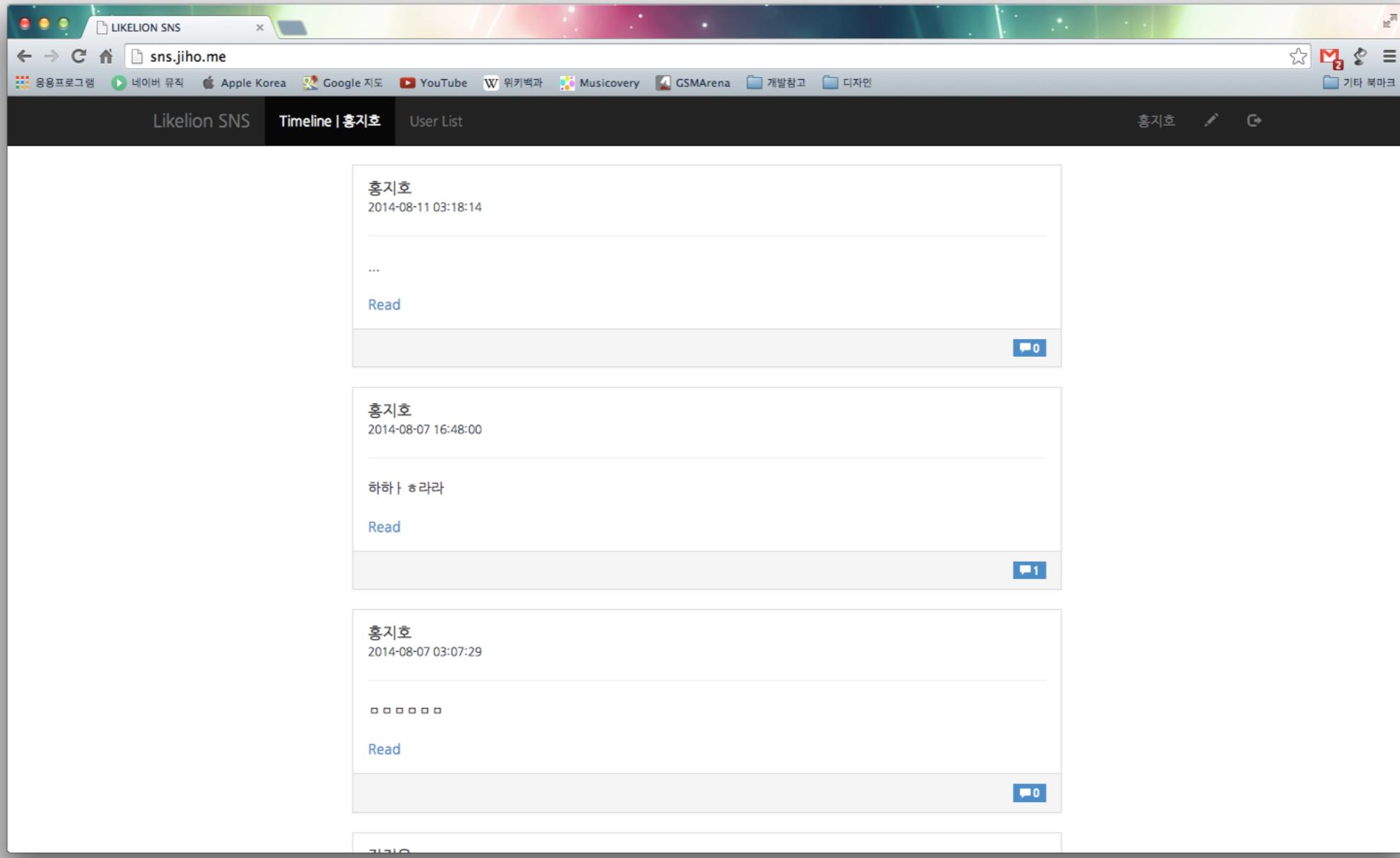
장진욱

2014-08-07 01:40:48



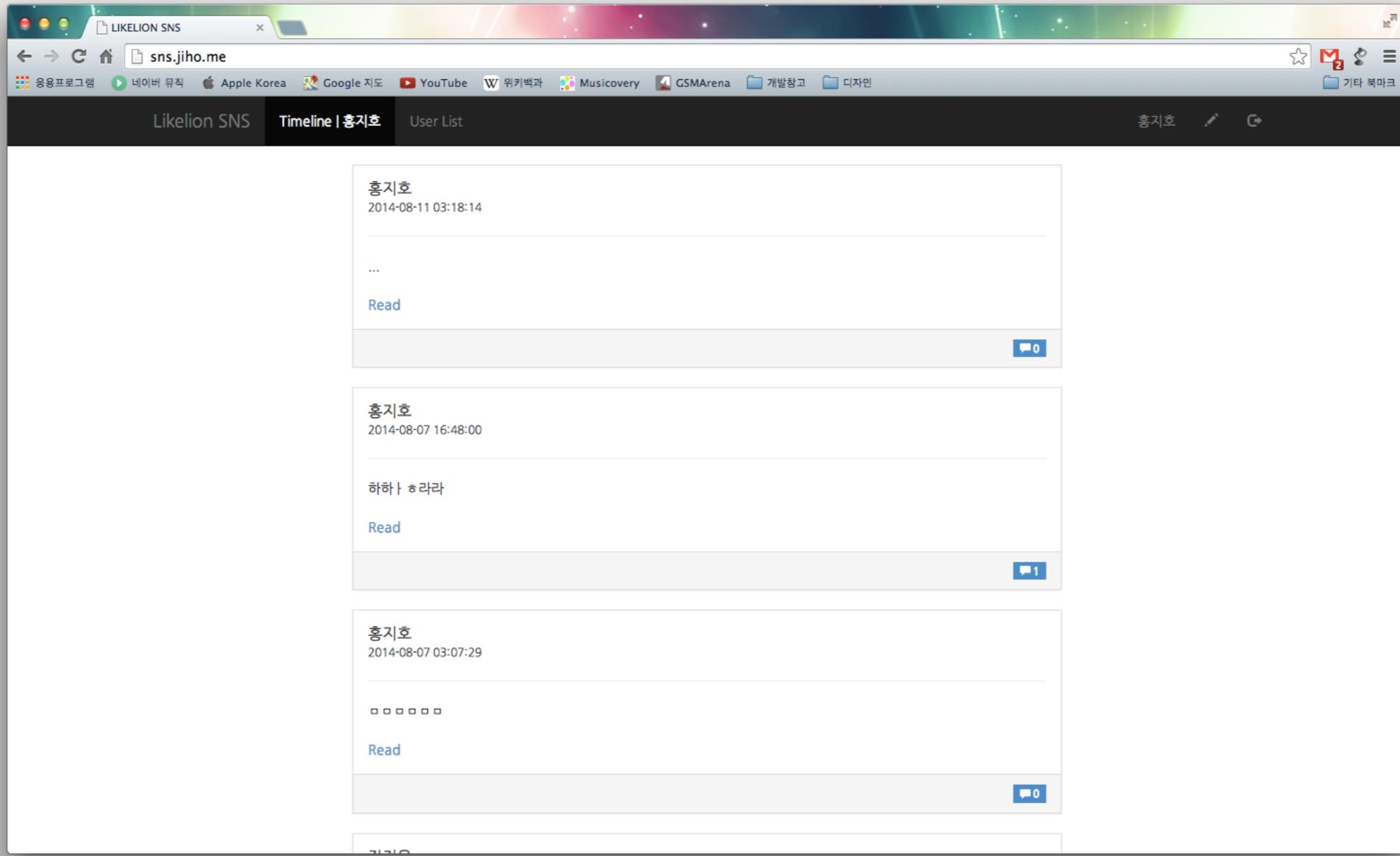
Likelion

HTML + CSS



Likelion

HTML+CSS+JavaScript



Likelion

Client-Server

Client

- Web Browser
- HTML, CSS, JavaScript

Server

- Web Server



HTML

웹 페이지의 구조



Likelion

<http://www.w3schools.com/>



Likelion

HTML

웹 페이지의 구조

```
<!DOCTYPE html>
<html>
  <head>

    </head>
    <body>

      </body>
    </html>
```



HTML

웹 페이지의 구조

```
<!DOCTYPE html>
<html> 이 문서는 HTML 문서다
      <head>

      </head>
      <body>

      </body>
</html>
```



HTML

웹 페이지의 구조

```
<!DOCTYPE html>
<html>
  <head>
    </head>
    <body>
      </body>
    </html>
```

HTML 문서의 시작과 끝



HTML

웹 페이지의 구조

```
<!DOCTYPE html>
<html>
  <head>
    </head>
  <body>
    </body>
</html>
```

문서에 대한 정보가 들어있는 부분
브라우저에는 표시되지 않음



HTML

웹 페이지의 구조

```
<!DOCTYPE html>
<html>
  <head>
    ...
  </head>
  <body>
    ...
  </body>
</html>
```

브라우저에 보이는
실제 내용이 들어가는 부분



HTML

웹 페이지의 구조

- 태그
- 속성
- 내용



HTML

웹 페이지의 구조

```
<a id="link-daum" class="link active" href="http://www.naver.com">다음</a>
```



HTML

웹 페이지의 구조

```
<a id="link-daum" class="link active" href="http://www.naver.com">다음</a>
```

시작 태그



HTML

웹 페이지의 구조

```
<a id="link-daum" class="link active" href="http://www.naver.com">다음</a>
```

종료 태그



HTML

웹 페이지의 구조

다음

태그 이름



HTML

웹 페이지의 구조

```
<a id="link-daum" class="link active" href="http://www.naver.com">다음</a>
```

속성명



HTML

웹 페이지의 구조

```
<a id="link-daum" class="link active" href="http://www.naver.com">다음</a>
```

속성값



HTML

웹 페이지의 구조

속성값은 큰 따옴표로 묶기

```
<a id="link-daum" class="link active" href="http://www.naver.com">다음</a>
```



HTML

웹 페이지의 구조

```
<a id="link-daum" class="link active" href="http://www.naver.com">다음</a>
```

여러 개의 클래스는 공백으로 구분



HTML

웹 페이지의 구조

```
<a id="link-daum" class="link active" href="http://www.naver.com">다음</a>
```

내용



HTML

웹 페이지의 구조

The screenshot shows a web browser window with the developer tools open. The left pane displays the HTML structure of a page titled "Likelion SNS". The right pane shows the "Styles" tab of the developer tools, listing CSS rules applied to the page, primarily from "bootstrap.min.css" and "common.css".

Developer Tools - http://sns.jiho.me/

Styles

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body>
    <nav class="navbar navbar-default navbar-inverse navbar-fixed-top" role="navigation">...</nav>
    <div class="container">
      ::before
      <div id="post-list" class="col-md-8 col-md-offset-2" data-post-id="38">
        <div class="panel panel-default" data-post-id="38">
          <div class="panel-body">
            ::before
            <h3 class="panel-title">...</h3>
            <small>2014-08-11 03:18:14</small>
            <hr>
            ...
            <br>
            <br>
            <a href="/read/38">Read</a>
            ::after
          </div>
          <div class="panel-footer">...</div>
        </div>
        <div class="panel panel-default" data-post-id="28">...</div>
        <div class="panel panel-default" data-post-id="27">...</div>
        <div class="panel panel-default" data-post-id="26">...</div>
        <div class="panel panel-default" data-post-id="21">...</div>
        ::after
      </div>
      <script src="/static/js/timeline.js"></script>
    </body>
</html>
```

Computed

Event Listeners

Inherited from html

Find in Styles



HTML

웹 페이지의 구조

The screenshot displays a web browser window with the developer tools open. The main content area shows a timeline of posts from a social networking site. One specific post is highlighted with a yellow box, and its HTML structure is shown in the element inspector. The CSS styles for this element are listed on the right, including Bootstrap's panel classes and common.css rules. The browser's address bar shows the URL 'sns.jiho.me'.

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body>
    <nav class="navbar navbar-default navbar-inverse navbar-fixed-top" role="navigation">...</nav>
    <div class="container">
      ::before
      <div id="post-list" class="col-md-8 col-md-offset-2">
        <div class="panel panel-default" data-post-id="38">
          <div class="panel-body">
            ::before
            <h3 class="panel-title">...</h3>
            <small>2014-08-11 03:18:14</small>
            <hr>
            ...
            <br>
            <br>
            <a href="/read/38">Read</a>
            ::after
          </div>
          <div class="panel-footer">...</div>
        </div>
        <div class="panel panel-default" data-post-id="28">...</div>
        <div class="panel panel-default" data-post-id="27">...</div>
        <div class="panel panel-default" data-post-id="26">...</div>
        <div class="panel panel-default" data-post-id="21">...</div>
        ::after
      </div>
      <script src="/static/js/timeline.js"></script>
    </body>
</html>
```



HTML

웹 페이지의 구조

The screenshot shows a web browser window with the developer tools open. The browser is displaying a website called "Likelion SNS". The developer tools are showing the "Elements" tab with the HTML structure and CSS styles for the selected element.

