

# 스타트업 개발자와 함께 공부하는 Node.js

## 05. 템플릿 엔진

강의 내용은 강사가 별도로 명시하지 않는 한 비공개로 간주합니다.  
녹음이나 사진 촬영을 허락하지 않으며 콘텐츠를 블로그, SNS 등에 게시하거나 공개적으로 공유하지 마세요.

콘텐츠 공유 가능 여부에 대해 궁금한 점이 있는 경우 강사에게 문의하시기 바랍니다.



# 목차

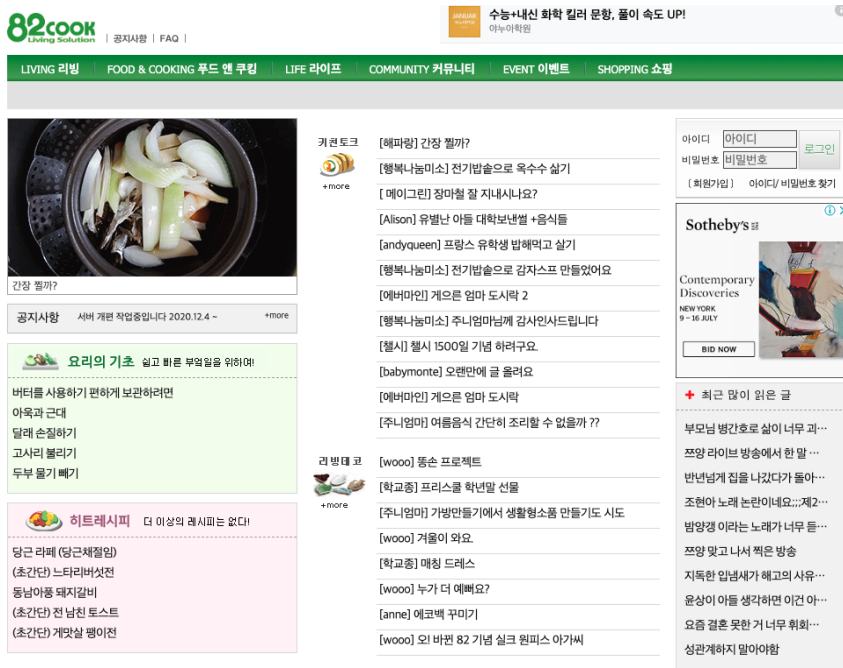
1. 템플릿 엔진 소개
2. Pug
3. Handlebars
4. EJS
5. 공지사항 요구사항 및 UI 설계
6. 공지사항 개발

# 플릿 엔진

# 템플릿 엔진

## 개요

- 템플릿 엔진을 사용하면 동적인 HTML 페이지를 쉽게 생성
- 템플릿 엔진은 HTML 파일에 데이터를 주입하여 동적으로 콘텐츠 생성



# 템플릿 엔진

## 종류

가장 인기 있는 템플릿 엔진 종류

❑ Pug(구 Jade)



❑ EJS(Embedded JavaScript)



❑ Handlebars(hbs)





# Pug

# Pug

## 프로젝트 생성 및 의존성 설치

```
npm init -y  
npm i express pug
```

- ❑ 템플릿 엔진 과정에서 사용될 프로젝트를 생성
- ❑ 디렉토리 : [C]-[nodejs]-[project]-[05]-[ch05\_01]



# Pug

## 기본 문법

```
doctype html
html
  head
    title= title
  body
    ul
      li Item A
      li Item B
```

```
<!DOCTYPE html>
<html>
  <head>
    <title>타이틀</title>
  </head>
  <body>
    <ul>
      <li>Item A</li>
      <li>Item B</li>
    </ul>
  </body>
</html>
```

# Pug

## 템플릿 파일 생성

```
1 doctype html
2 html
3   head
4     title= title
5   body
6     h1= message
7     p Webcome to ousr Pug example!
8
```

