스타트업 개발자와 함께 공부하는

Node.js

05. 템플릿 엔진

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

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

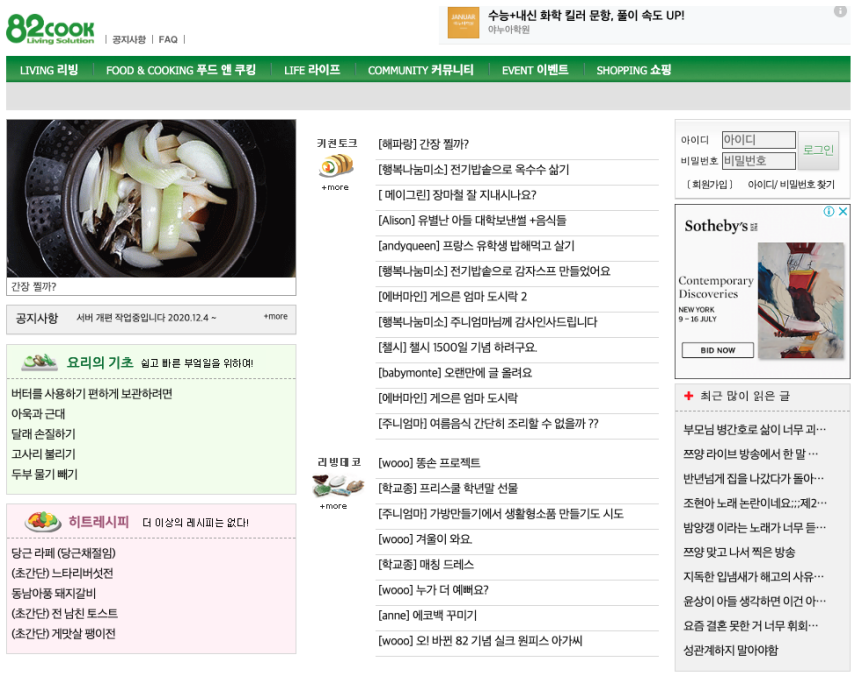
목차

템플릿 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

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Example App</title>
</head>
<body>

    {{{body}}}

</body>
</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

```
<h1>{{title}}</h1>
<h2>{{message}}</h2>
<p>Welcome to our simple Express.js
  and Handlebars example!</p>

```

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

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

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

```ejs
<html>
    <meta charset="UTF-8">
    <title>EJS Tag</title>
    <body>
        <% for (var i=0; i < 10; i++) { %>
            <h1>Tag Test</h1>
        <% } %>
    </body>
</html>

```

// 05/ch05_03/app.js

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

EJS

if 문

```
// 05/ch05_03/views/if.ejs
<html>
    <meta charset="UTF-8">
    <title>EJS Tag</title>
    <body>
        <% for (var i=0; i < 10; i++) {
            if( i % 2 == 0) {
        %>
            <h1>Tag Test => <%=i%></h1>
        <%
            }
        }
        %>
    </body>
</html>
```

```
// 05/ch05_03/app.js
app.get('/if', (req, res) => {
  res.render('if');
});
```

EJS

배열과 딕셔너리 출력

// 05/ch05_03/views/sample.ejs

```
<html>
    <meta charset="UTF-8">
    <title>EJS Tag</title>
    <body>
        <ul>
        <% for (var i=0; i < data.length; i++) { %>
            <li><%=data[i]['title']%> <%=data[i]['content']%></li>
        <% } %>
        </ul>
    </body>
</html>

```

// 05/ch05_03/app.js

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

EJS

JSON 파일에서 읽어서 출력

// 05/ch05_03/views/test.ejs

```
<html>
    <meta charset="UTF-8">
    <title>Print Content</title>
    <body>
        <table>
            <tbody>
            <%
                items.forEach(function(item) {
            %>
                <tr><td><%= item['title'] %></td>
                    <td><%= item['content'] %></td></tr>
            <%
                });
            %>
            </tbody>
        </table>
    </body>
</html>
```