The HTML structure is as follows:

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body>
    <nav class="navbar navbar-default navbar-inverse navbar-fixed-top" role="navigation">...</nav>
    <div class="container">
      ::before
      <div id="post-list" class="col-md-8 col-md-offset-2">
        <div class="panel panel-default" data-post-id="38">
          <div class="panel-body">
            ::before
            <h3 class="panel-title">...</h3>
            <small>2014-08-11 03:18:14</small>
            <hr>
            ...
            <br>
            <br>
            <a href="/read/38">Read</a>
            ::after
          </div>
          <div class="panel-footer">...</div>
        </div>
        <div class="panel panel-default" data-post-id="28">...</div>
        <div class="panel panel-default" data-post-id="27">...</div>
        <div class="panel panel-default" data-post-id="26">...</div>
        <div class="panel panel-default" data-post-id="21">...</div>
      </div>
      ::after
    </div>
    <script src="/static/js/timeline.js"></script>
  </body>
</html>
```

The CSS styles for the selected element (the first post title) are:

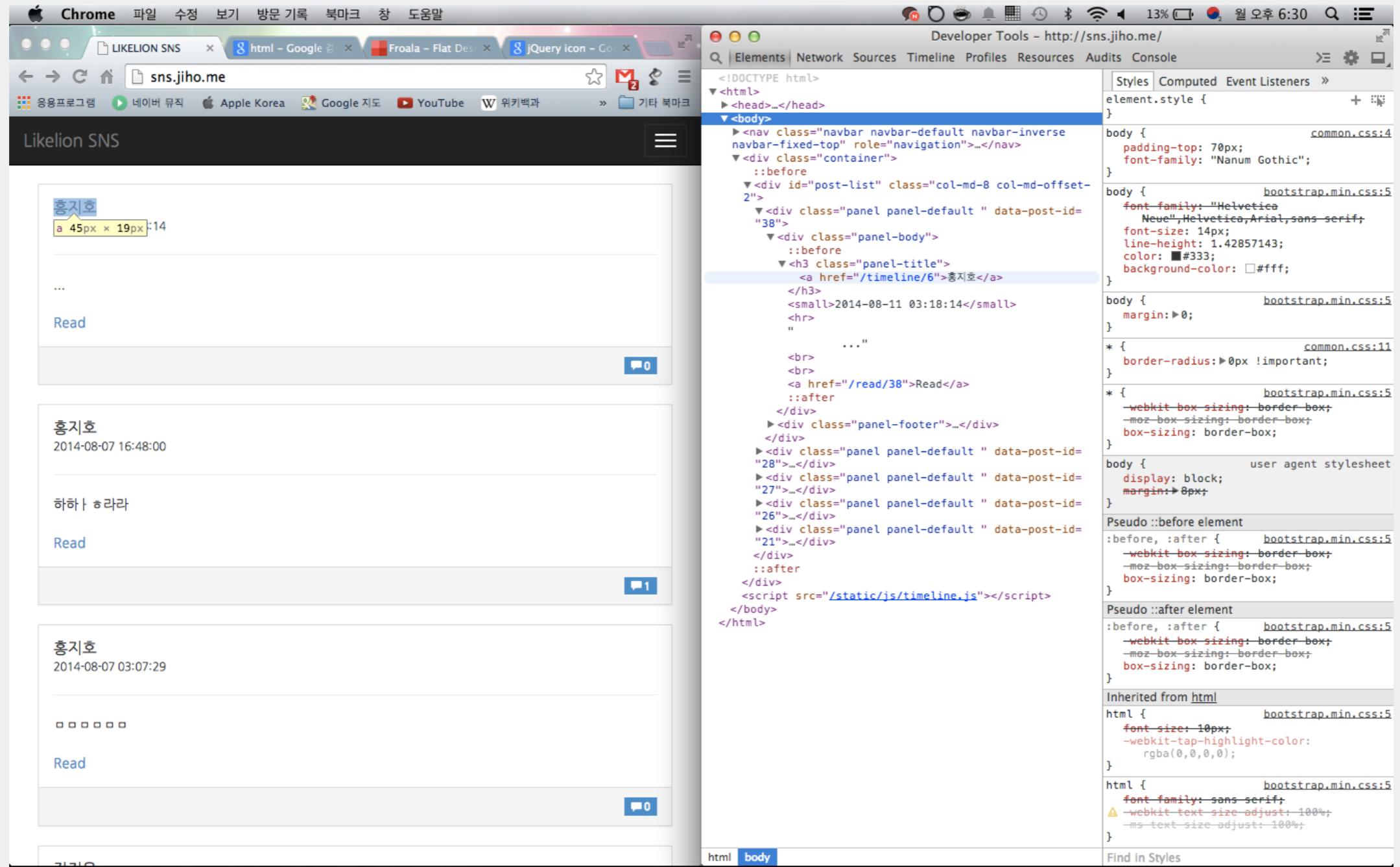
```
element.style {
}
body {
  padding-top: 70px;
  font-family: "Nanum Gothic";
}
body {
  font-family: "Helvetica Neue", Helvetica, Arial, sans-serif;
  font-size: 14px;
  line-height: 1.42857143;
  color: #333;
  background-color: #fff;
}
body {
  margin: 0;
}
* {
  border-radius: 0px !important;
}
* {
  -webkit-box-sizing: border-box;
  -moz-box-sizing: border-box;
  box-sizing: border-box;
}
body {
  user agent stylesheet
  display: block;
  margin: 0px;
}
Pseudo ::before element
:before, :after {
  -webkit-box-sizing: border-box;
  -moz-box-sizing: border-box;
  box-sizing: border-box;
}
Pseudo ::after element
:before, :after {
  -webkit-box-sizing: border-box;
  -moz-box-sizing: border-box;
  box-sizing: border-box;
}
Inherited from html
html {
  font-size: 10px;
  -webkit-tap-highlight-color: rgba(0,0,0,0);
}
html {
  font-family: sans-serif;
  -webkit-text-size-adjust: 100%;
  -ms-text-size-adjust: 100%;
}
```

The developer tools also show the "Computed" and "Event Listeners" tabs.



HTML

웹 페이지의 구조



Likelion

Client-Server

Client

- Web Browser
- HTML, CSS, JavaScript

Server

- Web Server



CSS

웹 페이지의 표현



Likelion

<http://www.w3schools.com/>



Likelion

CSS

웹 페이지의 표현

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Likelion SNS</title>

  <script src="//code.jquery.com/jquery-1.11.0.min.js"></script>

  <!-- Latest compiled and minified CSS -->
  <link rel="stylesheet" href="//maxcdn.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap.min.css">

  <!-- Latest compiled and minified JavaScript -->
  <script src="//maxcdn.bootstrapcdn.com/bootstrap/3.2.0/js/bootstrap.min.js"></script>
  >
  <link rel="stylesheet" href="{{ url_for('static', filename='css/common.css') }}>

</head>
<body>

  <nav class="navbar navbar-default" role="navigation">
    <div class="container">
      <!-- Brand and toggle get grouped for better mobile display -->
      <div class="navbar-header">
```



CSS

웹 페이지의 표현

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Likelion SNS</title>

    <script src="//code.jquery.com/jquery-1.11.0.min.js"></script>

    <!-- Latest compiled and minified CSS -->
    <link rel="stylesheet" href="//maxcdn.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap.min.css">

    <!-- Latest compiled and minified JavaScript -->
    <script src="//maxcdn.bootstrapcdn.com/bootstrap/3.2.0/js/bootstrap.min.js"></script>
    <br>
    <link rel="stylesheet" href="{{ url_for('static', filename='css/common.css') }}>

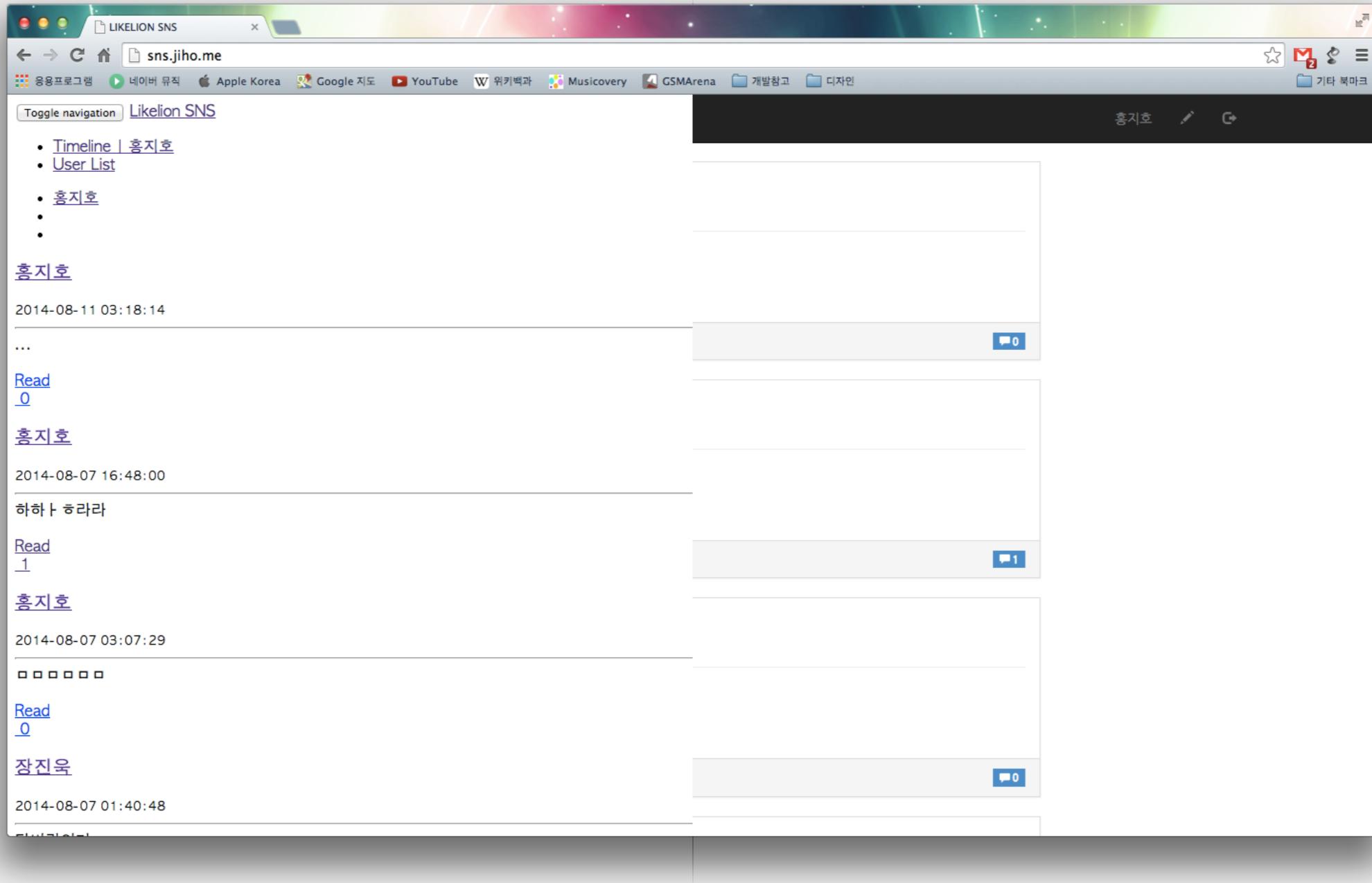
</head>
<body>

    <nav class="navbar navbar-default" role="navigation">
        <div class="container">
            <!-- Brand and toggle get grouped for better mobile display -->
            <div class="navbar-header">
```



CSS

웹 페이지의 표현



CSS

웹 페이지의 표현

The screenshot shows the Developer Tools interface for the URL <http://sns.jiho.me/>. The left pane displays the DOM tree, and the right pane shows the Styles tab of the Elements panel.

DOM Tree (Elements Tab):

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body>
    <nav class="navbar navbar-default navbar-inverse navbar-fixed-top" role="navigation">
      ::before
      <div class="container">
        ::before
        <!-- Brand and toggle get grouped for better mobile display -->
        <div class="navbar-header">...</div>
        <!-- Collect the nav links, forms, and other content for toggling -->
        <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">...</div>
        <!-- /.navbar-collapse -->
        ::after
      </div>
      <!-- /.container -->
      ::after
    </nav>
    <div class="container">...</div>
    <script src="/static/js/timeline.js"></script>
  </body>
</html>
```

Styles Tab:

The Styles tab lists the CSS rules applied to the selected element (`<div class='container'>`). The rules are categorized by source:

- `element.style {` (bootstrap.min.css:5)
- `.container {` (bootstrap.min.css:5)
padding-right: 15px;
padding-left: 15px;
margin-right: auto;
margin-left: auto;
- `* {` (common.css:11)
border-radius: 0px !important;
- `* {` (bootstrap.min.css:5)
-webkit box-sizing: border-box;
-moz box-sizing: border-box;
box-sizing: border-box;
- `div {` (user agent stylesheet)
display: block;
- Pseudo ::before element**
`.clearfix:before, .clearfix:after, .dl-horizontal dd:before, .dl-horizontal dd:after, .container:before,` (bootstrap.min.css:5)

CSS

웹 페이지의 표현

Developer Tools – http://sns.jiho.me/

Elements Network Sources Timeline Profiles Resources Audits Console

Styles Computed Event Listeners »