1. [C]-[nodejs]-[project]-[05]-[ch05\_01]-[views] 디렉토리 생성
2. 생성된 디렉토리에 index.pug 파일 생성
3. 왼쪽 코드 입력

# Pug

## app.js 파일

```
1 const express = require('express');
2 const app = express();
3 const PORT = 3000;
4
5 app.set('view engine', 'pug');
6 app.set('views', './views');
7
8 app.get('/', (req, res) => {
9   res.render('index',
10     {title: "Express Title",
11       message: 'Hello Pug'});
12 });
13
14 app.listen(PORT, () => {
15   console.log(`Server is running`);
16 });
```

- ❑ [C]-[nodejs]-[project]-[05]-[ch05\_01]  
디렉토리에 app.js 파일 추가
- ❑ 좌측 코드 입력
- ❑ vscode 터미널을 열고 node app.js 실행
- ❑ 브라우저에서 확인

# Handlebars

# Handlebars

## 프로젝트 생성 및 의존성 설치

```
npm init -y  
npm i express hbs
```

- ❑ 템플릿 엔진 과정에서 사용될 프로젝트를 생성
- ❑ 디렉토리 : [C]-[nodejs]-[project]-[05]-[ch05\_02]

# Handlebars

## 기본 문법

```
<!DOCTYPE html>
<html>
<head>
  <title>{{title}}</title>
</head>
<body>
  <h1>{{message}}</h1>
  <p>Welcome Handlebars</p>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
  <head>
    <title>타이틀</title>
  </head>
  <body>
    <h1>메시지</h1>
    <p>Welcome Handlebars</p>
  </body>
</html>
```



# Handlebars

## 템플릿 파일 생성 : main.handlebars

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <title>Example App</title>
6 </head>
7 <body>
8
9   {{{body}}}
10
11 </body>
12 </html>
```

1. [C]-[nodejs]-[project]-[05]-[ch05\_02]-[views]-[layouts] 디렉토리 생성
2. 생성된 디렉토리에 main.handlebars 파일 생성
3. 왼쪽 코드 입력

# Handlebars

## 템플릿 파일 생성 : index.handlebars

```
1 <h1>{{title}}</h1>
2 <h2>{{message}}</h2>
3 <p>Welcome to our simple Express.js
4 |   and Handlebars example!</p>
5
~
```

1. [C]-[nodejs]-[project]-[05]-[ch05\_02]-[views]
2. 디렉토리에 index.handlebars 파일 생성
3. 왼쪽 코드 입력

# Handlebars

## app.js 파일

```
1 const express = require('express');
2 const app = express();
3 const port = 3000;
4
5 // Handlebars 템플릿 엔진 설정
6 app.set('view engine', 'hbs');
7 app.set('views', './views');
8
9 // 기본 라우트 설정
10 app.get('/', (req, res) => {
11   res.render('index', {
12     title: 'Express and Handlebars Example',
13     message: 'Hello there!' });
14 });
15
16 // 서버 시작
17 app.listen(port, () => {
18   console.log(`Server is running
19     at http://localhost:${port}`);
20 });
```

- ❑ [C]-[nodejs]-[project]-[05]-[ch05\_02]  
디렉토리에 app.js 파일 추가
- ❑ 좌측 코드 입력
- ❑ vscode 터미널을 열고 node app.js 실행
- ❑ 브라우저에서 확인



EJS

```
function todoitem(data) {
  var self = this
  data = data || {}
  self.persistedProperties = {
    id: 'id',
    title: 'title',
    completed: 'completed'
  }
  self.observe({
    target: self,
    init: function() {
      self.title = data.title || ''
      self.completed = data.completed || false
    }
  })
  self.render = function() {
    return '<div id=' + self.id + ' class=' + self.className + '>' +
      '<div>' + self.title + '</div>' +
      '<div>' + self.completed + '</div>' +
      '</div>'
  }
}

