

[경기도형 대학생 취업브리지 사업]

# 차세대 웹모바일 플랫폼 기반 응용SW 개발 교육과정

11주차: Node.js 응용 (서비스 설계 및 개발)

2020.6.12

화상강의를 9시부터 시작합니다.

## 이번강좌에서는 ....

**Node.JS 기반 백엔드 서비스개발은** 쇼핑몰 판매 서비스를 주제로 웹사이트를 개발하는 과정을 프로젝트 형식으로 살펴보며, 각 단계마다 필요한 기술과 해결을 위한 접근방식을 소개합니다.

특히 이 강의를 통해 **Node.JS, express** 웹프레임워크, **ejs** 미들웨어, **Bootstrap 4.x** 기술을 활용하는 기술을 배우며, 소스코드 템플릿과 샘플을 직접 다루어 보다 빠른 개발능력을 키우는데 혁신적인 학습방법을 제공합니다.

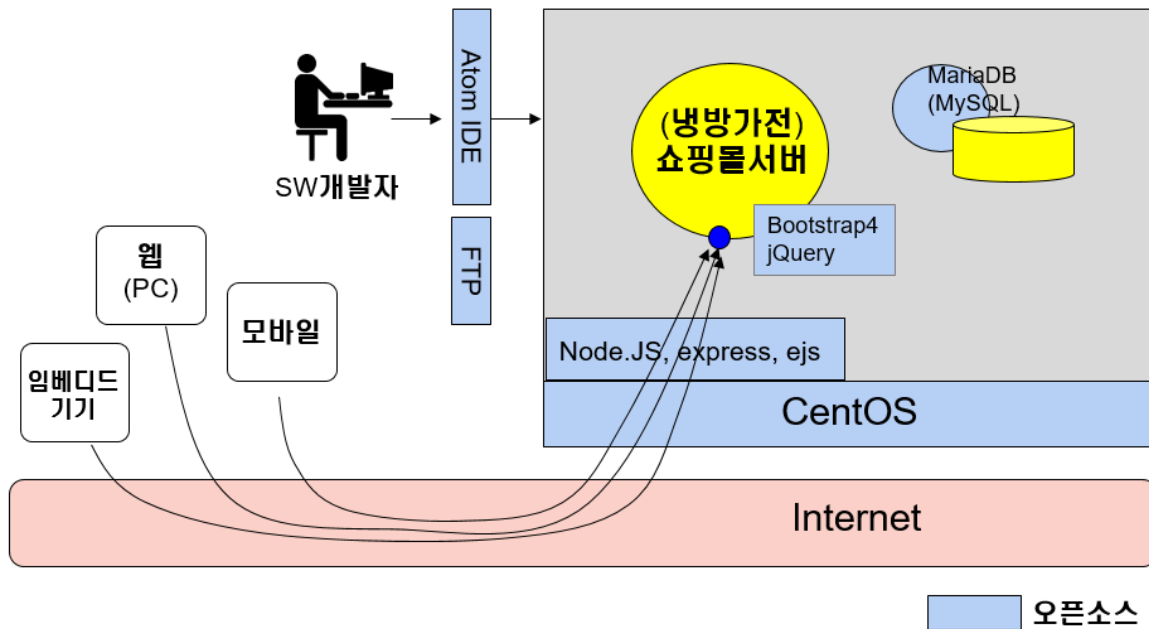


### ■ 목차

- 01. 서비스 소개(쇼핑몰 웹사이트 구축)
- 02. 개발환경 구축 (Node.js/Express, Bootstrap, Atom)