태그 선택

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body>
    <nav class="navbar navbar-default navbar-inverse navbar-fixed-top" role="navigation">
      ::before
      <div class="container">
        ::before
        <!-- Brand and toggle get grouped for better mobile display -->
        <div class="navbar-header">...</div>
        <!-- Collect the nav links, forms, and other content for toggling -->
        <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">...</div>
        <!-- /.navbar-collapse -->
        ::after
      </div>
      <!-- /.container -->
      ::after
    </nav>
    <div class="container">...</div>
    <script src="/static/js/timeline.js"></script>
  </body>
</html>
```

element.style { }

.container { padding-right: 15px; padding-left: 15px; margin-right: auto; margin-left: auto; } bootstrap.min.css:5

* { border-radius: 0px !important; } common.css:11

* { -webkit_box_sizing: border_box; -moz_box_sizing: border_box; box-sizing: border-box; } bootstrap.min.css:5

div { display: block; } user agent stylesheet

Pseudo ::before element

.clearfix:before, .clearfix:after, .dl-horizontal dd:before, .dl-horizontal dd:after, .container:before, bootstrap.min.css:5

CSS

웹 페이지의 표현

The screenshot shows the Developer Tools interface for the URL <http://sns.jiho.me/>. The left pane displays the DOM tree, and the right pane shows the Styles tab of the Elements panel. A specific CSS rule for the `.container` class is highlighted with a red box, and the text "스타일 적용" (Style Applied) is overlaid on the right side of the highlighted area.

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body>
    <nav class="navbar navbar-default navbar-inverse navbar-fixed-top" role="navigation">
      ::before
      <div class="container">
        ::before
        <!-- Brand and toggle get grouped for better mobile display -->
        <div class="navbar-header">...</div>
        <!-- Collect the nav links, forms, and other content for toggling -->
        <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">...</div>
        <!-- /.navbar-collapse -->
        ::after
      </div>
      <!-- /.container -->
      ::after
    </nav>
    <div class="container">...</div>
    <script src="/static/js/timeline.js"></script>
  </body>
</html>
```

Styles Computed Event Listeners »

element.style { } bootstrap.min.css:5

.container { padding-right: 15px; padding-left: 15px; margin-right: auto; margin-left: auto; }

* { border-radius: 0px !important; }

* { -webkit_box_sizing: border_box; -moz_box_sizing: border_box; box-sizing: border-box; }

div { display: block; }

Pseudo ::before element

.clearfix:before, .clearfix:after, .dl-horizontal dd:before, .dl-horizontal dd:after, .container:before,

스타일 적용

Client-Server

Client

- Web Browser
- HTML, CSS, JavaScript

Server

- Web Server



jQuery

JavaScript : 웹 페이지의 동작



Likelion

jQuery

JavaScript : 웹 페이지의 동작

```
$("_____")._____();
```



Likelion

jQuery

JavaScript : 웹 페이지의 동작

```
$("CSS Selector").                ();
```



jQuery

JavaScript : 웹 페이지의 동작

```
$("_____").Function( );
```



<http://api.jquery.com/>



jQuery

JavaScript : 웹 페이지의 동작

```
$("_____").text();  
$("_____").text("some text");  
$("_____").html();  
$("_____").html("<div>some html code</div>");  
$("_____").val();  
$("_____").val("7");
```



jQuery

JavaScript : 웹 페이지의 동작

```
$("_____").empty();  
$("_____").remove();  
$("_____").append("<div>some html code</div>");
```



jQuery > Event Handler

JavaScript : 웹 페이지의 동작

- Event Handler : 이벤트를 처리하는 **함수**
- 이벤트 : 클릭, 더블클릭, 키 입력, ...
 - <http://api.jquery.com/category/events/>



jQuery > Event Handler

JavaScript : 웹 페이지의 동작

```
$(<b>document</b>).ready(<b>function()</b>{  
});
```



jQuery > Event Handler

JavaScript : 웹 페이지의 동작

```
$(document).ready(function(){  
    대상  
});
```



jQuery > Event Handler

JavaScript : 웹 페이지의 동작

```
$(<b>document</b>).ready(function(){  
});
```

이벤트



jQuery > Event Handler

JavaScript : 웹 페이지의 동작

```
$document).ready(function(){  
});
```

핸들러



jQuery > Event Handler

JavaScript : 웹 페이지의 동작

```
$(document).ready(function(){  
    문서가  
});
```



jQuery > Event Handler

JavaScript : 웹 페이지의 동작

```
$document).ready(function(){  
    준비되면  
});
```



jQuery > Event Handler

JavaScript : 웹 페이지의 동작

```
$document).ready(function(){  
});
```

이 함수를 실행해라



jQuery > Event Handler

JavaScript : 웹 페이지의 동작

```
$(_document).ready(function(){  
    문서가           준비되면      이 함수를 실행해라  
});
```



jQuery > Event Handler

JavaScript : 웹 페이지의 동작

```
$(document).ready(function(){

    $("button.btn-delete").click(function(){
        // do something
    });

    $("form").submit(function(){
        //do something

        return false;
    });

});
```



jQuery > Event Handler

JavaScript : 웹 페이지의 동작

```
$(document).ready(function(){

    $("button.btn-delete").click(function(){
        // do something
    });

    $("form").submit(function(){
        //do something

        return false;
    });
});
```



jQuery > Event Handler

JavaScript : 웹 페이지의 동작

```
$(document).ready(function(){
    $("button.btn-delete").click(function(){
        // do something
    });
    $("form").submit(function(){
        //do something
        return false;
    });
});
```

class 가 btn-delete 인
button 태그를 클릭하면
이 함수를 실행해라



jQuery > Event Handler

JavaScript : 웹 페이지의 동작

```
$(document).ready(function(){

    $("button.btn-delete").click(function(){
        // do something
    });

    $("form").submit(function(){
        //do something
        return false;
    });
});
```

form 을 전송하면
이 함수를 실행해라



Client-Server

Client

- Web Browser
- HTML, CSS, JavaScript

Server

- Web Server



...



Likelion

Server-Side Back-end



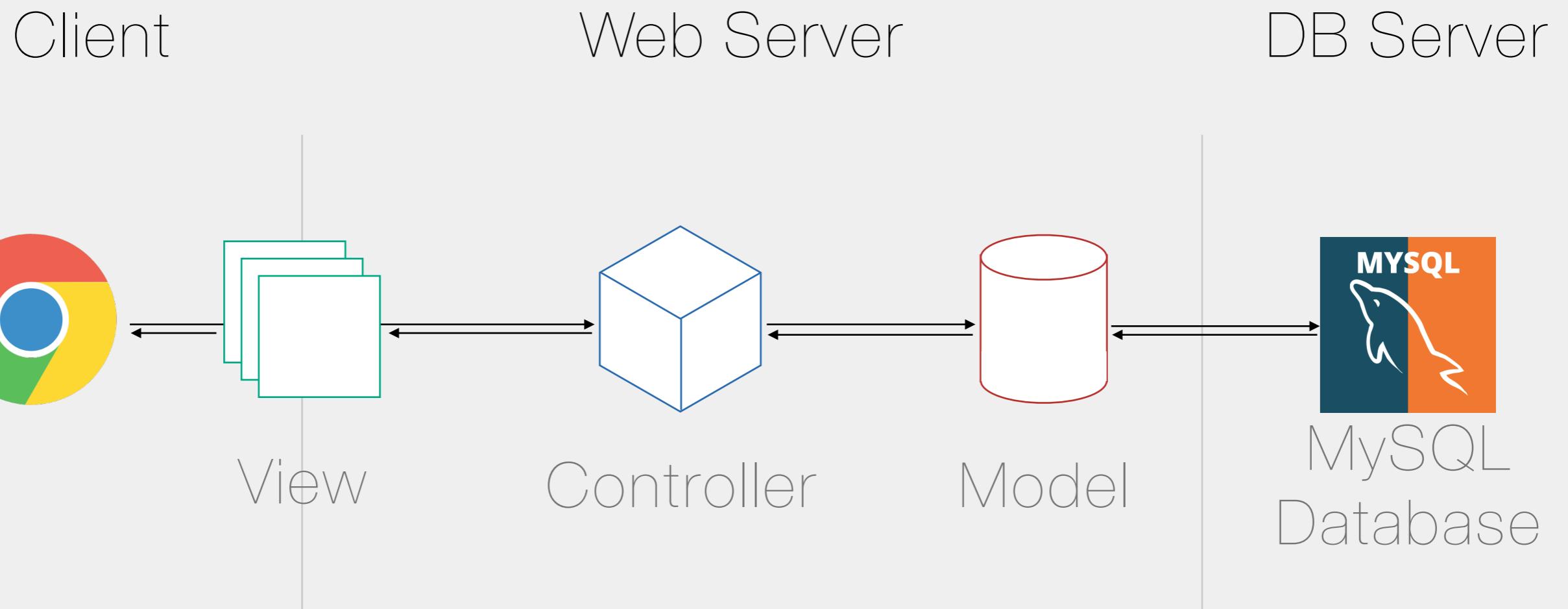
Python

Server-Side Script

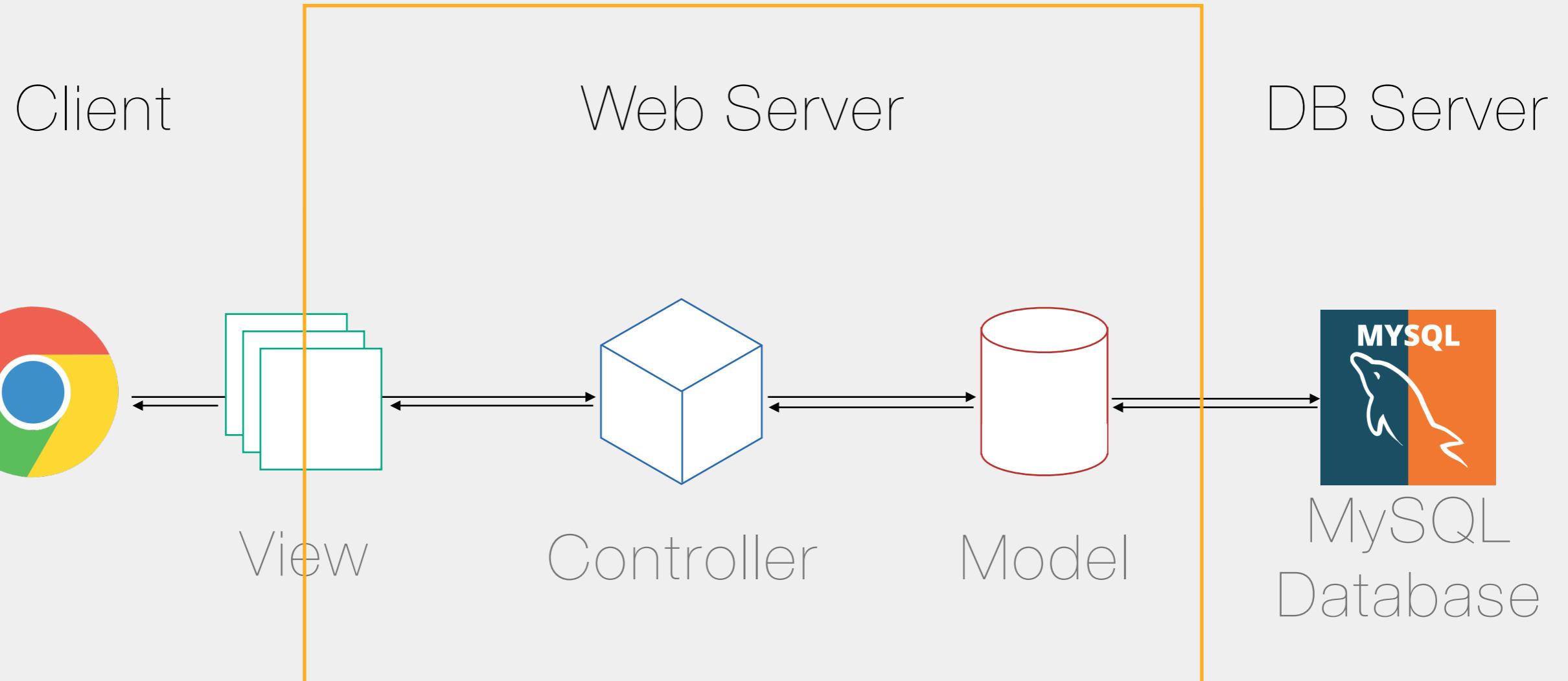


Likelion

Web Service Structure



Web Service Structure



Python : Programming Language



Programming Language?



Likelion

Programming

생각의 표현



Likelion

Programming

생각의 표현

Programming?