// Backbone persisted properties function
function todoitem(data) {
  var self = this
  data = data || {}
  self.persistedProperties = {
    id: 'id',
    title: 'title',
    completed: 'completed'
  }
  self.observe({
    target: self,
    init: function() {
      self.title = data.title || ''
      self.completed = data.completed || false
    }
  })
  self.render = function() {
    return '<div id=' + self.id + ' class=' + self.className + '>' +
      '<div>' + self.title + '</div>' +
      '<div>' + self.completed + '</div>' +
      '</div>'
  }
}
```

# EJS

## 프로젝트 생성 및 의존성 설치

```
npm init -y  
npm i express ejs
```

- ❑ 템플릿 엔진 과정에서 사용될 프로젝트를 생성
- ❑ 디렉토리 : [C]-[nodejs]-[project]-[05]-[ch05\_03]

# EJS

## 기본 문법

```
<!DOCTYPE html>
<html>
<head>
  <title><%=title%></title>
</head>
<body>
  <h1><%=message%></h1>
  <p>Welcome Handlebars</p>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
  <head>
    <title>타이틀</title>
  </head>
  <body>
    <h1>메시지</h1>
    <p>Welcome Handlebars</p>
  </body>
</html>
```



# EJS

## 템플릿 파일 생성

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 |   <title><%= title %></title>
5 </head>
6 <body>
7 |   <h1><%= message %></h1>
8 |   <p>Welcome to EJS example!</p>
9 </body>
10 </html>
```

1. [C]-[nodejs]-[project]-[05]-[ch05\_03]-[views] 디렉토리 생성
2. 생성된 디렉토리에 index.ejs 파일 생성
3. 왼쪽 코드 입력

# EJS

## app.js 파일

```
1 const express = require('express');
2 const app = express();
3 const port = 3000;
4
5 // EJS 템플릿 엔진 설정
6 app.set('view engine', 'ejs');
7 app.set('views', './views');
8
9 // 기본 라우트 설정
10 app.get('/', (req, res) => {
11   res.render('index', {
12     title: 'Express and EJS Example',
13     message: 'Hello there!' });
14 });
15
16 // 서버 시작
17 app.listen(port, () => {
18   console.log(`Server is running
19     at http://localhost:${port}`);
20 });
```

- ❑ [C]-[nodejs]-[project]-[05]-[ch05\_03]  
디렉토리에 app.js 파일 추가
- ❑ 좌측 코드 입력
- ❑ vscode 터미널을 열고 node app.js 실행
- ❑ 브라우저에서 확인

# EJS

## for 문

```
// 05/ch05_05/views/for.ejs
```

```
1 <html>
2   <meta charset="UTF-8">
3   <title>EJS Tag</title>
4   <body>
5     <% for (var i=0; i < 10; i++) { %>
6       <h1>Tag Test</h1>
7     <% } %>
8   </body>
9 </html>
10
```

```
// 05/ch05_03/app.js
```

```
app.get('/for', (req, res) => {
  res.render('for');
});
```

# EJS

## if 문

```
// 05/ch05_03/views/if.ejs
```

```
1 <html>
2   <meta charset="UTF-8">
3   <title>EJS Tag</title>
4   <body>
5     <% for (var i=0; i < 10; i++) {
6       |   if( i % 2 == 0) {
7       |
8       |   <h1>Tag Test => <%=i%></h1>
9       |   <%
10      |   }
11      |   }
12      |   <%>
13    </body>
14 </html>
```

```
// 05/ch05_03/app.js
```

```
app.get('/if', (req, res) => {
  |   res.render('if');
  | });
```