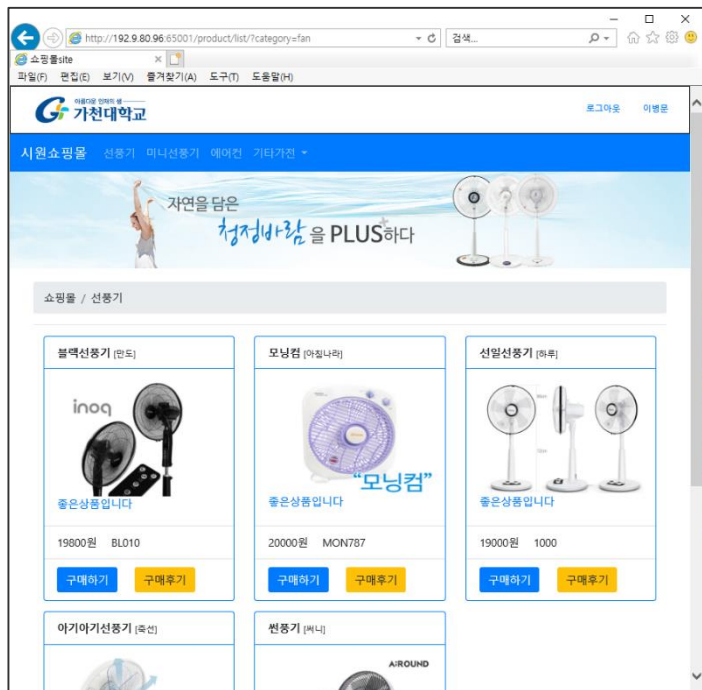
## ■ 서비스 개요

- 오픈소스 기반의 (데모용 냉방가전) 쇼핑몰 서비스 개발

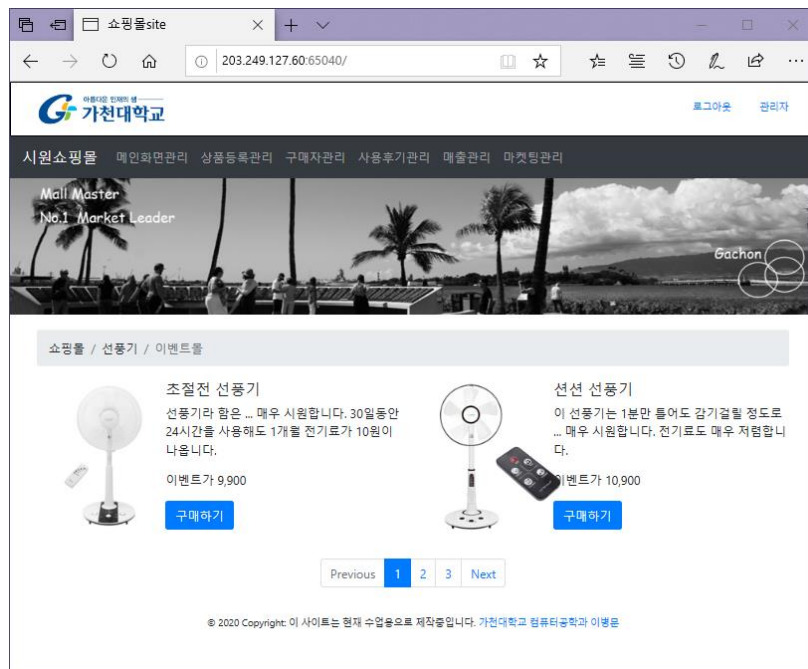


## ■ 서비스 개발 프로세스

- 오픈소스 기반의 (데모용 냉방가전) 쇼핑몰 서비스 개발

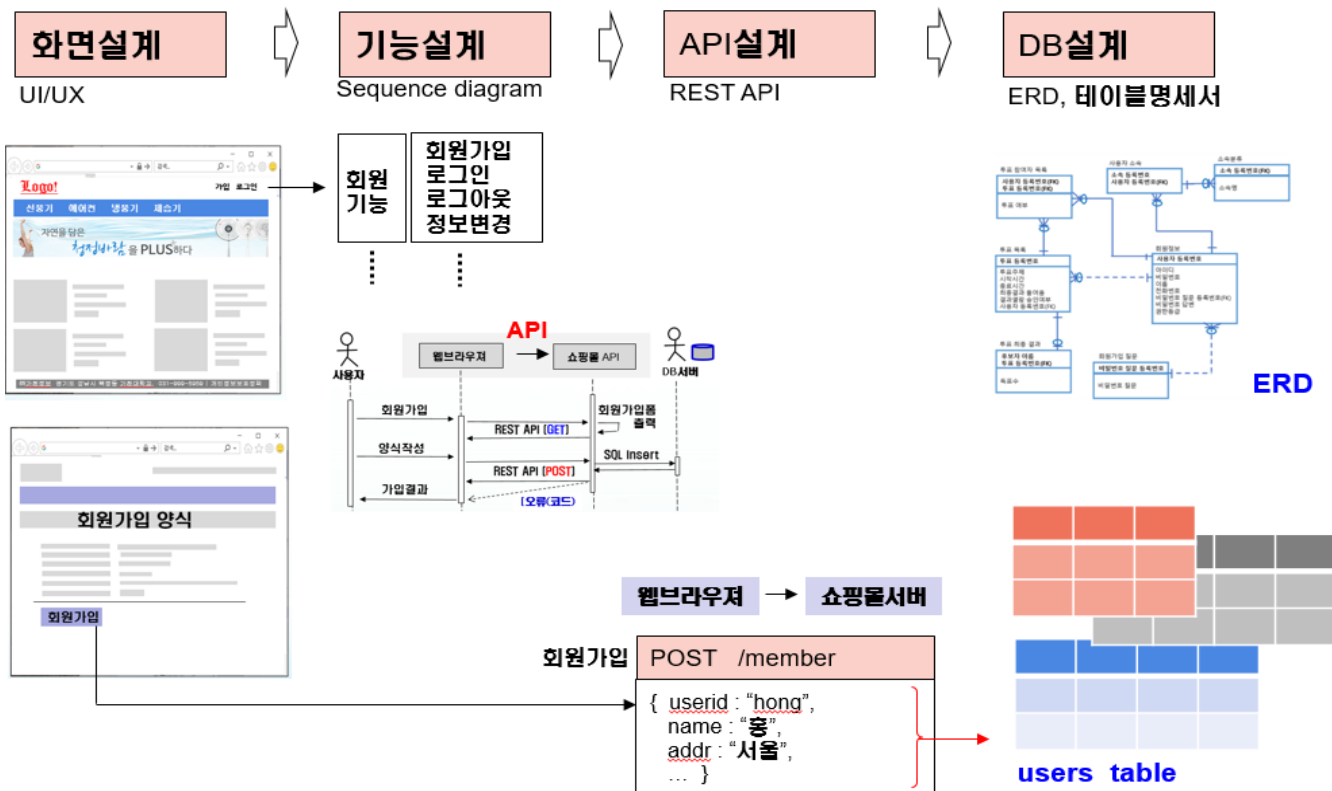


구매자서비스

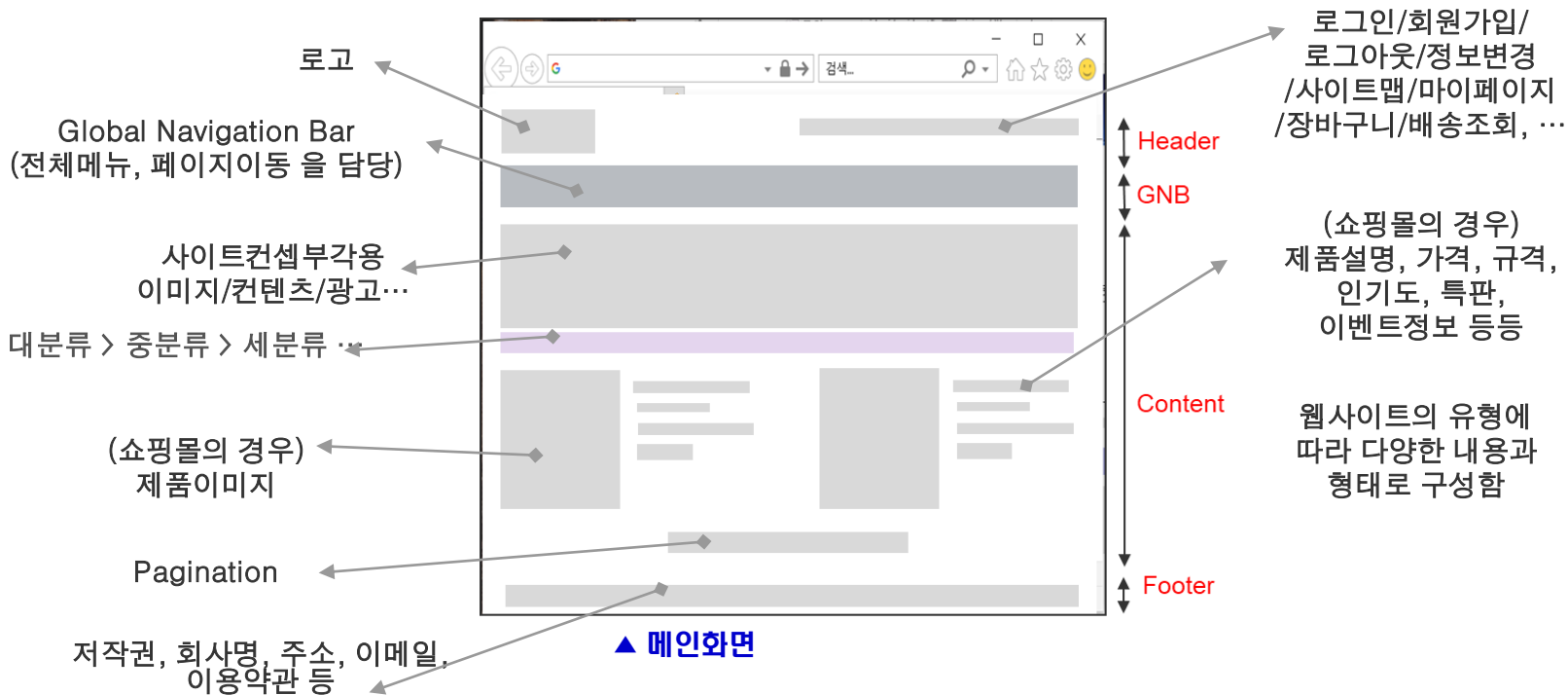
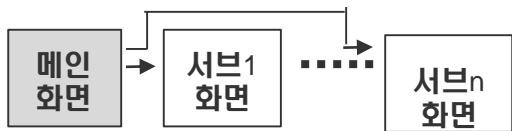