Likelion

Programming

생각의 표현

할 일을 순서대로 나열한 후



Likelion

Programming

생각의 표현

할 일을 순서대로 나열한 후

프로그래밍 언어로 표현



Likelion

Programming

생각의 표현

컴퓨터는 한 번에 하나의 일만 할 수 있다



Likelion

Programming

생각의 표현

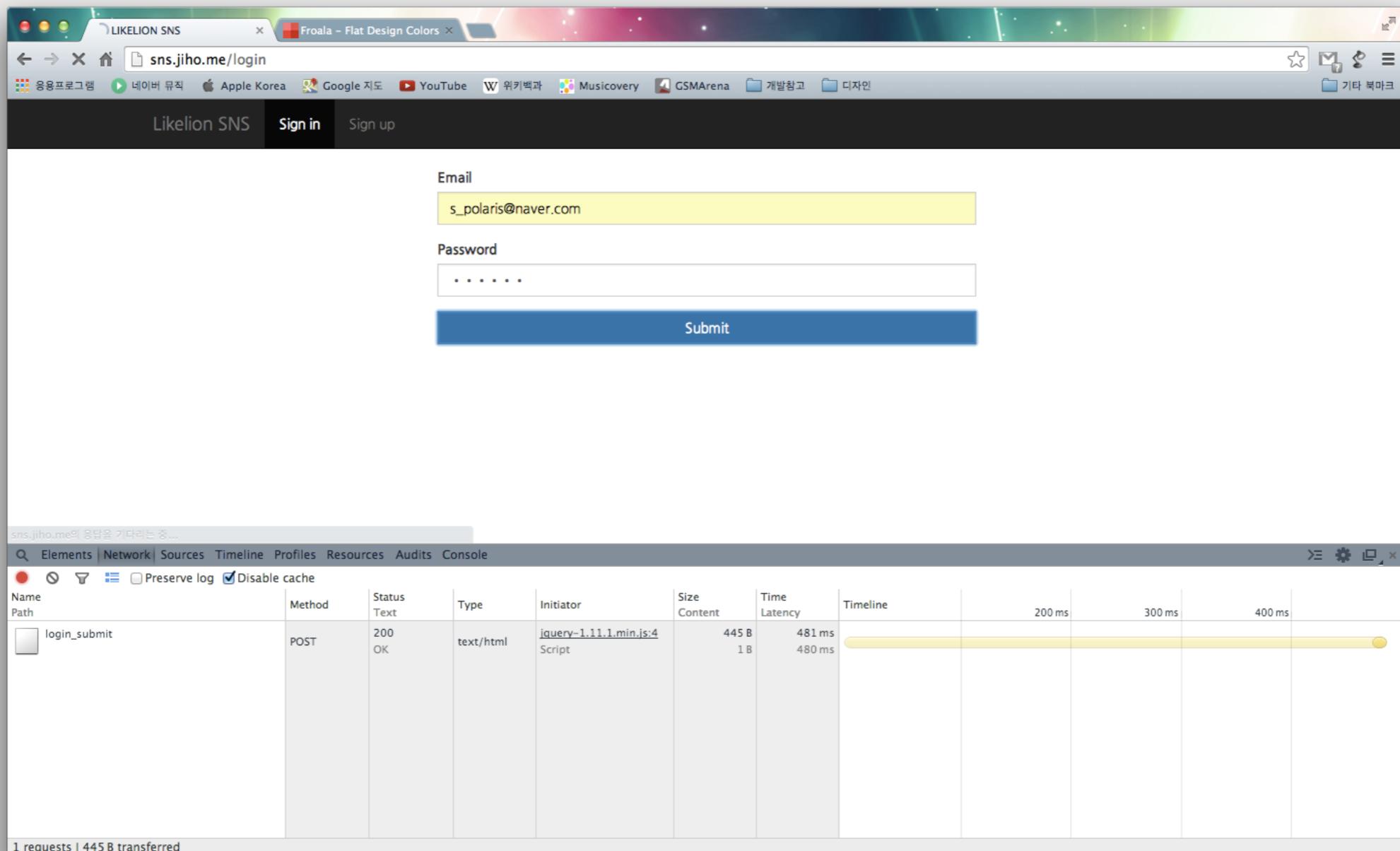
예시 : 로그인



Likelion

Programming

생각의 표현



Likelion

Programming

생각의 표현

```
auth.py ○
20 @app.route('/login_submit', methods = ['POST'])
21 def login_submit():
22     email = request.form['email']
23     password = request.form['password']
24
25     user = login_check(email, password)
26     if user.count() == 1:
27         # login success
28         user = user.one()
29
30         session['username'] = user.username
31         session['user_id'] = user.id
32         return '0'
33     else:
34         return '1'

user_manager.py ○
24 def login_check(email, password):
25     return User.query.filter(User.email == email,
26                             User.password == db.func.md5(password))
27
28
29
30
31
32
33
34
35
```



Client-Server

Client

- Web Browser
- HTML, CSS, JavaScript

Server

- Web Server
- Python



HTML Form



HTML Form

- 사용자가 입력한 데이터를 서버로 보낼 때 사용



HTML Form

```
<form role="form" method="post" action="{{ url_for('login_submit') }}>

<div class="form-group">
    <label for="input-email">Email</label>
    <input type="email" id="input-email"
        name="email" class="form-control" placeholder="Email" autofocus
        required>
</div>
<div class="form-group">
    <label for="input-password">Password</label>
    <input type="password" class="form-control" id="input-password"
        name="password" placeholder="Password" required>
</div>

<button type="submit" class="btn btn-primary btn-block">Submit</button>

</form>
```



HTML Form

```
<form role="form" method="post" action="{{ url_for('login_submit') }}>

<div class="form-group">
    <label for="input-email">Email</label>
    <input type="email" id="input-email"
        name="email" class="form-control" placeholder="Email" autofocus
        required>
</div>
<div class="form-group">
    <label for="input-password">Password</label>
    <input type="password" class="form-control" id="input-password"
        name="password" placeholder="Password" required>
</div>

<button type="submit" class="btn btn-primary btn-block">Submit</button>

</form>
```



HTML Form

```
<form role="form" method="post" action="{{ url_for('login_submit') }}>

<div class="form-group">
    <label for="input-email">Email</label>
    <input type="email" id="input-email"
        name="email" class="form-control" placeholder="Email" autofocus
        required>
</div>
<div class="form-group">
    <label for="input-password">Password</label>
    <input type="password" class="form-control" id="input-password"
        name="password" placeholder="Password" required>
</div>

<button type="submit" class="btn btn-primary btn-block">Submit</button>

</form>
```



HTML Form

```
<form role="form" method="post" action="{{ url_for('login_submit') }}>

<div class="form-group">
    <label for="input-email">Email</label>
    <input type="email" id="input-email"
        name="email" class="form-control" placeholder="Email" autofocus
        required>
</div>
<div class="form-group">
    <label for="input-password">Password</label>
    <input type="password" class="form-control" id="input-password"
        name="password" placeholder="Password" required>
</div>

<button type="submit" class="btn btn-primary btn-block">Submit</button>

</form>
```



HTML Form

```
<form role="form" method="post" action="{{ url_for('login_submit') }}>

<div class="form-group">
    <label for="input-email">Email</label>
    <input type="email" id="input-email"
        name="email" class="form-control" placeholder="Email" autofocus
        required>
</div>
<div class="form-group">
    <label for="input-password">Password</label>
    <input type="password" class="form-control" id="input-password"
        name="password" placeholder="Password" required>
</div>

<button type="submit" class="btn btn-primary btn-block">Submit</button>

</form>
```



HTML Form

```
<form role="form" method="post" action="{{ url_for('login_submit') }}>

<div class="form-group">
    <label for="input-email">Email</label>
    <input type="email" id="input-email"
        name="email" class="form-control" placeholder="Email" autofocus
        required>
</div>
<div class="form-group">
    <label for="input-pass">Password</label>
    <input type="password" id="input-password"
        name="password" placeholder="Password" required>
</div>

<button type="submit" class="btn btn-primary" value="Submit">Submit</button>
</form>
```

```
@app.route('/login_submit', methods = ['POST'])
def login_submit():
    email = request.form['email']
    password = request.form['password']

    user = login_check(email, password)
    if user:
        # login success
        user = user.first()
        session['username'] = user.username
        session['user_id'] = user.id
        return '0'
    else:
        return '1'
```



HTML Form

```
<form role="form" method="post" action="{{ url_for('login_submit') }}>

<div class="form-group">
    <label for="input-email">Email</label>
    <input type="email" id="input-email"
        name="email" class="form-control" placeholder="Email" autofocus
        required>
</div>
<div class="form-group">
    <label for="input-pass">Password</label>
    <input type="password" id="input-pass"
        name="password" placeholder="Password" required>
</div>

<button type="submit" class="btn btn-primary">Submit</button>
</form>
```

```
@app.route('/login_submit', methods = ['POST'])
def login_submit():
    email = request.form['email']
    password = request.form['password']

    user = login_check(email, password)
    if user:
        # login success
        user = user.first()
        session['username'] = user.username
        session['user_id'] = user.id
        return '0'
    else:
        return '1'
```