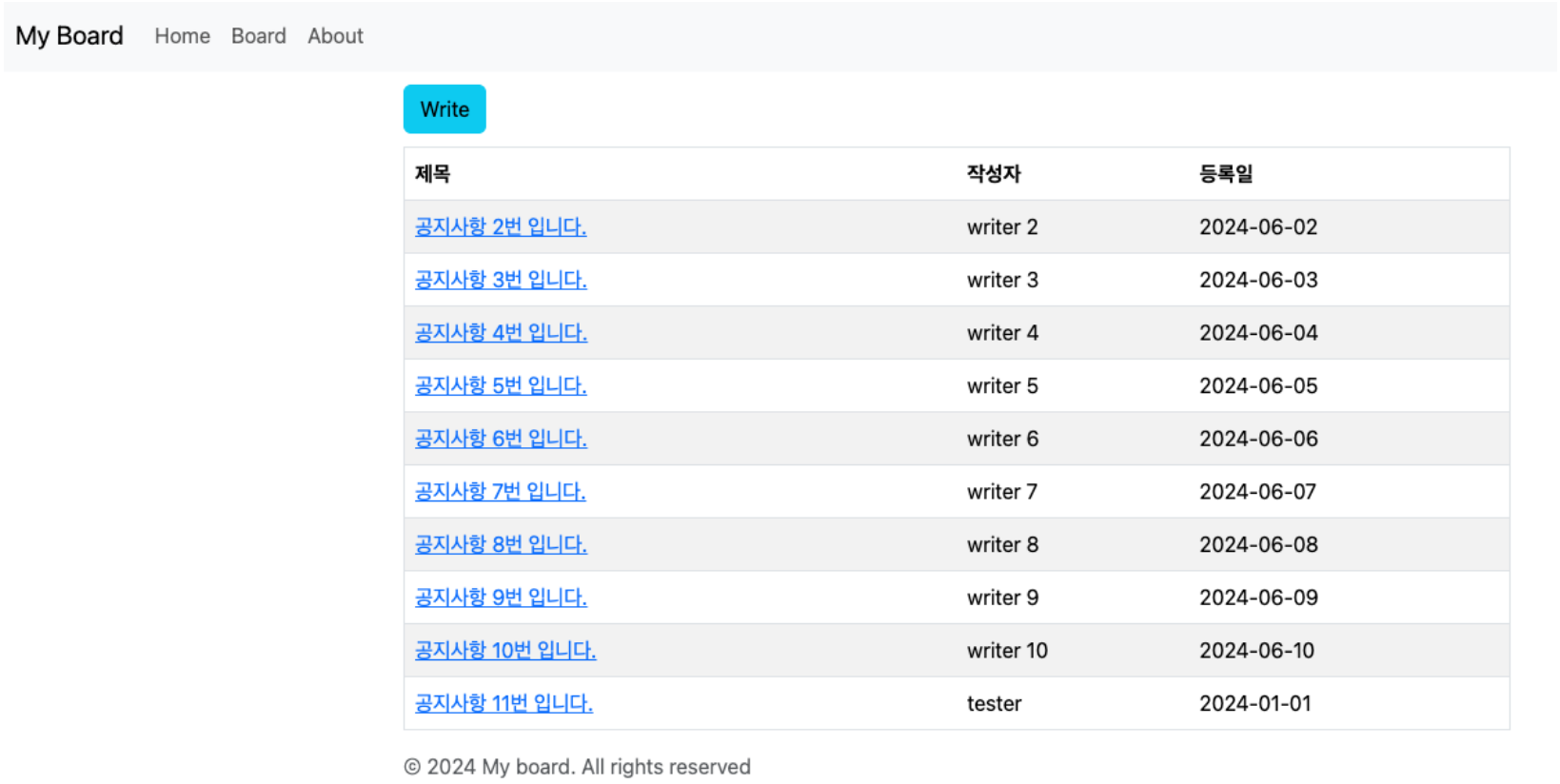
// 05/ch05_03/app.js

```
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



공지사항 게시판 설계

주요기능

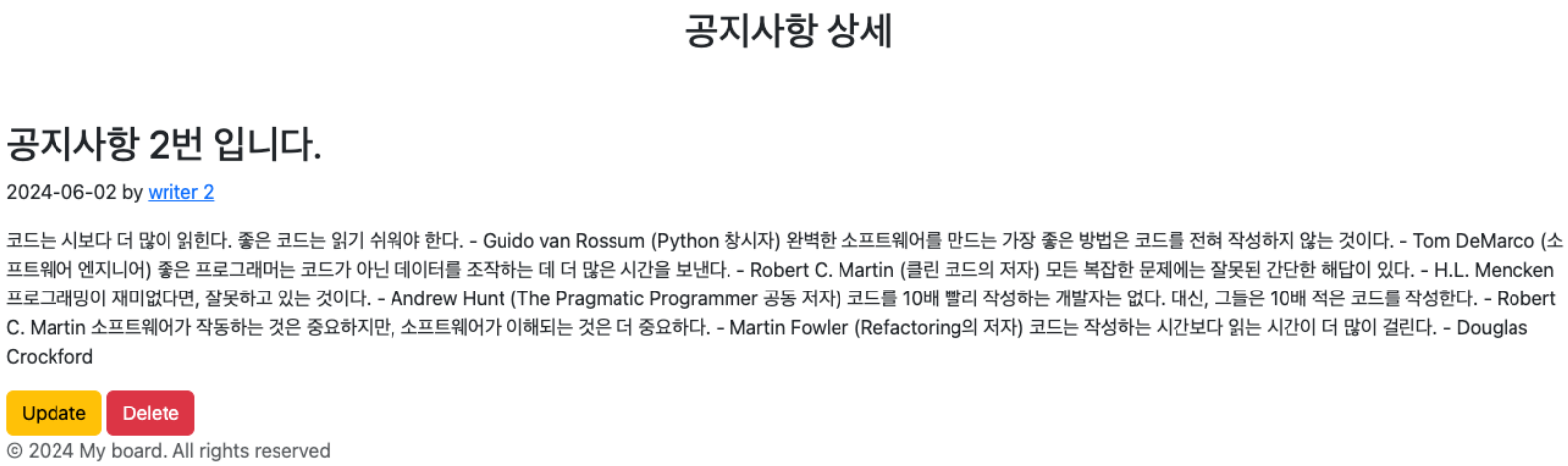
주요기능	설명	URI	method