관리자서비스

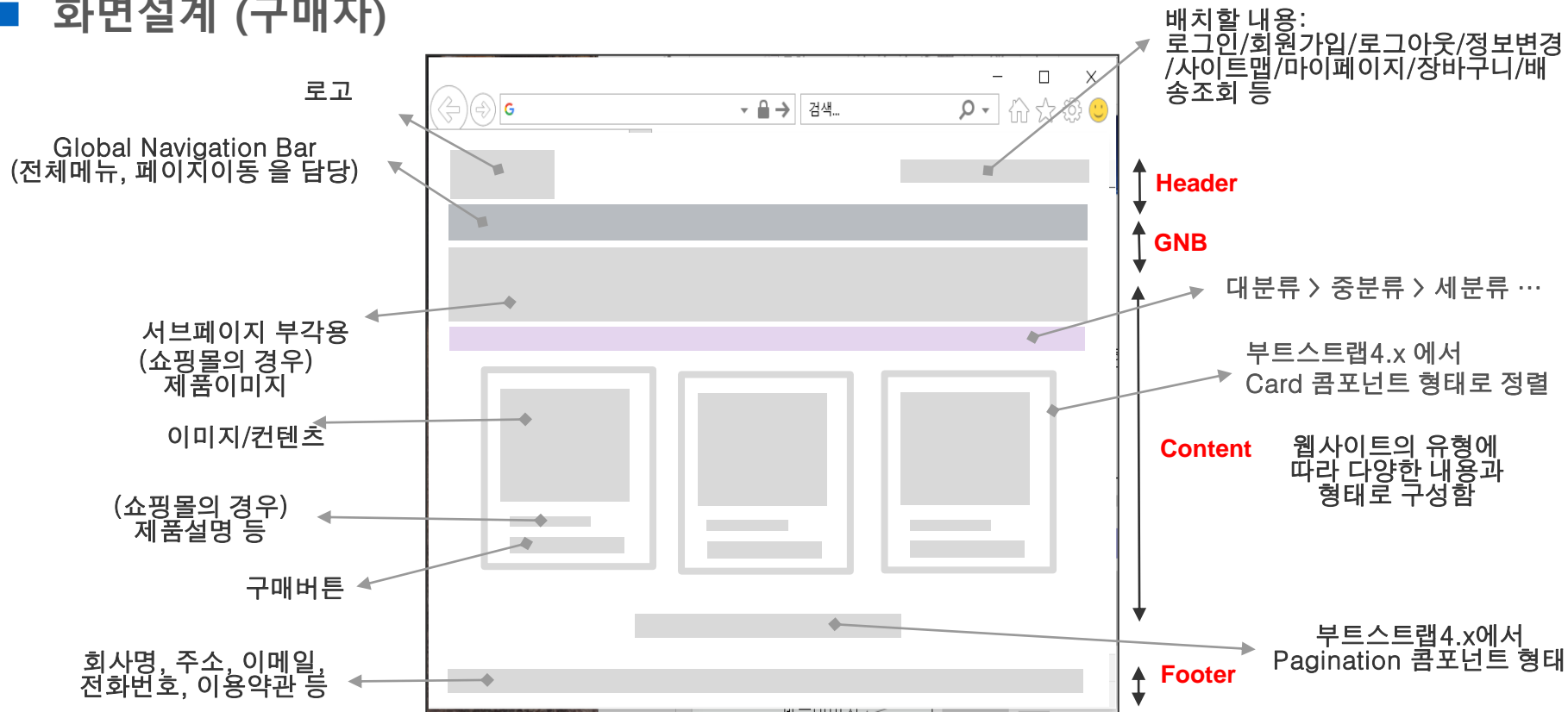
## ■ 서비스 설계



## ■ 화면설계 (공통)



## ■ 화면설계 (구매자)



▲ 구매자 서브화면

## ■ 화면설계 (관리자)

로그아웃 / 암호변경/마이페이지 등

관리자 전용메뉴

- 메인화면관리
- 제품등록관리
- 사용자관리
- 게시판관리
- 데이터백업관리, ...)

로고

(관리자 컨셉용)이미지

리스트명령메뉴  
(등록, 수정, 삭제)

회사명, 주소, 이메일,  
전화번호, 이용약관 등



Header

GNB

대분류 > 중분류 > 세분류 ...

리스트(목록)

Content

Pagination

Footer

▲ 관리자 서버화면



## ■ 기능설계

- 사용자별 기능을 정의한다 (아래예시)

### ○ 사용자 기능

역할이 서로 다른 사용자  
(예, 구매자 vs 판매자, ...)

주요기능	요구기능	사용자1	사용자2
메인 화면	헤더	O	O
	GNB	O	O
	Content	O	O
	Footer	O	O
회원 기능	회원가입	X	O
	로그인	X	O
	로그아웃	X	O
	정보변경	O	X