Session

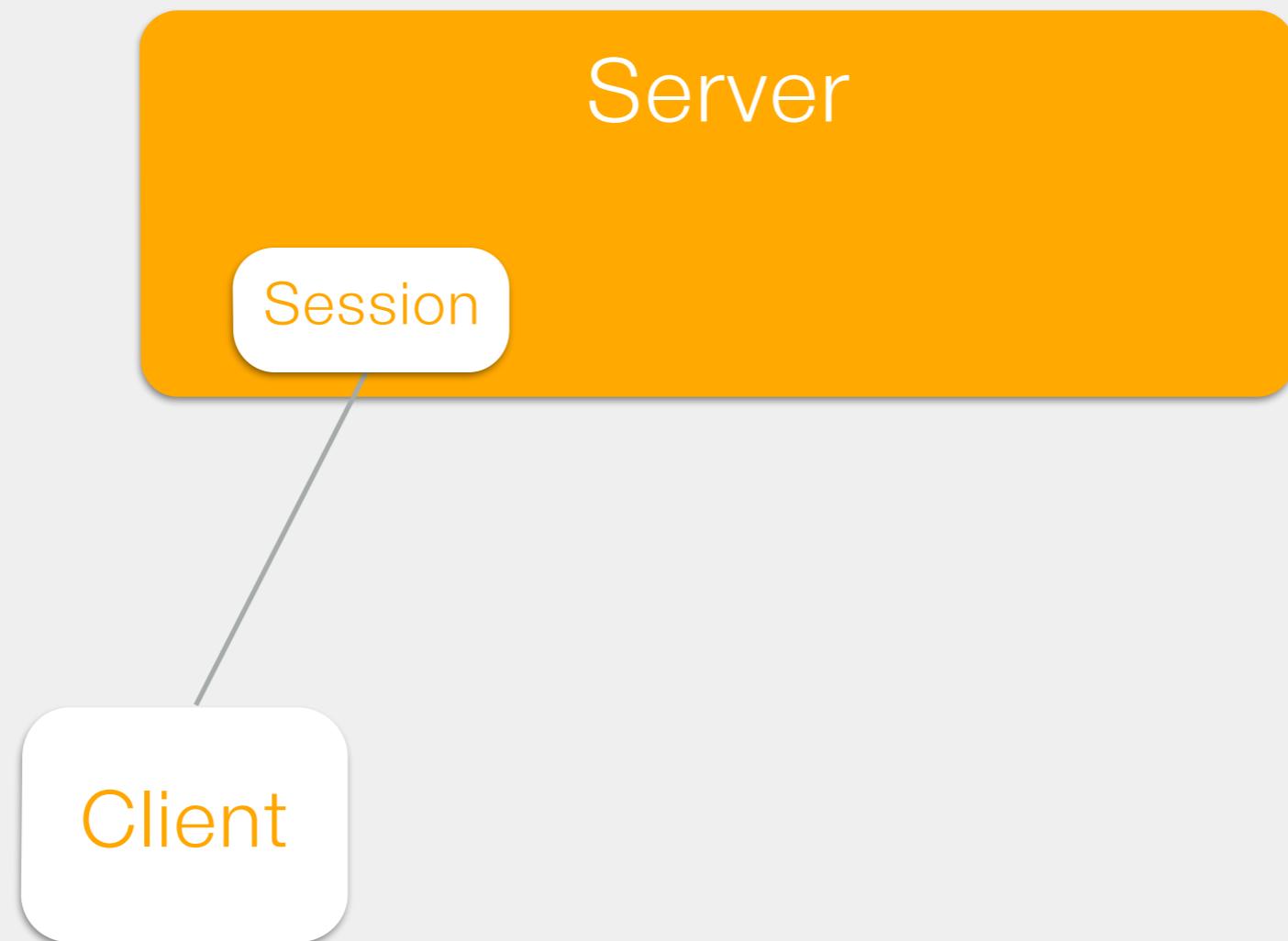


Session

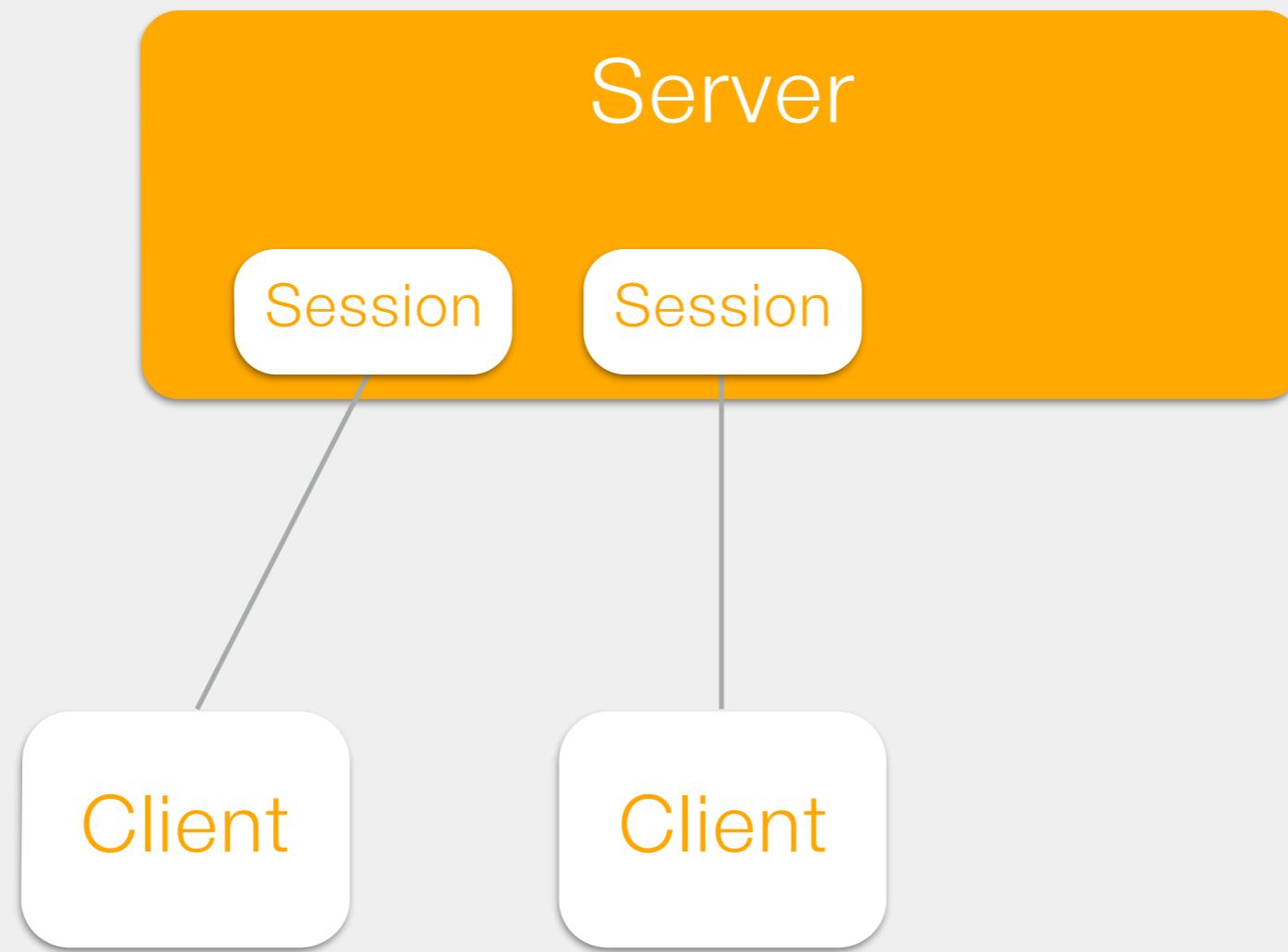
Server



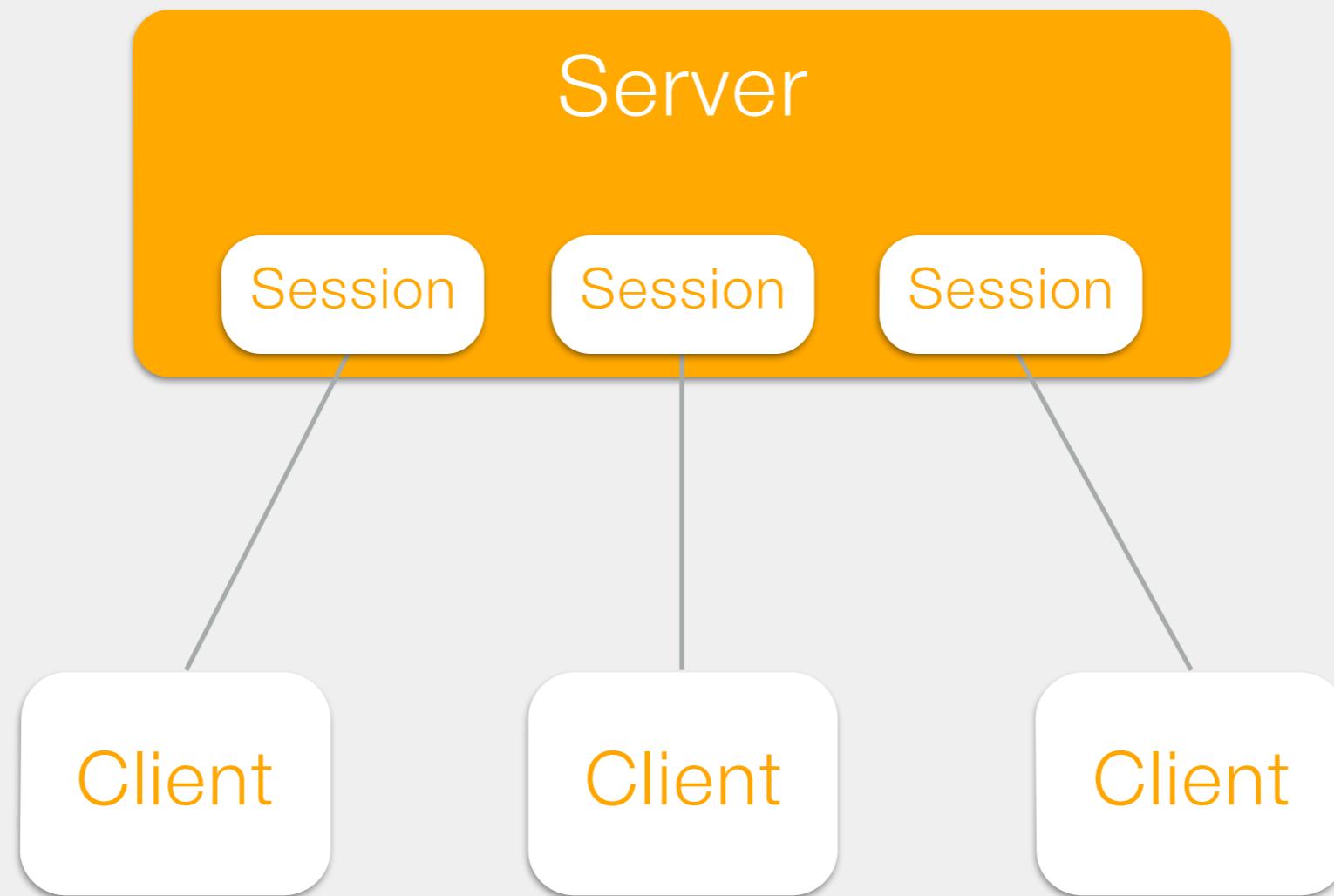
Session



Session



Session



Session

예시) 로그인

```
@app.route('/login_submit', methods = ['POST'])
def login_submit():
    email = request.form['email']
    password = request.form['password']

    user = login_check(email, password)
    if user:
        # login success
        user = user.first()
        session['username'] = user.username
        session['user_id'] = user.id
        return '0'
    else:
        return '1'
```



Session

예시) 담벼락

```
@app.route('/', defaults={'wall_id':0})
@app.route('/timeline', defaults={'wall_id':0})
@app.route('/timeline/<int:wall_id>')
def timeline(wall_id):
    if not is_login():
        return redirect(url_for('login'))
    if wall_id == 0:
        wall_id = session['user_id']

    session['wall_id'] = wall_id
    session['wall_username'] = get_user(wall_id).username

    context = {
        'posts':get_wall_posts(wall_id,0),
        'BS':BS
    }
    postList = render_template('post_list.html',context=context)
    context = {
        'post_list':postList
    }
    return render_template('timeline.html', context=context)
```



AJAX

Asynchronous JavaScript and XML



Likelion

AJAX

The screenshot shows a web browser window for 'LIKELION SNS' displaying a post by '홍지호'. The browser's address bar shows the URL `sns.jiho.me/read/38`. Below the address bar is a toolbar with various icons. The main content area displays the post details and a list of comments. At the bottom of the page, the browser's developer tools are open, specifically the Network tab. This tab shows a timeline of requests made to the server. The requests listed are:

Name	Path	Method	Status	Type	Initiator	Size Content	Time Latency	Timeline
38	/read	GET	200 OK	text/html	Other	1.7 KB 4.1 KB	438 ms 437 ms	400 ms 600 ms 800 ms 1.00 s 1.20 s 1.40 s 1.60 s
bootstrap.min.css	/static/css	GET	200 OK	text/css	Parser	18.0 KB 107 KB	298 ms 290 ms	400 ms 600 ms 800 ms 1.00 s 1.20 s 1.40 s 1.60 s
common.css	/static/css	GET	200 OK	text/css	Parser	471 B 181 B	264 ms 262 ms	400 ms 600 ms 800 ms 1.00 s 1.20 s 1.40 s 1.60 s
jquery-1.11.1.min.js	/static/js	GET	200 OK	application/...	Parser	32.8 KB 93.5 KB	413 ms 272 ms	400 ms 600 ms 800 ms 1.00 s 1.20 s 1.40 s 1.60 s
bootstrap.min.js	/static/js	GET	200 OK	application/...	Parser	8.6 KB 31.1 KB	267 ms 265 ms	400 ms 600 ms 800 ms 1.00 s 1.20 s 1.40 s 1.60 s
common.js		GET	200 OK	application/...	Parser	1.4 KB	270 ms	400 ms 600 ms 800 ms 1.00 s 1.20 s 1.40 s 1.60 s

At the bottom of the timeline, it says '17 requests | 1.5 MB transferred | 1.59 s (load: 1.62 s, DOMContentLoaded: 1.36 s)'



Likelion

AJAX

The screenshot shows a web browser window for 'LIKELION SNS' displaying a post by '홍지호'. The post content is a series of short, random strings: 'asdf'sdaf', 'asd', 'f', 'asdf', 'as', 'efg', 'awegaweg', and 'awegawew'. Below the content are two buttons: '수정' (Edit) and '삭제' (Delete). A yellow callout bubble points from the bottom left towards the browser window, containing the Korean text '페이지 이동 시 보내는 요청' (Request sent when navigating to a page).

페이지 이동 시 보내는 요청

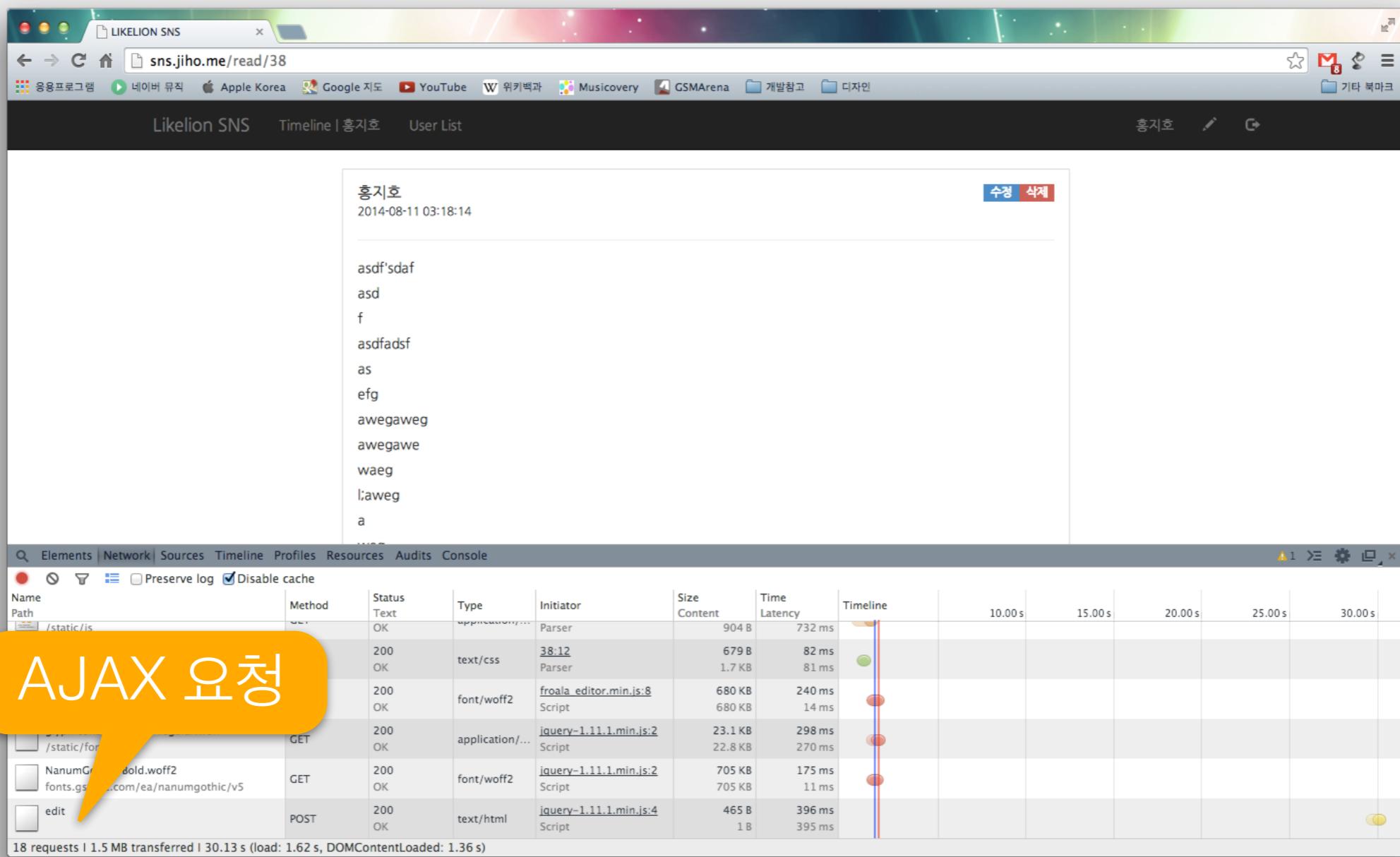
Name	Path	Method	Status	Type	Initiator	Size Content	Time Latency	Timeline
	38 /read	GET	200 OK	text/html	Other	1.7 KB 4.1 KB	438 ms 437 ms	400 ms 600 ms 800 ms 1.00 s 1.20 s 1.40 s 1.60 s
	bootstrap.min.css /static/css	GET	200 OK	text/css	Parser	18.0 KB 107 KB	298 ms 290 ms	
	common.css /static/css	GET	200 OK	text/css	Parser	471 B 181 B	264 ms 262 ms	
	jquery-1.11.1.min.js /static/js	GET	200 OK	application/...	Parser	32.8 KB 93.5 KB	413 ms 272 ms	
	bootstrap.min.js /static/js	GET	200 OK	application/...	Parser	8.6 KB 31.1 KB	267 ms 265 ms	
	common.js	GET	200 OK	...	38:14	1.4 KB	270 ms	

17 requests | 1.5 MB transferred | 1.59 s (load: 1.62 s, DOMContentLoaded: 1.36 s)



Likelion

AJAX



AJAX 요청



Likelion

AJAX

The screenshot shows the Network tab of the Chrome DevTools developer tools. A specific network request is selected, revealing its Headers, Preview, Response, Cookies, and Timing details.