공지사항 목록	공지사항 목록을 조회 합니다.	/list	GET
공지사항 글 상세	목록에서 링크를 클릭하면 상세내용을 보여줍니다	/detail/:id	GET
공지사항 글 쓰기	공지사항의 내용을 입력합니다. 제목과 내용	/write	GET,POST
공지사항 글 수정	공지사항의 내용을 수정합니다.	/update/:id	GET,POST
공지사항 글 삭제	공지사항을 삭제합니다.	/delete/:id	GET
홈페이지	홈 화면입니다.		

공지사항 게시판 설계

화면 UI 설계 – 공지사항 목록

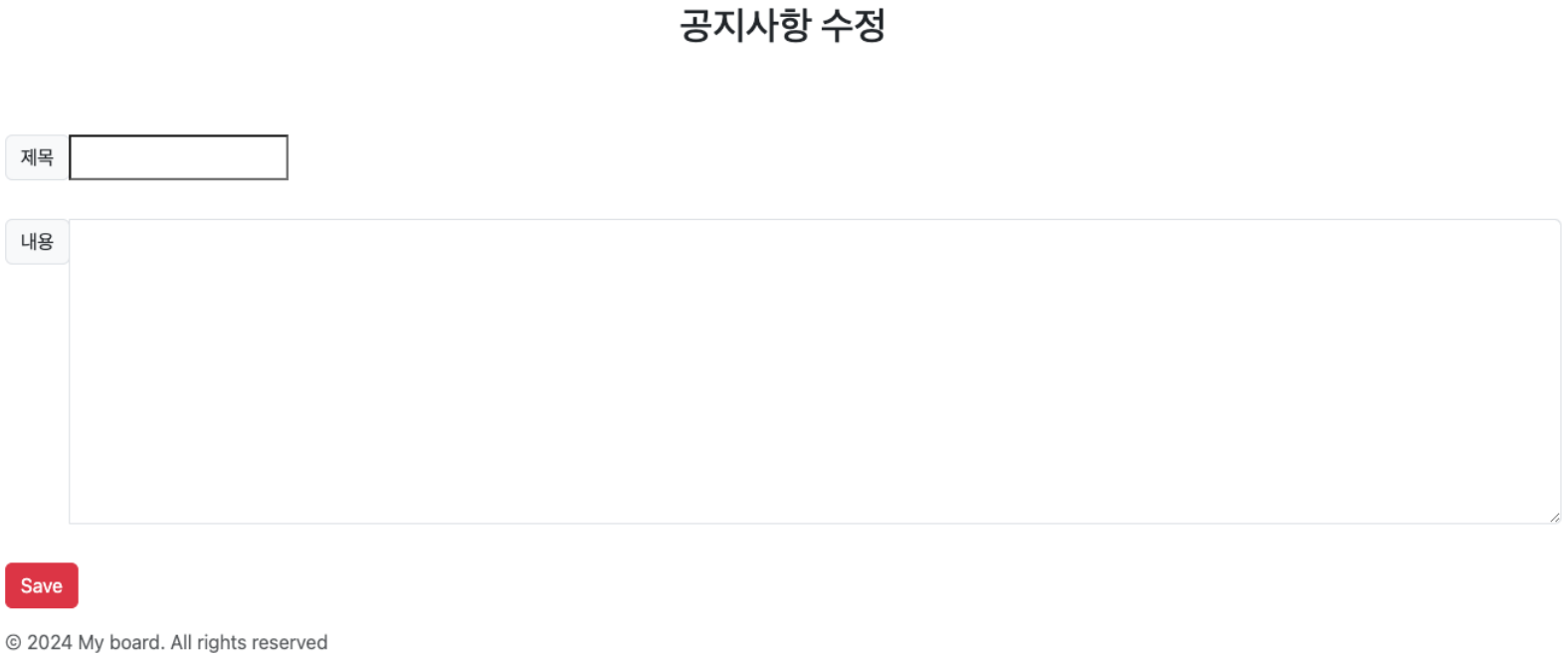
공지사항 게시판 설계

화면 UI 설계 – 공지사항 상세



공지사항 게시판 설계

화면 UI 설계 – 공지사항 글쓰기



공지사항 게시판 설계

화면 UI 설계 – 공지사항 수정

공지사항 게시판 설계

공지사항 데이터 : JSON