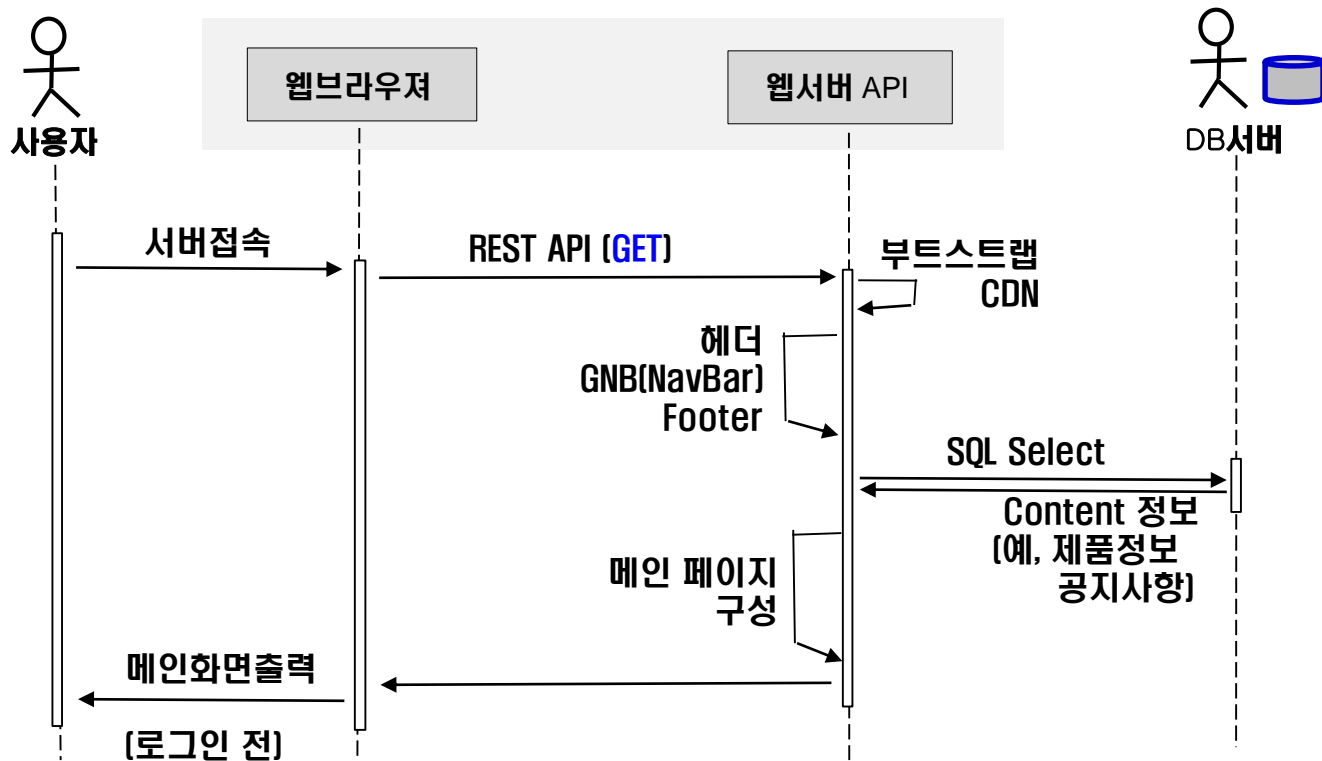
### ○ 관리자 기능

주요기능	요구기능
메인 화면	헤더
	GNB
	Content
	Footer
회원 관리	구매자목록 승인처리 강제탈퇴

같은 방식으로 모든기능을 정의한다

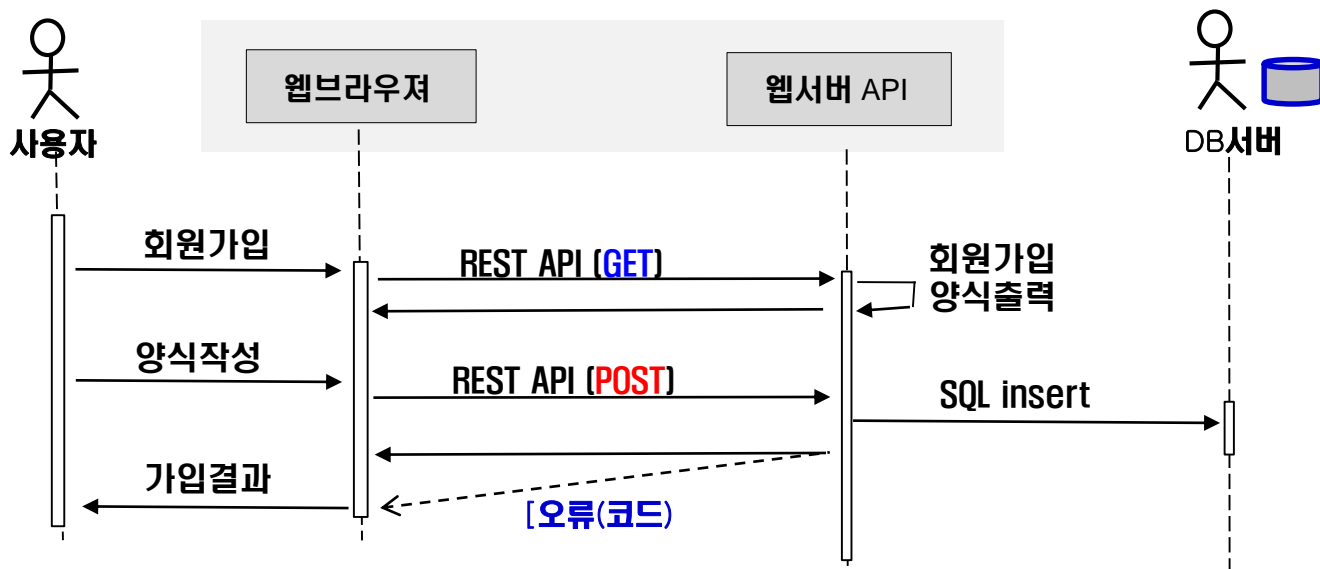
## ■ 기능설계

### ■ 메인화면접속



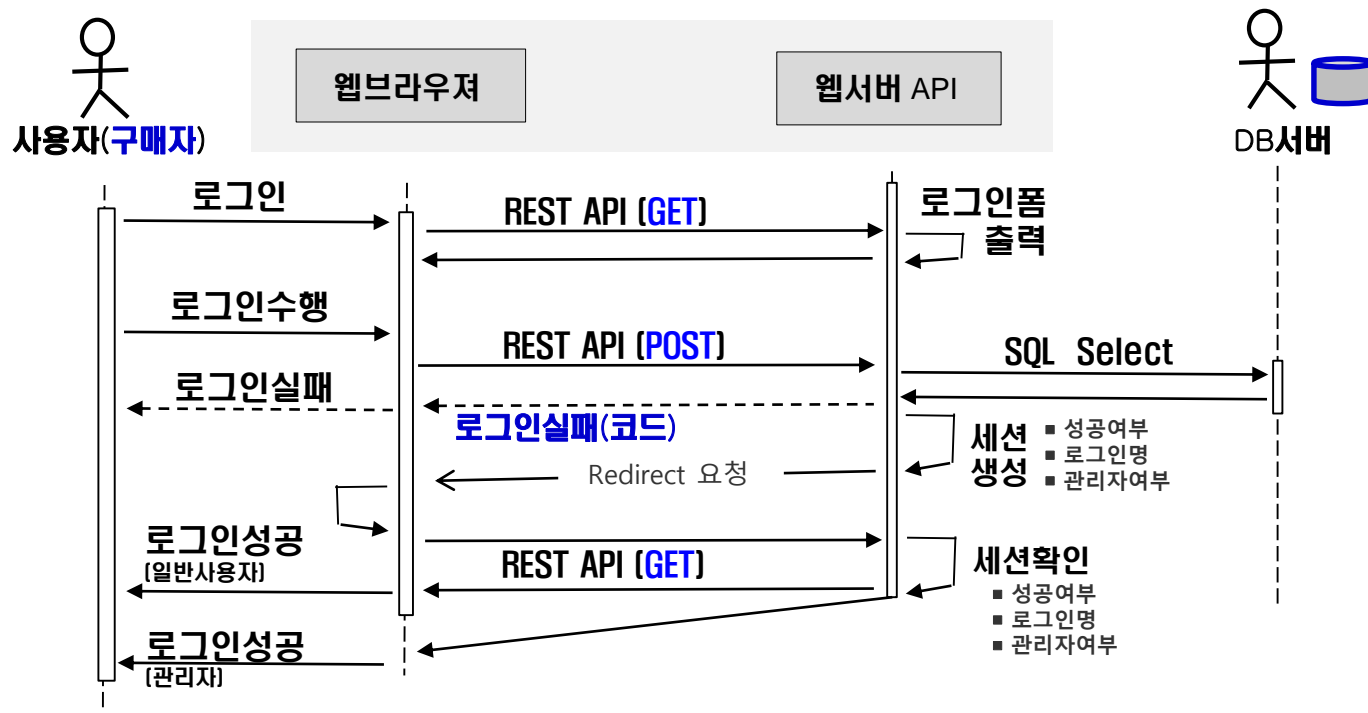
## ■ 기능설계

- 회원기능 (회원가입기능)