# EJS

## 배열과 딕셔너리 출력

```
// 05/ch05_03/views/sample.ejs
```

```
1 <html>
2   <meta charset="UTF-8">
3   <title>EJS Tag</title>
4   <body>
5     <ul>
6       <% for (var i=0; i < data.length; i++) { %>
7         <li><%=data[i]['title']%> <%=data[i]['content']%></li>
8       <% } %>
9     </ul>
10  </body>
11 </html>
12
```

```
// 05/ch05_03/app.js
```

```
24 const data = [
25   {'title': 'Title # 1', 'content': 'This is Content 1'},
26   {'title': 'Title # 2', 'content': 'This is Content 2'},
27   {'title': 'Title # 3', 'content': 'This is Content 3'},
28 ];
29 app.get('/sample', (req, res) => {
30   res.render('sample', {data: data})
31 })
```

# EJS

## JSON 파일에서 읽어서 출력

```
// 05/ch05_03/views/test.ejs
```

```
1 <html>
2   <meta charset="UTF-8">
3   <title>Print Content</title>
4   <body>
5     <table>
6       <tbody>
7         <%
8           items.forEach(function(item) {
9             %>
10            <tr><td><%= item['title'] %></td>
11              <td><%= item['content'] %></td></tr>
12            <%
13              %>
14            %>
15            %>
16          %>
17        %>
18      </tbody>
19    </table>
20  </body>
21 </html>
```

```
// 05/ch05_03/app.js
```

```
app.get('/test', (req, res) => {
  const result = fs.readFileSync(
    'test.json', 'utf-8');
  const data = JSON.parse(result);
  res.render('test', {items: data["result"]})
});
```



# 공지사항 게시판 설계



# 공지사항 게시판 설계

## Preview

My Board

Home

Board

About

Write

제목	작성자	등록일
<a href="#">공지사항 2번 입니다.</a>	writer 2	2024-06-02
<a href="#">공지사항 3번 입니다.</a>	writer 3	2024-06-03
<a href="#">공지사항 4번 입니다.</a>	writer 4	2024-06-04
<a href="#">공지사항 5번 입니다.</a>	writer 5	2024-06-05
<a href="#">공지사항 6번 입니다.</a>	writer 6	2024-06-06
<a href="#">공지사항 7번 입니다.</a>	writer 7	2024-06-07
<a href="#">공지사항 8번 입니다.</a>	writer 8	2024-06-08
<a href="#">공지사항 9번 입니다.</a>	writer 9	2024-06-09
<a href="#">공지사항 10번 입니다.</a>	writer 10	2024-06-10
<a href="#">공지사항 11번 입니다.</a>	tester	2024-01-01

© 2024 My board. All rights reserved

# 공지사항 게시판 설계

## 주요기능

주요기능	설명	URI	method
공지사항 목록	공지사항 목록을 조회 합니다.	/list	GET
공지사항 글 상세	목록에서 링크를 클릭하면 상세내용을 보여줍니다	/detail/:id	GET
공지사항 글 쓰기	공지사항의 내용을 입력합니다. 제목과 내용	/write	GET,POST
공지사항 글 수정	공지사항의 내용을 수정합니다.	/update/:id	GET,POST
공지사항 글 삭제	공지사항을 삭제합니다.	/delete/:id	GET
홈페이지	홈 화면입니다.		

# 공지사항 게시판 설계

## 화면 UI 설계 – 공지사항 목록

공지사항 목록

Write

제목	작성자	등록일
<a href="#">공지사항 2번 입니다.</a>	writer 2	2024-06-02
<a href="#">공지사항 3번 입니다.</a>	writer 3	2024-06-03
<a href="#">공지사항 4번 입니다.</a>	writer 4	2024-06-04
<a href="#">공지사항 5번 입니다.</a>	writer 5	2024-06-05
<a href="#">공지사항 6번 입니다.</a>	writer 6	2024-06-06
<a href="#">공지사항 7번 입니다.</a>	writer 7	2024-06-07
<a href="#">공지사항 8번 입니다.</a>	writer 8	2024-06-08
<a href="#">공지사항 9번 입니다.</a>	writer 9	2024-06-09
<a href="#">공지사항 10번 입니다.</a>	writer 10	2024-06-10
<a href="#">공지사항 11번 입니다.</a>	tester	2024-01-01
<a href="#">안녕하세요</a>	tester	2024-01-01
<a href="#">하이요</a>	tester	2024-07-12