Selected Request: /static/js

Headers:

- Pragma: no-cache
- Referer: http://sns.jiho.me/read/38
- User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/67.0.3396.87 Safari/537.36
- X-Requested-With: XMLHttpRequest

Form Data:

- post_id: 38
- body: <div contenteditable="true" class="fr-ds fr-fil fr-fil-100" style="border: 1px solid #ccc; padding: 5px; width: 100%; height: 100px; margin-bottom: 10px;">

asdfadsf

as

efg

</div><button type="button" class="fr-btn fr-btn-primary fr-btn-large fr-btn-block">edit

Response Headers:

- Alternate-Protocol: 80:quic,80:quic
- Cache-Control: private
- Content-Encoding: gzip



AJAX

The screenshot shows the Chrome DevTools Network tab. The left sidebar lists resources by name and path. The right panel shows a table with one row, indicating 1 request and 0 responses.

		Headers	Preview	Response	Cookies	Timing
1	0					

Resources Listed:

- /static/js
- nanumgothic.css
fonts.googleapis.com/early/
- NanumGothic-Regular.w...
- fonts.gstatic.com/ea/nanu
- glyphicon-halflings-re...
- /static/fonts
- NanumGothic-Bold.woff2
- fonts.gstatic.com/ea/nanu
- edit



AJAX

예시 : 글 수정

The screenshot shows a code editor interface with two main panes. On the left, there's a sidebar with a tree view of project files. On the right, there are two code editors: one for JavaScript and one for Python.

JavaScript Editor (edit.js):

```
1 $(document).ready(function(){
2     $(document).on('click','#btn-edit-post-submit',
3         function(){
4             var body = $("#post-body").html();
5             var postId = $("#input-post-id").val();
6
7             show_progress();
8             $.ajax({
9                 url:'/edit',
10                data:{'post_id':postId,'body':body},
11                type:'post',
12                success:function(response){
13                    if (response=='0') {
14                        show_toast('수정되었습니다','success');
15                    } else if (response=='1') {
16                        show_toast('수정 권한이 없습니다','error');
17                    } else {
18                        show_toast('오류가 발생하였습니다. 잠시 후에 다시
19                            시도해주세요','error');
20                    },
21                    error:function(){
22                        show_toast('오류가 발생하였습니다. 잠시 후에 다시
23                            시도해주세요','error');
24                    },
25                    complete:function(){
26                        hide_progress();
27                    }
28                });
29            });
30        });
31    });


```

Python Editor (post.py):

```
46 @app.route('/edit',methods=['POST'])
47 def edit():
48     targetPost = get_post(request.form['post_id'])
49     if targetPost.user_id != session['user_id']:
50         return '1'
51
52     edit_post(request.form)
53     return '0'


```



AJAX

예시 : Autoload

The screenshot shows a code editor with two main panes. On the left, the file structure is visible under 'GROUP 1' and 'GROUP 2'. In the center, there are two tabs: 'timeline.js' and 'index.py'. The 'timeline.js' tab contains the following JavaScript code:

```
//autoload
var last=false;
var running=false;

$(document).ready(function(){
    //get_posts();
});

$(document).on('scroll',function(){
    if (last) return false;

    var lastPost = $('#post-list>div.panel:last-child');
    var lastPostId = lastPost.data('postId');

    if ($(body).scrollTop() + $(window).height() > lastPost.offset().top) {
        get_posts(lastPostId);
    }
});

function get_posts(start_id) {
    if (!running) {
        running = true;

        show_progress();
        $.post('/get_posts',{'start_id':start_id},function(response){
            if (response.trim()=='') {
                last=true;
                return false;
            }
            $('#post-list').append(response);
        }).always(function(){
            hide_progress();
            running = false;
        });
    }
}
```

The 'index.py' tab contains the following Python code:

```
@app.route('/get_posts', methods=['POST'])
def get_posts():
    posts = get_wall_posts(session['wall_id'], request.form['start_id'])

    if posts.count() == 0:
        return ''

    context = {
        'posts':posts,
        'BS':BS
    }

    return render_template('post_list.html',context=context)
```



JSON

JavaScript Object Notation

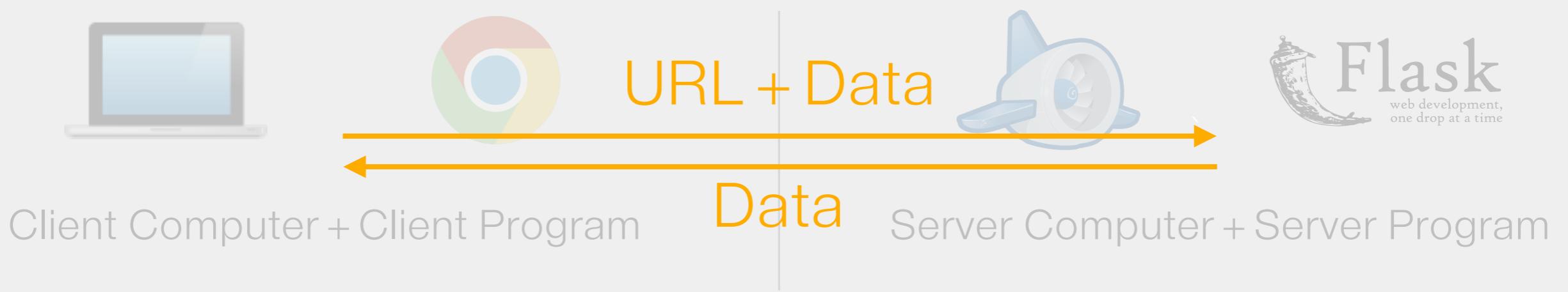


Likelion

JSON

Web Client

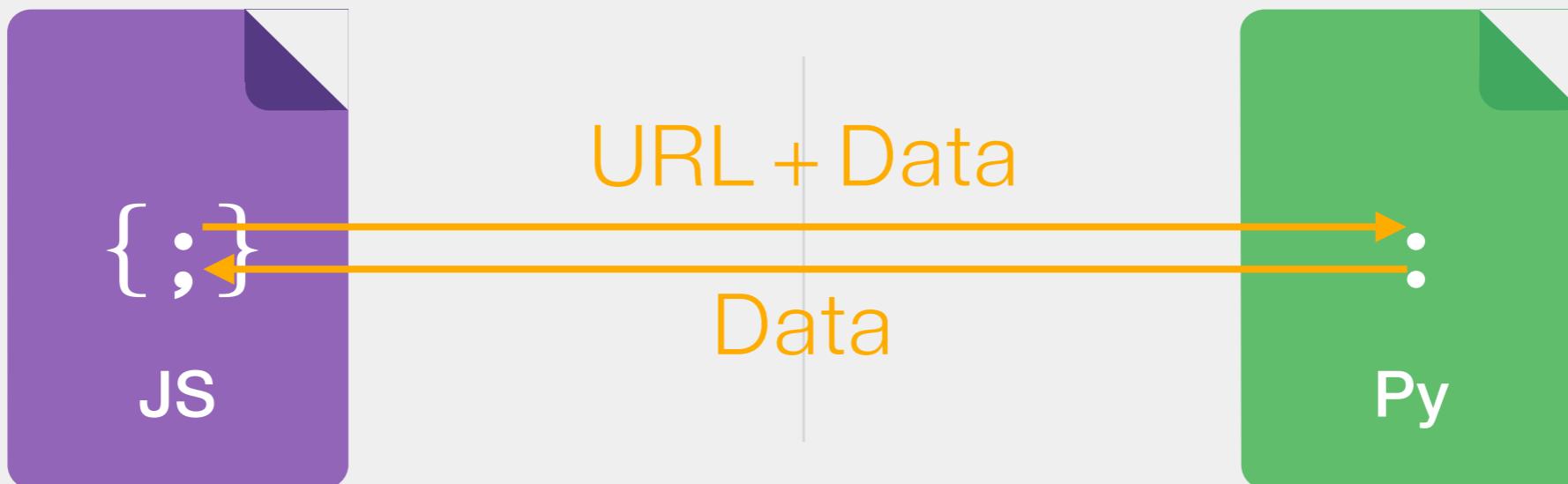
Web Server



JSON

Web Client

Web Server



JSON

```
1 // JavaScript – Object
2
3 var jiho = {
4     name:'Jiho Hong',
5     email:'jiho.hong@hotmail.com',
6     phone:'010-4573-6906'
7 };
8
```

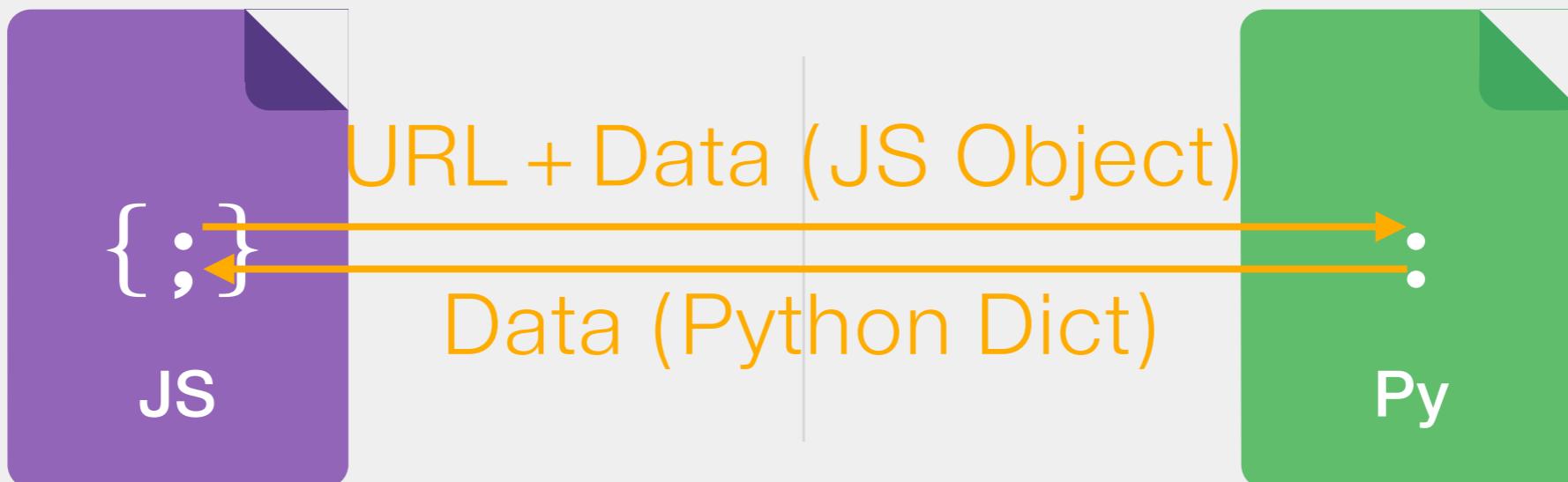
```
1 # Python – Dict
2
3 jiho = {
4     'name':'Jiho Hong',
5     'email':'jiho.hong@hotmail.com',
6     'phone':'010-4573-6906'
7 }
8
```



JSON

Web Client

Web Server



JSON

- JavaScript Object Notation
- 서로 다른 프로그래밍 언어 사이에서 데이터를 주고받기 위해 사용되는 데이터 표현 방식 중 하나
- 자바스크립트에서 object를 표현하는 문법을 그대로 문자열로 바꾼 것



JSON

Request



JSON

Response



...