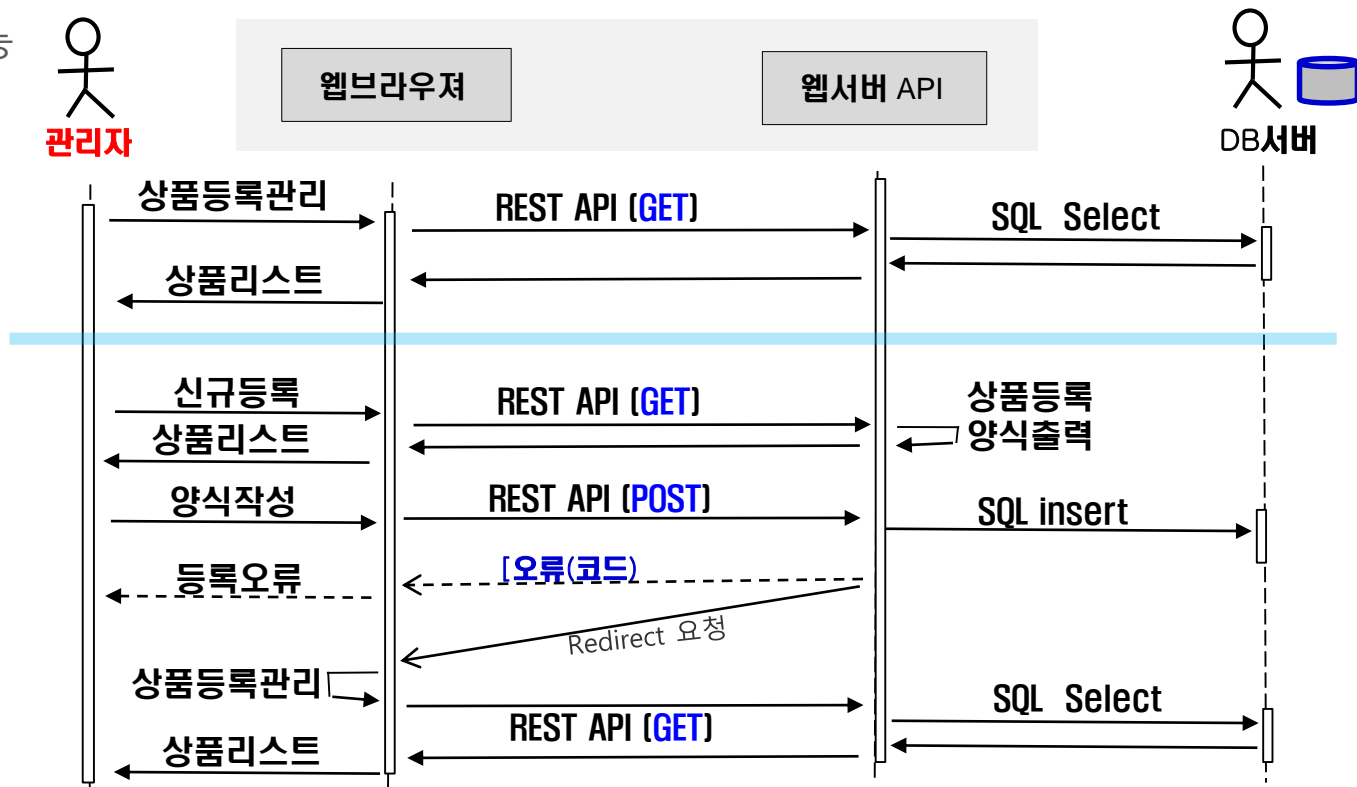
## ■ 기능설계

- 회원기능 (로그인기능)



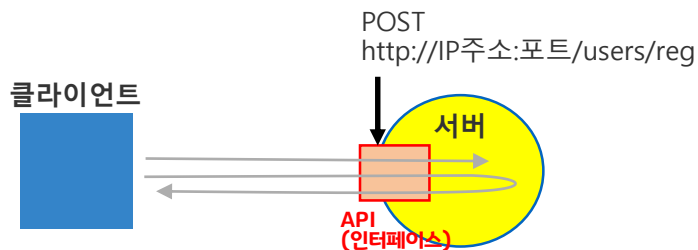
## ■ 기능설계

- 상품등록관리기능  
(관리자기능)



## ■ API설계 (REST API)

- POST
- GET
- PUT
- DELETE



### 회원가입

POST /users/reg

웹브라우저

→

회원가입

→ (IN)

쇼핑몰 서버

←

가입결과

← (OUT)

Parameter

속성	전송방향		Type	Description
	IN	OUT		
userid	O		string	사용자 아이디
pwd	O		string	비밀번호
uname	O		string	이름
status		O	int	200: 가입성공 561: 입력데이터없음 562: 가입실패