```
{
    "result": [
        {
            "id": 2,
            "title": "공지사항 2번 입니다.",
            "content": " 코드는 시보다 더 많이 읽힌다. 좋은 코드는 읽기 쉬워야 한다. - Guido van Rossum (Python 창시자) \r\n완벽한 소프트웨어를 만드는 가장 
            "writer": "writer 2",
            "write_date": "2024-06-02"
        },
        {
            "id": 3,
            "title": "공지사항 3번 입니다.",
            "content": "코드는 시보다 더 많이 읽힌다. 좋은 코드는 읽기 쉬워야 한다. - Guido van Rossum (Python 창시자) 완벽한 소프트웨어를 만드는 가장 좋은 방
            "writer": "writer 3",
            "write_date": "2024-06-03"
        },
        {
            "id": 4,
            "title": "공지사항 4번 입니다.",
            "content": "코드는 시보다 더 많이 읽힌다. 좋은 코드는 읽기 쉬워야 한다. - Guido van Rossum (Python 창시자) 완벽한 소프트웨어를 만드는 가장 좋은 방
            "writer": "writer 4",
            "write_date": "2024-06-04"
        },
```

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

```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

```server
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

```ejs
<form action="/update/<%=detail['id']%>" method="get" id="update"></form>
<form action="/delete/<%=detail['id']%>" method="get" id="delete"></form>
  <div class="row">
    <div class="blog-post">
      <h2 class="blog-post-title"><%=detail['title']%></h2>
      <p class="blog-post-meta"><%=detail['write_date']%> by <a href="#"><%=detail['writer']%></a></p>
      <p><%=detail['content']%></p>
    </div>
</div>
<button type="submit" form="update" value="submit" class="btn btn-warning">Update</button>
<button type="submit" onclick="delete_item()" value="submit" class="btn btn-danger">Delete</button>
```

// server.js

```js
app.get('/detail/:id', (req, res) => {
    const id = req.params.id;

    const result = fs.readFileSync('test.json', 'utf-8');
    const data = JSON.parse(result);
    let detail = {};

    data['result'].forEach((item) => {
        if(item['id'] == id) {
            detail = item;
        }
    });

    // const detail = data['result'].filter((item) => {
    //     return item['id'] == id;
    // })[0];

    res.render('pages/detail', {detail: detail});
});
```

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

```ejs
<form action="/update/<%=detail['id']%>" method="post" id="update">
<div class="row">
    <div class="col-sm-12 py-3">
      <div class="input-group">
        <div class="input-group-prepend">
          <span class="input-group-text">제목</span>
        </div>
        <input type="text" name="title" id="title"
          value="<%=detail['title']%>" size="20"></td>
      </div>
    </div>
    <div class="col-sm-12 py-3">
      <div class="input-group">
        <div class="input-group-prepend">
          <span class="input-group-text">내용</span>
        </div>
        <textarea class="form-control" aria-label="내용" name="content" id="content"
          rows="10" cols="20"> <%=detail['content']%></textarea>
      </div>
    </div>
    <div class="col-sm-12 py-3">
      <button type="submit" form="update" value="submit"
      class="btn btn-danger">Save</button>
    </div>
</div>
</form>
```

// server.js

```js
app.get('/update/:id', (req, res) => {
    const id = req.params.id;

    const result = fs.readFileSync('test.json', 'utf-8');
    const data = JSON.parse(result);
    let detail = {};

    data['result'].forEach((item) => {
        if(item['id'] == id) {
            detail = item;
        }
    });

    res.render('pages/update', {detail: detail});
});
```

공지사항 개발

글 수정

```
//server.js
app.post('/update/:id', (req, res) => {
    const id = req.params.id;

    const result = fs.readFileSync('test.json', 'utf-8');
    let data = JSON.parse(result);

    for(item of data['result']) {
        if(item['id'] == id) {
            item['title'] = req.body.title;
            item['content'] = req.body.content;
        }
    }

    fs.writeFileSync('test.json', JSON.stringify(data), 'utf-8');
    res.redirect('/detail/'+id)

});
```

공지사항 개발

글 삭제

```
//server.js
app.get('/delete/:id', (req, res) => {
    const id = req.params.id;

    const result = fs.readFileSync('test.json', 'utf-8');
    let data = JSON.parse(result);

    let element_idx = 0;

    data['result'].forEach((e, i) => {
        if(e['id'] == id) {
            element_idx = i;
        }
    });
    data['result'].splice(element_idx, 1);

    fs.writeFileSync('test.json', JSON.stringify(data), 'utf-8');
    res.redirect('/list');

});
```

백문이 불여일타