# 공지사항 게시판 설계

## 화면 UI 설계 – 공지사항 상세

### 공지사항 상세

#### 공지사항 2번 입니다.

2024-06-02 by [writer 2](#)

코드는 시보다 더 많이 읽힌다. 좋은 코드는 읽기 쉬워야 한다. - Guido van Rossum (Python 창시자) 완벽한 소프트웨어를 만드는 가장 좋은 방법은 코드를 전혀 작성하지 않는 것이다. - Tom DeMarco (소프트웨어 엔지니어) 좋은 프로그래머는 코드가 아닌 데이터를 조작하는 데 더 많은 시간을 보낸다. - Robert C. Martin (클린 코드의 저자) 모든 복잡한 문제에는 잘못된 간단한 해답이 있다. - H.L. Mencken 프로그래밍이 재미없다면, 잘못하고 있는 것이다. - Andrew Hunt (The Pragmatic Programmer 공동 저자) 코드를 10배 빨리 작성하는 개발자는 없다. 대신, 그들은 10배 적은 코드를 작성한다. - Robert C. Martin 소프트웨어가 작동하는 것은 중요하지만, 소프트웨어가 이해되는 것은 더 중요하다. - Martin Fowler (Refactoring의 저자) 코드는 작성하는 시간보다 읽는 시간이 더 많이 걸린다. - Douglas Crockford

[Update](#)[Delete](#)

© 2024 My board. All rights reserved

# 공지사항 게시판 설계

## 화면 UI 설계 – 공지사항 글쓰기

공지사항 수정

제목

내용

Save

© 2024 My board. All rights reserved



# 공지사항 게시판 설계

## 화면 UI 설계 – 공지사항 수정

### 공지사항 수정

제목

공지사항 2번 입니다.

내용

코드는 시보다 더 많이 읽힌다. 좋은 코드는 읽기 쉬워야 한다. - Guido van Rossum (Python 창시자)

완벽한 소프트웨어를 만드는 가장 좋은 방법은 코드를 전혀 작성하지 않는 것이다. - Tom DeMarco (소프트웨어 엔지니어)

좋은 프로그래머는 코드가 아닌 데이터를 조작하는 데 더 많은 시간을 보낸다. - Robert C. Martin (클린 코드의 저자)

모든 복잡한 문제에는 잘못된 간단한 해답이 있다. - H.L. Mencken

프로그래밍이 재미없다면, 잘못하고 있는 것이다. - Andrew Hunt (The Pragmatic Programmer 공동 저자)

코드를 10배 빨리 작성하는 개발자는 없다. 대신, 그들은 10배 적은 코드를 작성한다. - Robert C. Martin

소프트웨어가 작동하는 것은 중요하지만, 소프트웨어가 이해되는 것은 더 중요하다. - Martin Fowler (Refactoring의 저자)

코드는 작성하는 시간보다 읽는 시간이 더 많이 걸린다. - Douglas Crockford

Save

© 2024 My board. All rights reserved

# 공지사항 게시판 설계

## 공지사항 데이터 : JSON

```
1 {  
2   "result": [  
3     {  
4       "id": 2,  
5       "title": "공지사항 2번 입니다.",  
6       "content": "코드는 시보다 더 많이 읽힌다. 좋은 코드는 읽기 쉬워야 한다. - Guido van Rossum (Python 창시자) \r\n완벽한 소프트웨어를 만드는 가장  
7       "writer": "writer 2",  
8       "write_date": "2024-06-02"  
9     },  
10    {  
11      "id": 3,  
12      "title": "공지사항 3번 입니다.",  
13      "content": "코드는 시보다 더 많이 읽힌다. 좋은 코드는 읽기 쉬워야 한다. - Guido van Rossum (Python 창시자) 완벽한 소프트웨어를 만드는 가장 좋은 방  
14      "writer": "writer 3",  
15      "write_date": "2024-06-03"  
16    },  
17    {  
18      "id": 4,  
19      "title": "공지사항 4번 입니다.",  
20      "content": "코드는 시보다 더 많이 읽힌다. 좋은 코드는 읽기 쉬워야 한다. - Guido van Rossum (Python 창시자) 완벽한 소프트웨어를 만드는 가장 좋은 방  
21      "writer": "writer 4",  
22      "write_date": "2024-06-04"  
23    },  
24  ]  
25 }
```