전송방향 : IN

{

"userid" : "201912345",

"pwd" : "passwd1234",

"uname" : "마마무",

}

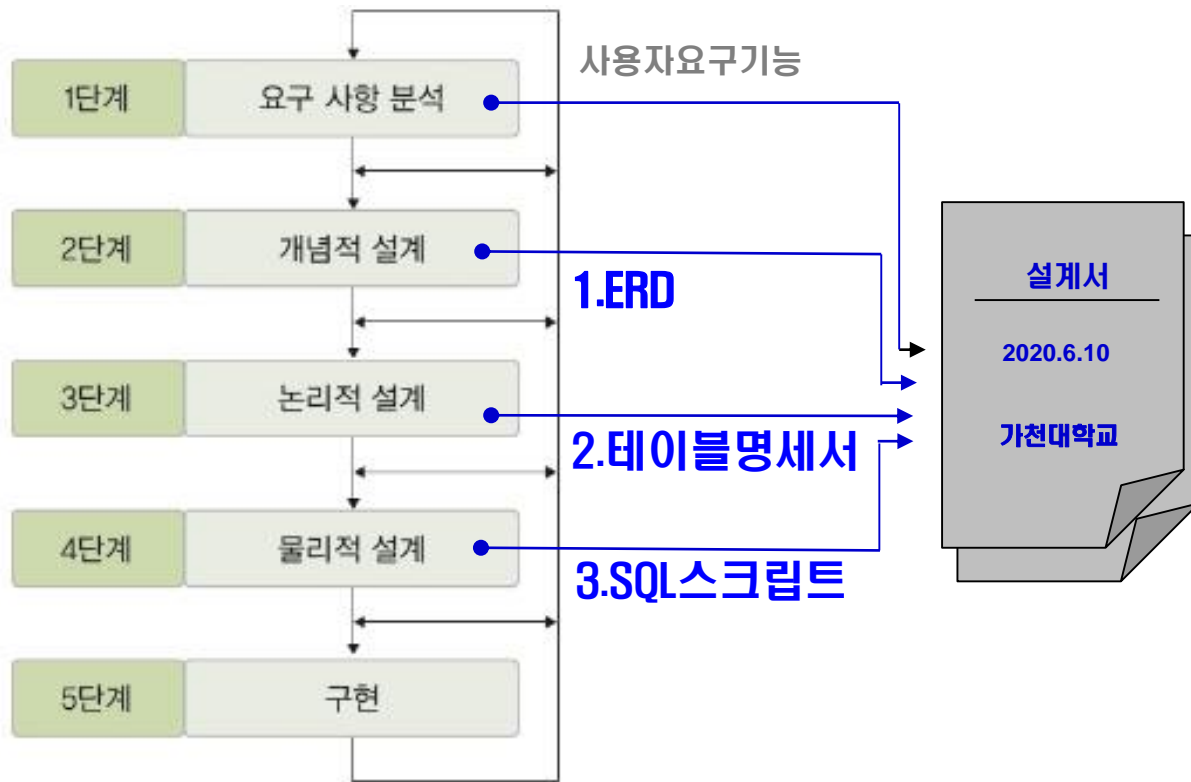
전송방향 : OUT

{

"status" : 200

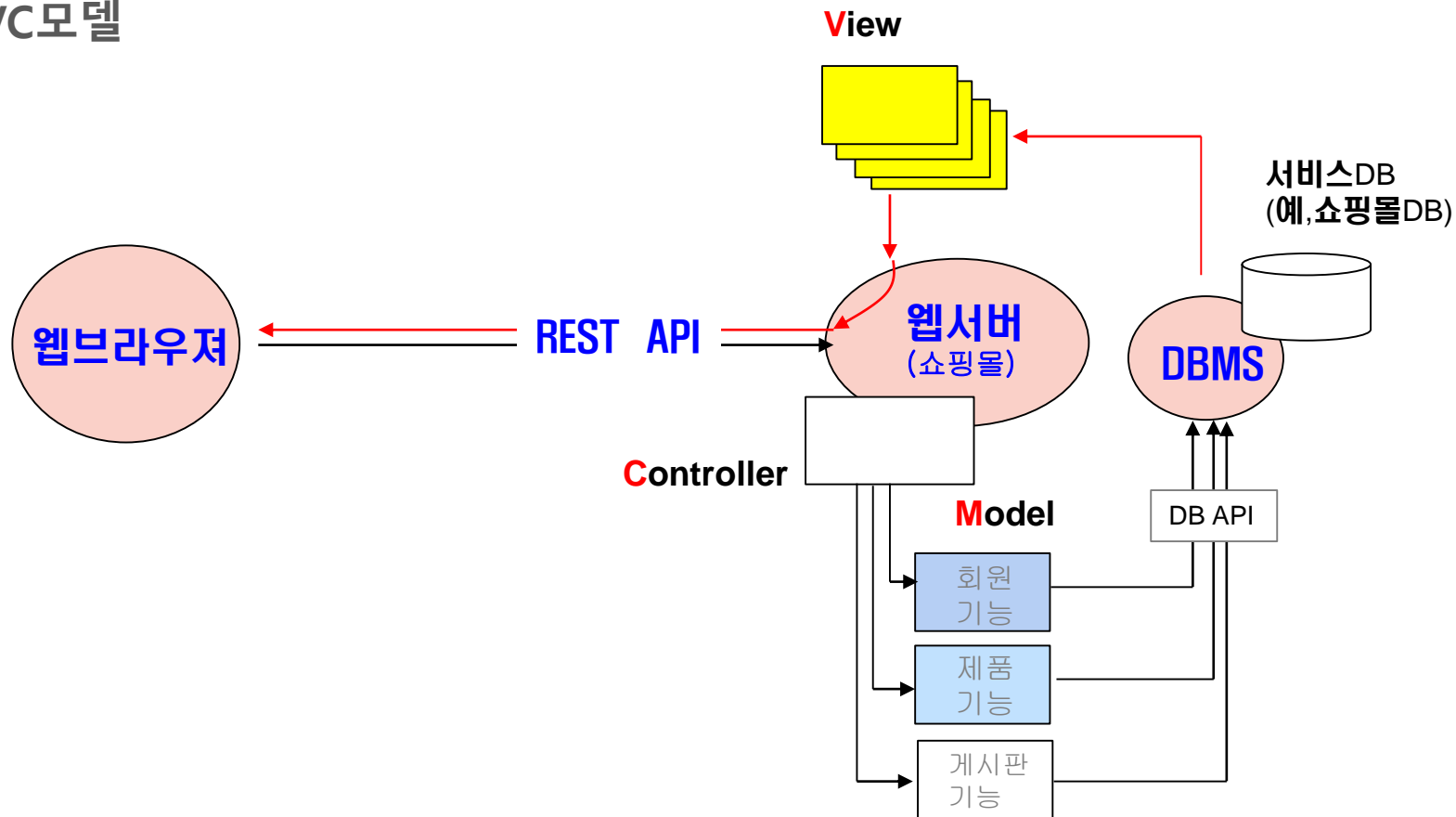
}

## ■ DB설계



데이터베이스 설계의 과정

## ■ MVC모델





### ■ 서비스 개발

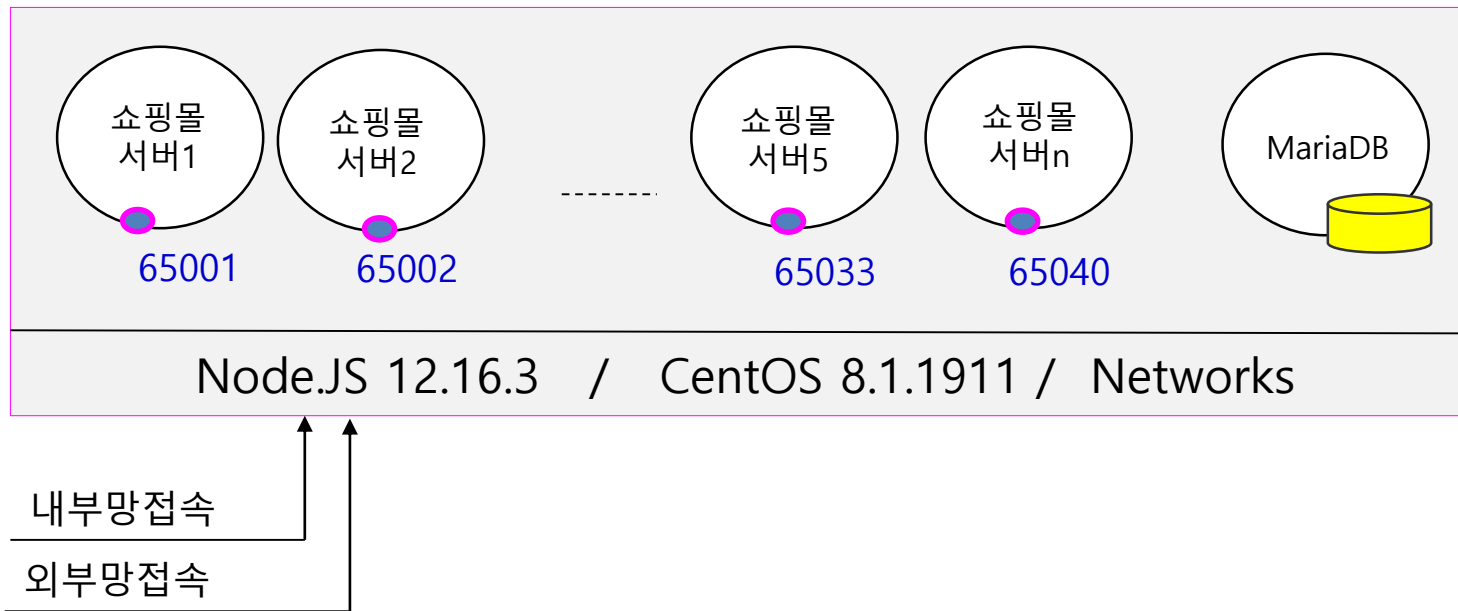
#### ■ 목차

- 01. 서비스 소개(쇼핑몰 웹사이트 구축)
- 02. 개발환경 구축 (Node.js/Express, Bootstrap, jQuery, Atom)
- 03. 개발사례 분석(샘플기능/코드)
- 04. 프로젝트 미션(서비스 완성)

### ■ 개발환경

- 클라이언트/서버 환경 구조