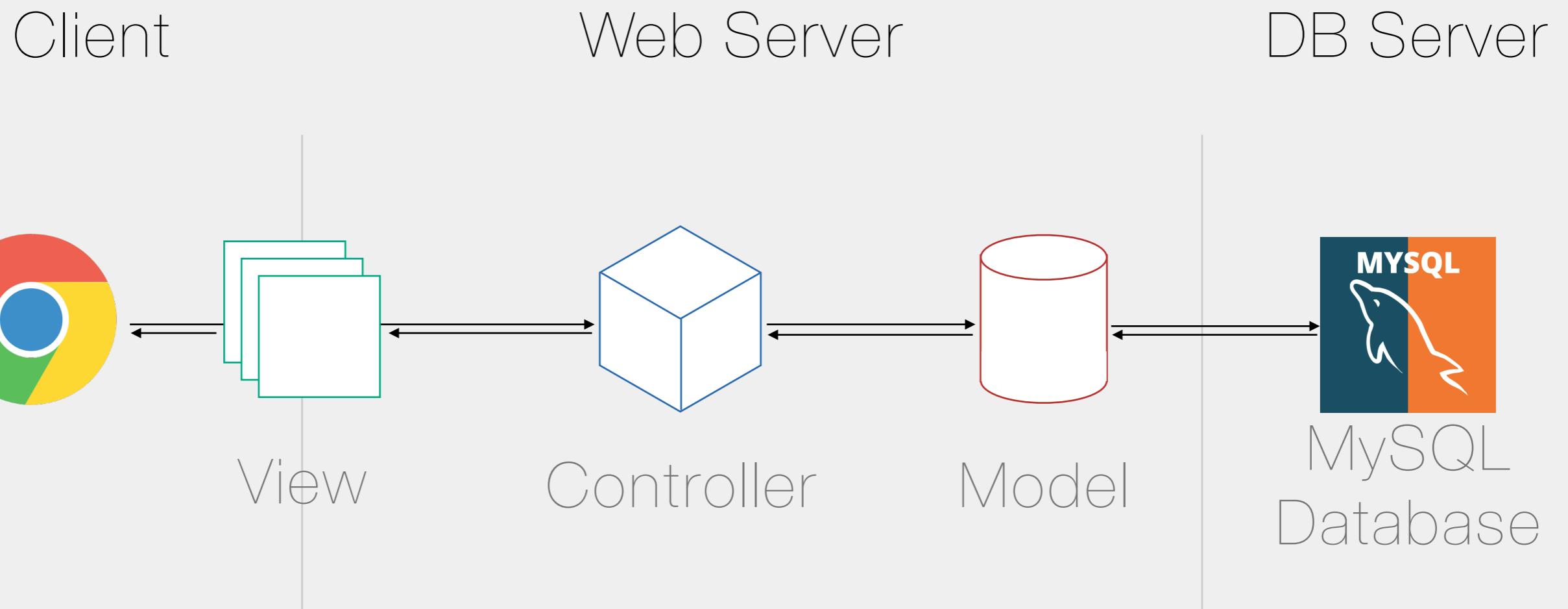
Likelion

MVC

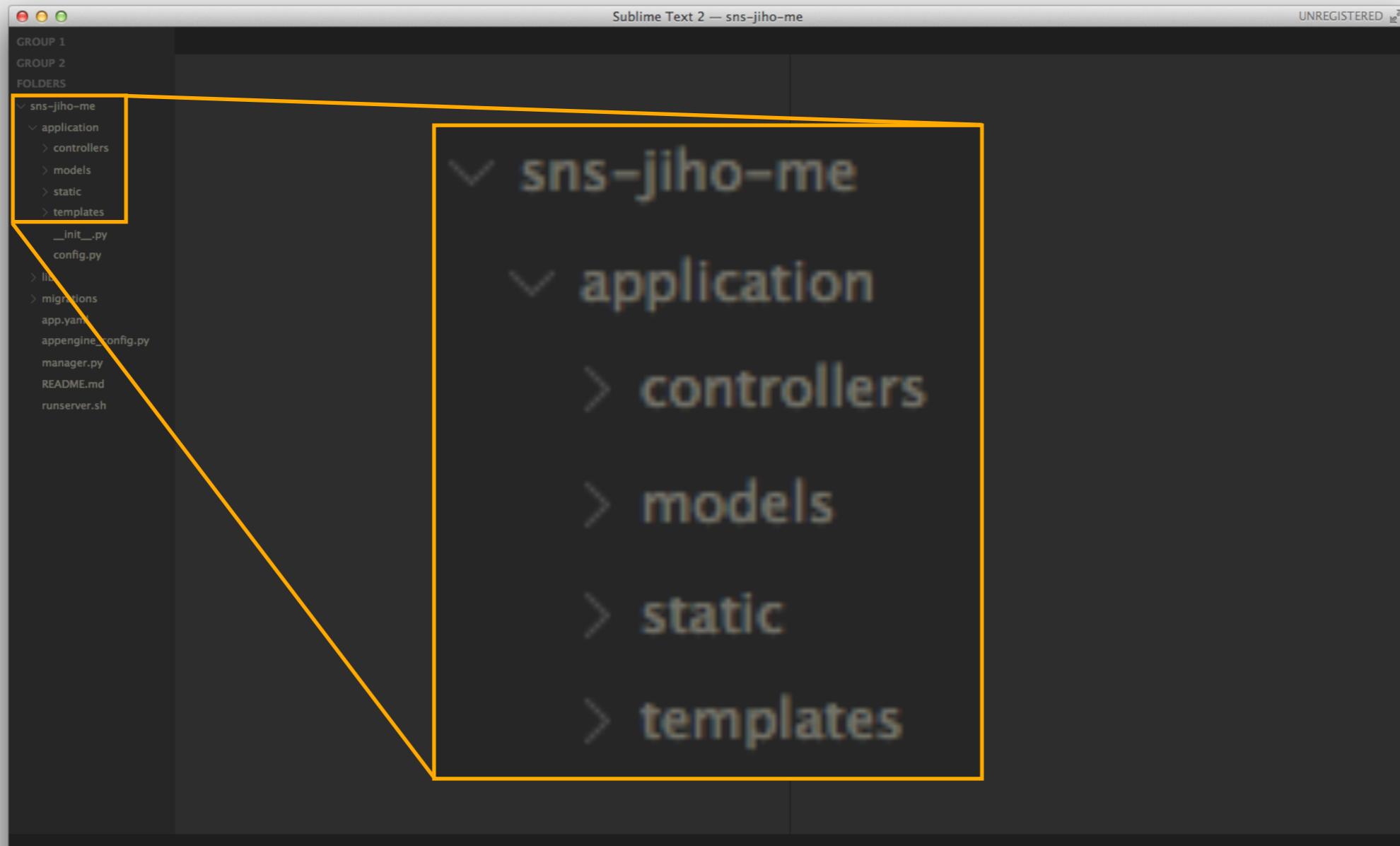


Likelion

Web Service Structure



MVC



MVC

```
✓ sns-jiho-me
  ✓ application
    > controllers
    > models
    > static
    > templates
```



MVC > View

- 사용자의 요청에 대한 결과를 최종적으로 보여주는 역할
- Jinja : 템플릿 HTML 파일 + render_template()



MVC > View

The image shows a code editor interface with two main panes. On the left, the file `read.html` is displayed, which is a Jinja2 template. On the right, the file `post.py` is displayed, which is a Python script with Flask routes and logic.

read.html (Jinja2 Template)

```
1  {% extends 'layout.html' %}\n2\n3  {% block head %}\n4  {{ super() }}\n5  <script src="{{ url_for('static', filename='js/read.js') }}></script>\n6  <link rel="stylesheet" href="{{ url_for('static',filename='css/read.css') }}>\n7\n8\n9  {% if context.is_author %}\n10 <link href="{{ url_for('static', filename='editor/css/font-awesome.min.css') }}" rel="stylesheet" type="text/css">\n11 <link href="{{ url_for('static',filename='editor/css/froala_editor.min.css') }}" rel="stylesheet" type="text/css">\n12 <link href="{{ url_for('static',filename='editor/css/themes/dark.min.css') }}" rel="stylesheet" type="text/css">\n13 <script src="{{ url_for('static', filename='editor/js/froala_editor.min.js') }}></script>\n14 <script src="{{ url_for('static', filename='js/edit.js') }}></script>\n15\n16\n17\n18\n19\n20\n21\n22\n23\n24\n25\n26\n27\n28\n29\n30\n31\n32\n33\n34\n35\n36\n37\n38\n39\n40\n41\n42\n43\n44\n45\n46\n47\n48
```

post.py (Python Script)

```
10  @app.route('/read/<int:post_id>')\n11  def read(post_id):\n12      if not is_login():\n13          return redirect(url_for('login'))\n14\n15      post = get_post(post_id)\n16      context = {\n17          'post':post,\n18          'is_author':post.user_id == session['user_id']\n19      }\n20      return render_template('read.html', context = context)\n21\n22\n23  @app.route('/write')\n24  def write():\n25      if not is_login():\n26          return redirect(url_for('login'))\n27\n28      return render_template('write.html')\n29\n30  @app.route('/post_submit', methods = ['POST'])\n31  def post_submit():\n32      if not is_login():\n33          return redirect(url_for('login'))\n34\n35      postData = {\n36          'user_id' : session['user_id'],\n37          'wall_id' : session['wall_id'],\n38          'is_secret' : '1' if 'is_secret' in request.form\n39              else '0',\n40          'body' : request.form['body']\n41      }\n42      add_post(postData)\n43\n44      return redirect(url_for('timeline',wall_id = session['\n45          wall_id']))\n46\n47  @app.route('/edit',methods=['POST'])\n48  def edit():\n49      targetPost = get_post(request.form['post_id'])
```

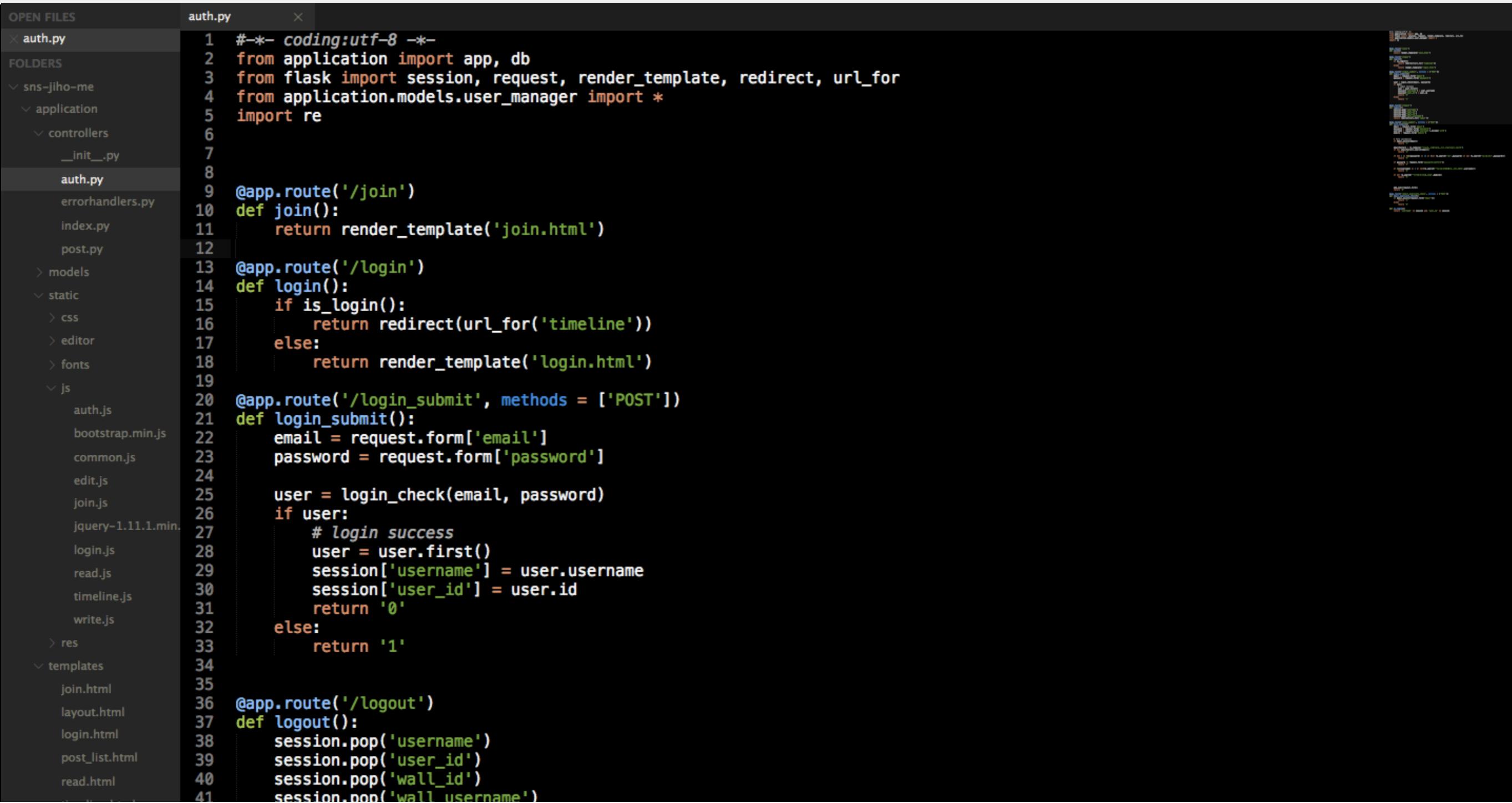


MVC > Controller

- 사용자 요청 분석
- 사용자가 제공한 데이터를 가공해 모델에게 전달
- 사용자가 요청한 데이터를 모델에서 가져와서 뷰로 전달



MVC > Controller



The screenshot shows a code editor interface with a dark theme. On the left is a sidebar titled "OPEN FILES" containing a tree view of project files. The file "auth.py" is currently selected and open in the main editor area.

```
#-*- coding:utf-8 -*-
from application import app, db
from flask import session, request, render_template, redirect, url_for
from application.models.user_manager import *
import re

@app.route('/join')
def join():
    return render_template('join.html')

@app.route('/login')
def login():
    if is_login():
        return redirect(url_for('timeline'))
    else:
        return render_template('login.html')

@app.route('/login_submit', methods = ['POST'])
def login_submit():
    email = request.form['email']
    password = request.form['password']

    user = login_check(email, password)
    if user:
        # login success
        user = user.first()
        session['username'] = user.username
        session['user_id'] = user.id
        return '0'
    else:
        return '1'

@app.route('/logout')
def logout():
    session.pop('username')
    session.pop('user_id')
    session.pop('wall_id')
    session.pop('wall_username')
```