# 공지사항 게시판 개발



# 공지사항 개발

## 프로젝트 생성

```
npm init -y  
npm install express ejs moment nodemon
```

```
“dev”: “nodemon server.js”
```

- ❑ 공지사항 과정에서 사용될 프로젝트를 생성
- ❑ 디렉토리 : [C]-[nodejs]-[project]-[05]-[ch05\_04]
- ❑ package.json 에 스크립트 추가

# 공지사항 개발

## 공지사항 목록

//views/pages/list.ejs

```
<div class="col-sm-12">
  <table class="table table-striped">
    <thead class="thead-dark">
      <tr>
        <th scope="col" width="50%">제목</th>
        <th scope="col">작성자</th>
        <th scope="col">등록일</th>
      </tr>
    </thead>
    <tbody>
      <% items.forEach((item) => {%>
        <tr scope="row">
          <td><a href="/detail/<%=item['id']%>"><%=item['title']%></a></td>
          <td><%=item['writer']%></td>
          <td><%=item['write_date']%></td>
        </tr>
      <% }>};%>
    </tbody>
  </table>
```

//server.js

```
app.get('/list', (req, res) => {
  const result = fs.readFileSync('test.json', 'utf-8');
  const data = JSON.parse(result);

  res.render('pages/list', {items: data['result']});
});
```

# 공지사항 개발

## 글 상세

//views/pages/detail.ejs

```
36
37 <form action="/update/<%=detail['id']%>" method="get" id="update"></form>
38 <form action="/delete/<%=detail['id']%>" method="get" id="delete"></form>
39 <div class="row">
40 <div class="blog-post">
41 <h2 class="blog-post-title"><%=detail['title']%></h2>
42 <p class="blog-post-meta"><%=detail['write_date']%> by <a href="#"><%=detail['writer']%></a></p>
43 <p><%=detail['content']%></p>
44 </div>
45 </div>
46 <button type="submit" form="update" value="submit" class="btn btn-warning">Update</button>
47 <button type="submit" onclick="delete_item()" value="submit" class="btn btn-danger">Delete</button>
```

// server.js

```
22 app.get('/detail/:id', (req, res) => {
23   const id = req.params.id;
24
25   const result = fs.readFileSync('test.json', 'utf-8');
26   const data = JSON.parse(result);
27   let detail = {};
28
29   data['result'].forEach((item) => {
30     if(item['id'] == id) {
31       detail = item;
32     }
33   });
34
35   // const detail = data['result'].filter((item) => {
36   //   return item['id'] == id;
37   // })[0];
38
39   res.render('pages/detail', {detail: detail});
40 });
```

# 공지사항 개발

## 글쓰기 페이지

### //views/pages/write.ejs

```
28 | <form action="/write" method="post" id="update">
29 |   <div class="row">
30 |     <div class="col-sm-12 py-3">
31 |       <div class="input-group">
32 |         <div class="input-group-prepend">
33 |           <span class="input-group-text">제목</span>
34 |         </div>
35 |         <input type="text" name="title" id="title" value="" size="20"></td>
36 |       </div>
37 |     </div>
38 |     <div class="col-sm-12 py-3">
39 |       <div class="input-group">
40 |         <div class="input-group-prepend">
41 |           <span class="input-group-text">내용</span>
42 |         </div>
43 |         <textarea class="form-control" aria-label="내용"
44 |           name="content" id="content" rows="10" cols="20"></textarea>
45 |       </div>
46 |     </div>
47 |     <div class="col-sm-12 py-3">
48 |       <button type="submit" form="update" value="submit"
49 |         class="btn btn-danger">Save</button>
50 |     </div>
51 |   </div>
52 | </form>
```