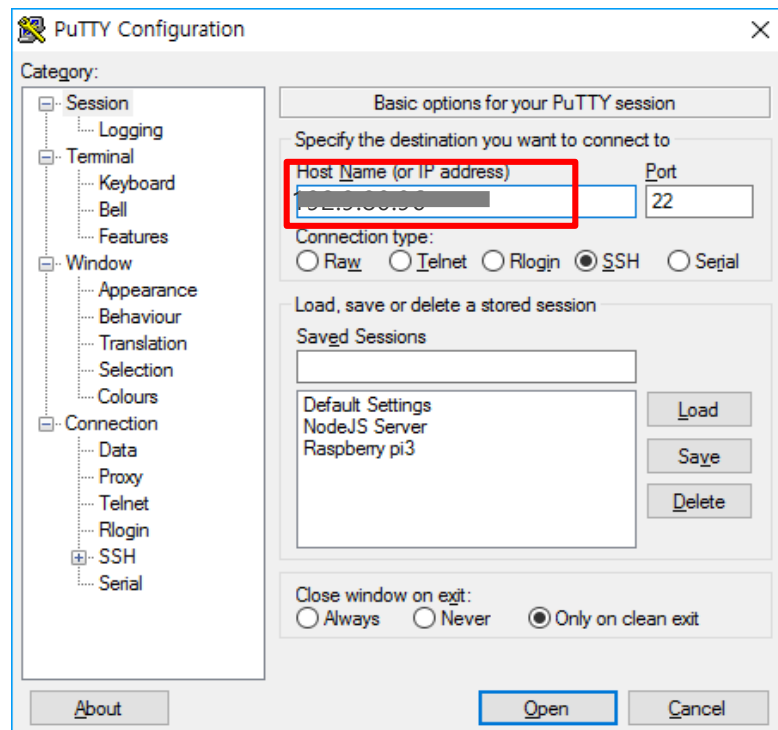
#### 클라우드서버



### ■ 개발환경

#### ■ 클라우드서버접속

- 내부 IP : 000.000.000.000
- 외부IP : XXX.XXX.XXX.XXX
- 방화벽 오픈 포트 :
  - ssh(22)
  - ftp(21)
  - 개인별(전용계정 및 포트)



### ■ Node.JS/Express 설치

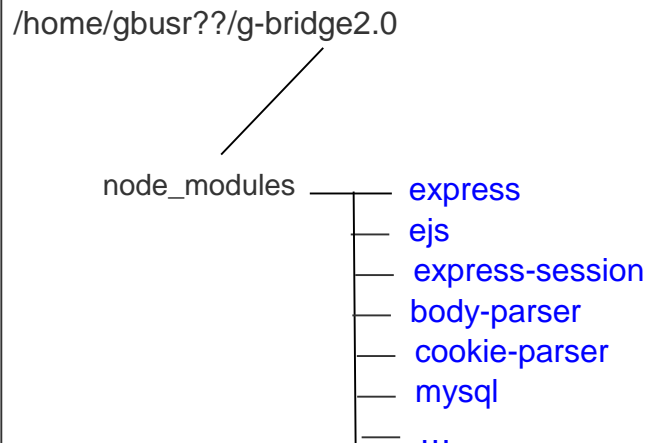
- Linux 환경에서 설치

### ■ 외부모듈 확장설치

```
$ cd g-bridge
$ pwd
/home/gbusr??/g-bridge2.0

$ npm install ejs
$ npm install express
$ npm install cookie-parser
$ npm install body-parser
$ npm install express-session
$ npm install mysql
....
```