MVC > Model

- 데이터베이스 CRUD
 - Create
 - Read
 - Update
 - Delete
- 컨트롤러는 모델을 통해서 데이터베이스에 접근



MVC > Model



The screenshot shows a code editor interface with a dark theme. On the left, there's a sidebar titled "OPEN FILES" listing various Python files and their paths. The file "post_manager.py" is currently selected and open in the main editor area. The code itself is a Python class definition for managing posts.

```
#-*- coding:utf-8 -*-
from schema import Post
from application import db

def add_post(data):
    post = Post(
        wall_id = data['wall_id'],
        user_id = data['user_id'],
        body = data['body'],
        is_secret = data['is_secret']
    )
    db.session.add(post)
    db.session.commit()
    return post

def get_post(id):
    return Post.query.get(id)

def get_all_posts():
    return Post.query.order_by(desc(Post.edited_time)).all()

def get_wall_posts(wall_id,start_id = 0,
                   post_count = 5):
    start_id = int(start_id)
    if start_id == 0:
        return Post.query.filter(Post.wall_id == wall_id).order_by(db.desc(Post.edited_time)).limit(post_count)
    else:
        return Post.query.filter(Post.wall_id == wall_id, Post.id < start_id).order_by(db.desc(Post.edited_time)).limit(
            post_count)

def delete_post(id):
    db.session.delete(get_post(id))
    db.session.commit()

def edit_post(data):
```



Database - ORM



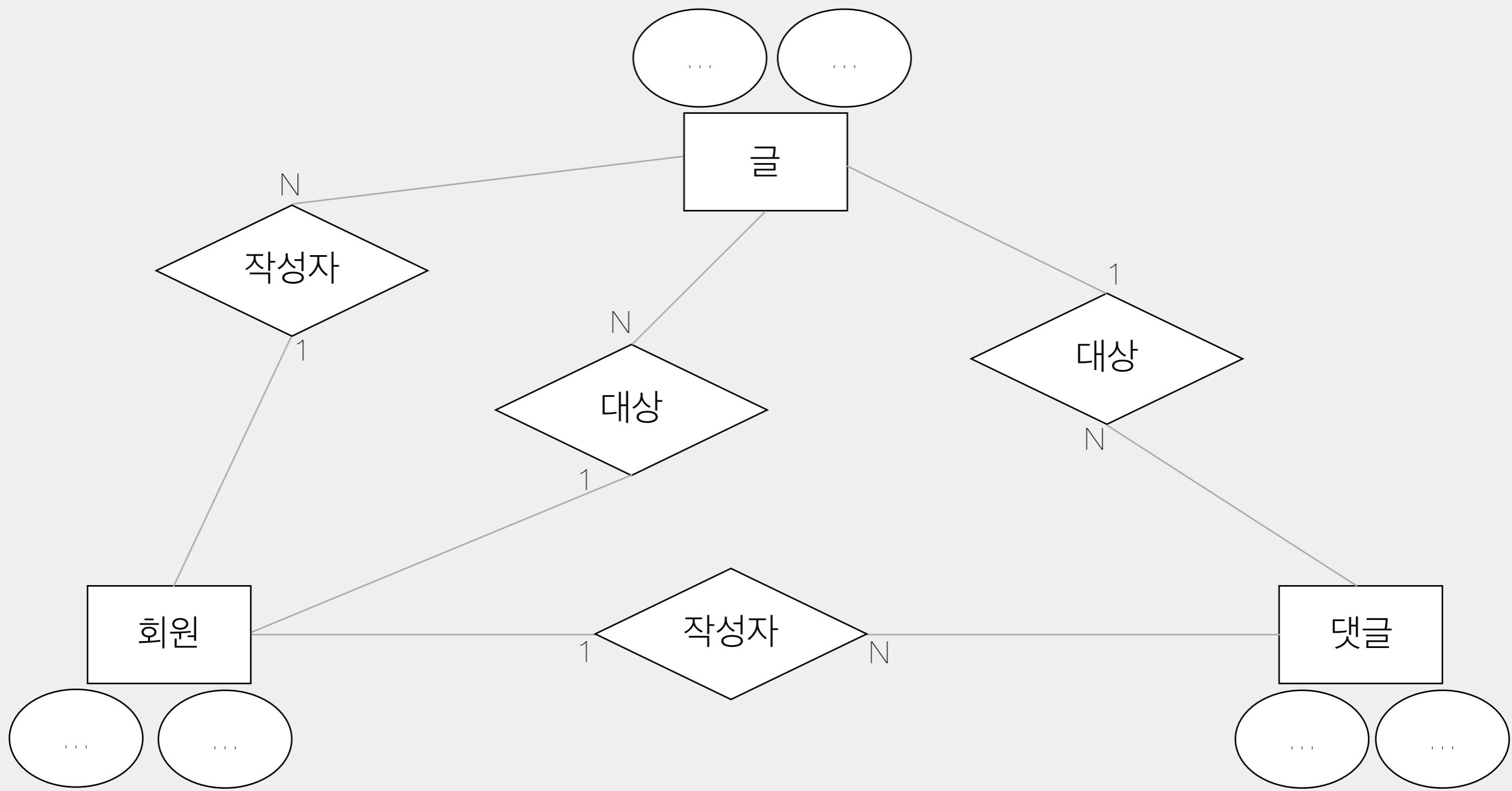
Likelion

관계형 데이터베이스

- 개체
- 관계
- 속성



관계형 데이터베이스



관계형 데이터베이스

사용자

ID	이름	이메일	...

글

ID	내용	시간	...

댓글

ID	내용	시간	...



관계형 데이터베이스

사용자

ID	이름	이메일	...

글

ID	내용	시간	작성자ID	답변락ID	...

댓글

ID	내용	시간	글ID	작성자ID	...

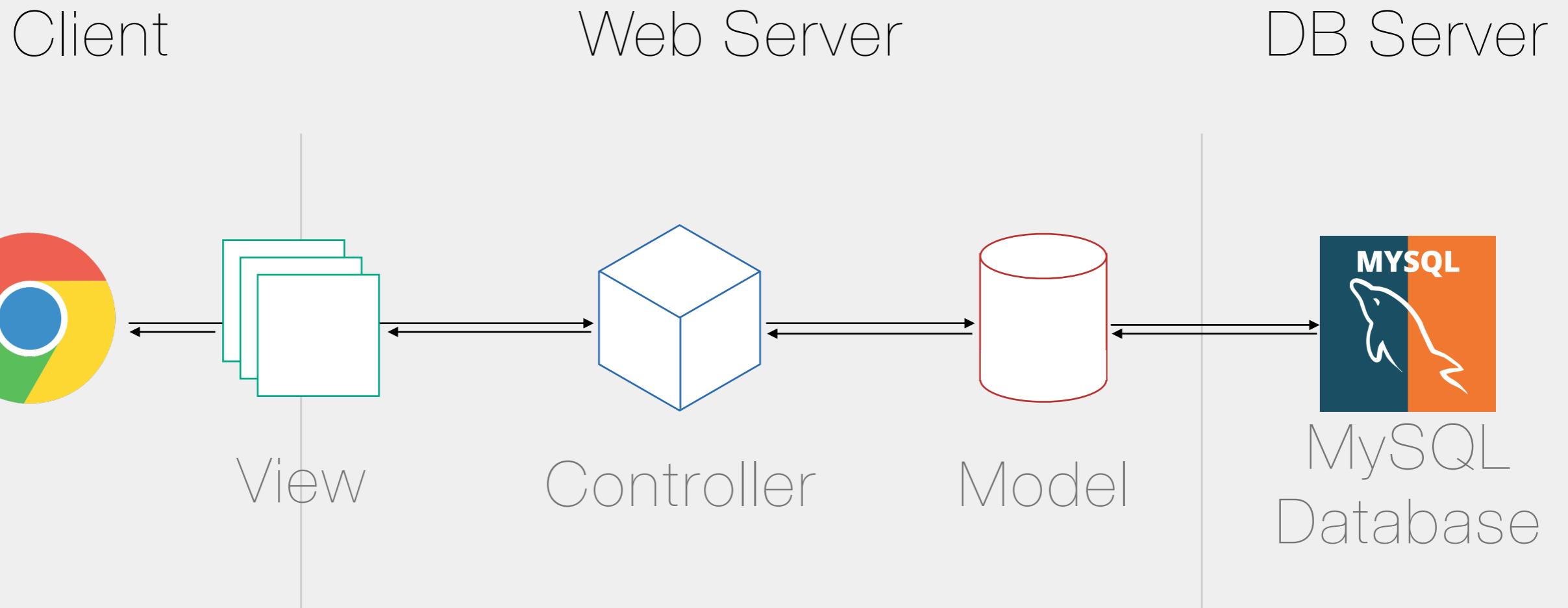


Database > CRUD

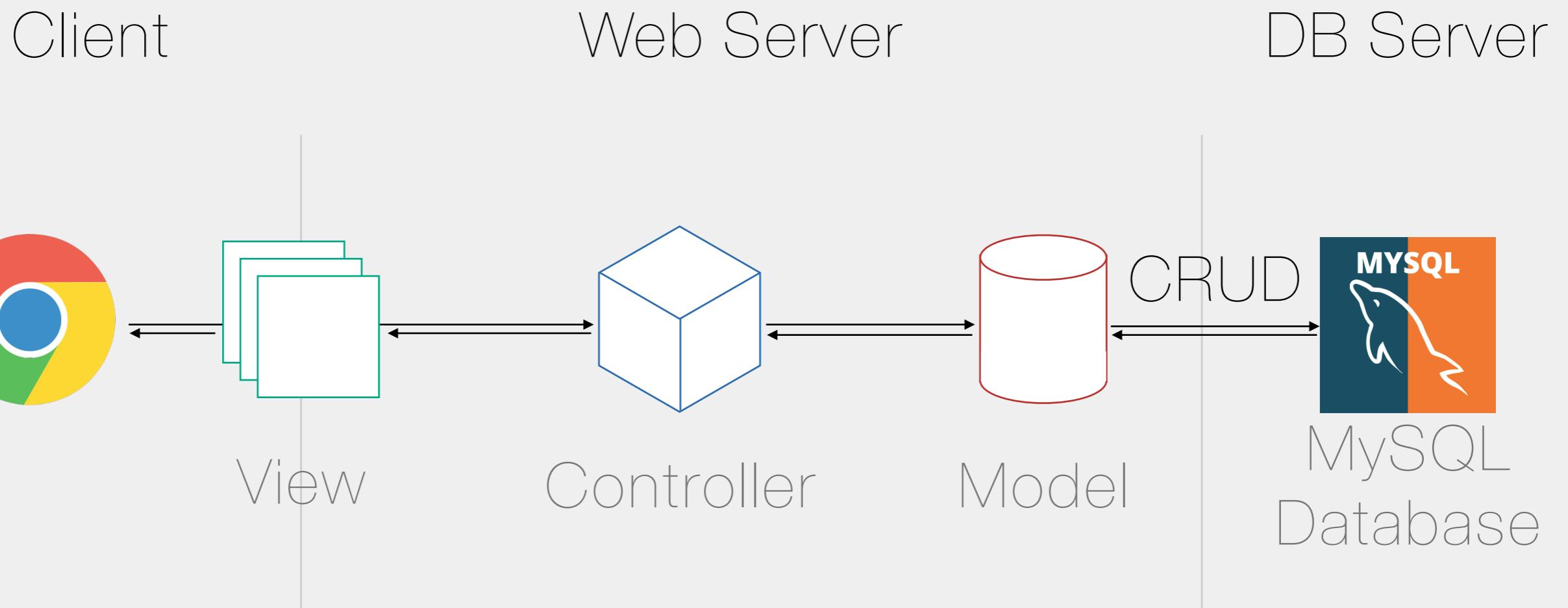
- Create : 데이터베이스의 테이블에 데이터 추가
- Read : 데이터베이스의 테이블에서 데이터를 가져옴
- Update : 데이터베이스 테이블의 데이터 수정
- Delete : 데이터베이스 테이블의 데이터 삭제



Database > CRUD



Database > CRUD



ORM

Object-Relational Mapping

쿼리를 간단하게 수행할 수 있는 방법
CRUD에 해당하는 파이썬 함수 제공



ORM > Schema

```
class Post(db.Model):
    id      = db.Column(db.Integer, primary_key = True)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'))
    user    = db.relationship('User', foreign_keys = [user_id])
    #,backref = db.backref('posts', cascade = 'all, delete-orphan', lazy = 'dynamic')
    )
    wall_id = db.Column(db.Integer, db.ForeignKey('user.id'))
    wall    = db.relationship('User', foreign_keys = [wall_id],
        backref = db.backref('wall_posts', cascade = 'all, delete-orphan', lazy = 'dynamic'))
    body    = db.Column(db.Text())
    edited_time = db.Column(db.DateTime, default = db.func.now(), onupdate = db.func.now())
    created_time = db.Column(db.DateTime, default = db.func.now())
    is_edited  = db.Column(db.Boolean, default = '0')
    is_secret   = db.Column(db.Boolean, default = '0')
```



ORM > Create

```
user = User(  
    email = data['email'],  
    username = data['username'],  
    gender = data['gender'],  
    password = db.func.md5(data['password']),  
    mobile = data['mobile'],  
    birthday = data['birthday'])  
  
db.session.add(user)  
db.session.commit()
```



ORM > Read

```
Post.query.filter(Post.wall_id == 10).order_by(db.desc(Post.edited_time)).limit(5)
```



ORM > Update

```
targetPost = get_post(data['post_id'])
targetPost.body = data['body']

db.session.commit()
```



ORM > Delete

```
targetPost = get_post(id)
db.session.delete(targetPost)
db.session.commit()
```



...



Likelion

Web Service Structure