### //server.js

```
43 app.get('/write', (req, res) => {
44 |   res.render('pages/write');
45 | });
```

# 공지사항 개발

## 글 저장

//server.js

```
47 app.use(express.urlencoded({ extended: true }));
48
49 app.post('/write', (req, res) => {
50   console.log('/write post', req.body);
51   // save data to file
52   const result = fs.readFileSync('test.json', 'utf-8');
53   let data = JSON.parse(result);
54
55   const last = data['result'].slice(-1);
56   const last_id = last[0]['id'] + 1;
57   const write_date = moment().format('YYYY-MM-DD');
58
59   data['result'].push({
60     'id': last_id, 'title': req.body.title, 'content': req.body.content, 'writer': 'tester' , 'write_date': write_date
61   });
62
63   fs.writeFileSync('test.json', JSON.stringify(data), 'utf-8');
64   res.redirect('/list')
65 });
```



# 공지사항 개발

## 글 수정 페이지

### //views/pages/update.ejs

```

30 | <form action="/update/<%=detail['id']%>" method="post" id="update">
31 | <div class="row">
32 |   <div class="col-sm-12 py-3">
33 |     <div class="input-group">
34 |       <div class="input-group-prepend">
35 |         <span class="input-group-text">제목</span>
36 |       </div>
37 |       <input type="text" name="title" id="title"
38 |         value="<%=detail['title']%>" size="20"></td>
39 |     </div>
40 |   </div>
41 |   <div class="col-sm-12 py-3">
42 |     <div class="input-group">
43 |       <div class="input-group-prepend">
44 |         <span class="input-group-text">내용</span>
45 |       </div>
46 |       <textarea class="form-control" aria-label="내용" name="content" id="content"
47 |         rows="10" cols="20"> <%=detail['content']%></textarea>
48 |     </div>
49 |   </div>
50 |   <div class="col-sm-12 py-3">
51 |     <button type="submit" form="update" value="submit"
52 |       class="btn btn-danger">Save</button>
53 |   </div>
54 | </div>
55 | </form>

```

### // server.js

```

68 app.get('/update/:id', (req, res) => {
69   const id = req.params.id;
70
71   const result = fs.readFileSync('test.json', 'utf-8');
72   const data = JSON.parse(result);
73   let detail = {};
74
75   data['result'].forEach((item) => {
76     if(item['id'] == id) {
77       detail = item;
78     }
79   });
80
81   res.render('pages/update', {detail: detail});
82 });

```

# 공지사항 개발

## 글 수정

//server.js

```
84 app.post('/update/:id', (req, res) => {
85   const id = req.params.id;
86
87   const result = fs.readFileSync('test.json', 'utf-8');
88   let data = JSON.parse(result);
89
90   for(item of data['result']) {
91     if(item['id'] == id) {
92       item['title'] = req.body.title;
93       item['content'] = req.body.content;
94     }
95   }
96
97   fs.writeFileSync('test.json', JSON.stringify(data), 'utf-8');
98   res.redirect('/detail/'+id)
99
100 });
```

# 공지사항 개발

## 글 삭제

//server.js

```
102 app.get('/delete/:id', (req, res) => {
103     const id = req.params.id;
104
105     const result = fs.readFileSync('test.json', 'utf-8');
106     let data = JSON.parse(result);
107
108     let element_idx = 0;
109
110     data['result'].forEach((e, i) => {
111         if(e['id'] == id) {
112             element_idx = i;
113         }
114     });
115     data['result'].splice(element_idx, 1);
116
117     fs.writeFileSync('test.json', JSON.stringify(data), 'utf-8');
118     res.redirect('/list');
119
120 });
```



백문이 불여일타