Local설치 디렉터리 위치



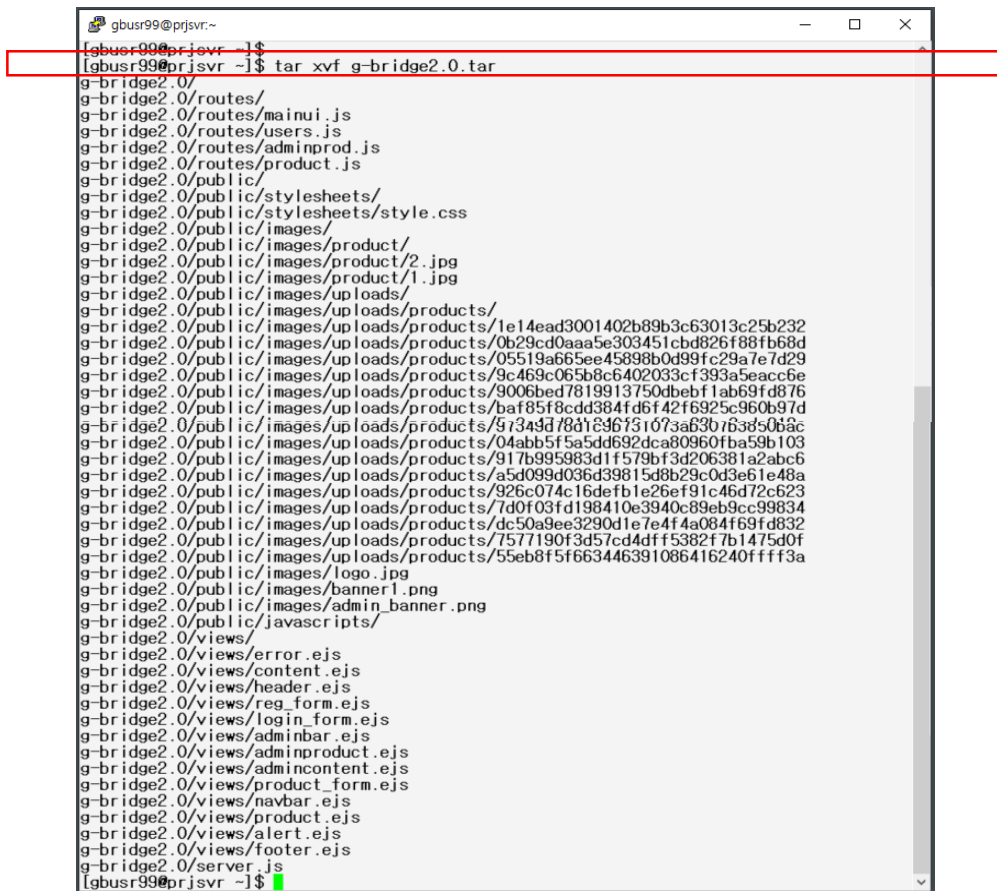
## ■ 외부모듈 확장설치

- 소스환경에서 설치

```
gbusr99@prjsvr:~  
[gbusr99@prjsvr ~]$ ftp 192.9.80.96  
Connected to 192.9.80.96 (192.9.80.96).  
220 (vsFTPd 3.0.3)  
Name (192.9.80.96:gbusr99): anonymous  
331 Please specify the password.  
Password:  
230 Login successful.  
Remote system type is UNIX.  
Using binary mode to transfer files.  
ftp> cd pub  
250 Directory successfully changed.  
ftp> ls  
227 Entering Passive Mode (192,9,80,96,188,41).  
150 Here comes the directory listing.  
-r--r--r-- 1 0 0 1986560 May 31 12:49 g-bridge2.0.tar  
-r--r--r-- 1 0 0 5345280 May 31 12:49 node_modules.tar  
226 Directory send OK.  
ftp> mget g* n*  
mget g-bridge2.0.tar? y  
227 Entering Passive Mode (192,9,80,96,59,67).  
150 Opening BINARY mode data connection for g-bridge2.0.tar (1986560 bytes).  
226 Transfer complete.  
1986560 bytes received in 0.00147 secs (1348648.96 Kbytes/sec)  
mget node_modules.tar? y  
227 Entering Passive Mode (192,9,80,96,95,95).  
150 Opening BINARY mode data connection for node_modules.tar (5345280 bytes).  
226 Transfer complete.  
5345280 bytes received in 0.00391 secs (1367428.99 Kbytes/sec)  
ftp> quit  
221 Goodbye.  
[gbusr99@prjsvr ~]$ ls  
day1 g-bridge2.0.tar node_modules.tar  
[gbusr99@prjsvr ~]$
```

## ■ 외부모듈 확장설치

- 소스환경에서 설치



```
gbusr99@prjsvr:~$  
gbusr99@prjsvr:~$ tar xvf g-bridge2.0.tar  
g-bridge2.0/  
g-bridge2.0/routes/  
g-bridge2.0/routes/mainui.js  
g-bridge2.0/routes/users.js  
g-bridge2.0/routes/adminprod.js  
g-bridge2.0/routes/product.js  
g-bridge2.0/public/  
g-bridge2.0/public/stylesheets/  
g-bridge2.0/public/stylesheets/style.css  
g-bridge2.0/public/images/  
g-bridge2.0/public/images/product/  
g-bridge2.0/public/images/product/2.jpg  
g-bridge2.0/public/images/product/1.jpg  
g-bridge2.0/public/images/uploads/  
g-bridge2.0/public/images/uploads/products/  
g-bridge2.0/public/images/uploads/products/1e14ead3001402b89b3c63013c25b232  
g-bridge2.0/public/images/uploads/products/0b29cd0aaa5e303451cbd826f88fb68d  
g-bridge2.0/public/images/uploads/products/05519a665ee45898b0d99cf29a7e7d29  
g-bridge2.0/public/images/uploads/products/9c469c065b8c6402033cf393a5eacc6e  
g-bridge2.0/public/images/uploads/products/9006bed7819913750dbef1ab69fd876  
g-bridge2.0/public/images/uploads/products/baf85f8cdd384fd6f42f6925c960b97d  
g-bridge2.0/public/images/uploads/products/91349d78d1c36f31073a63b763d50b6ac  
g-bridge2.0/public/images/uploads/products/04abb5f5a5dd692dca80960fba59b103  
g-bridge2.0/public/images/uploads/products/917b995983d1f579bf3d206381a2abc6  
g-bridge2.0/public/images/uploads/products/a5d099d036d39815d8b29c0d3e61e48a  
g-bridge2.0/public/images/uploads/products/926c074c16defb1e26ef91c46d72c623  
g-bridge2.0/public/images/uploads/products/7d0f03fd198410e3940c89eb9cc99834  
g-bridge2.0/public/images/uploads/products/dc50a9ee3290d1e7e4f4a084f69fd832  
g-bridge2.0/public/images/uploads/products/7577190f3d57cd4dff5382f7b1475d0f  
g-bridge2.0/public/images/uploads/products/55eb8f5f663446391086416240ffff3a  
g-bridge2.0/public/images/logo.jpg  
g-bridge2.0/public/images/banner1.png  
g-bridge2.0/public/images/admin_banner.png  
g-bridge2.0/public/javascripts/  
g-bridge2.0/views/  
g-bridge2.0/views/error.ejs  
g-bridge2.0/views/content.ejs  
g-bridge2.0/views/header.ejs  
g-bridge2.0/views/reg_form.ejs  
g-bridge2.0/views/login_form.ejs  
g-bridge2.0/views/adminbar.ejs  
g-bridge2.0/views/adminproduct.ejs  
g-bridge2.0/views/admincontent.ejs  
g-bridge2.0/views/product_form.ejs  
g-bridge2.0/views/navbar.ejs  
g-bridge2.0/views/product.ejs  
g-bridge2.0/views/alert.ejs  
g-bridge2.0/views/footer.ejs  
g-bridge2.0/server.js  
gbusr99@prjsvr:~$
```

## ■ 외부모듈 확장설치

- 소스환경에서 설치

```

gbusr99@prjsvr:~$ ls
day1 g-bridge2.0 g-bridge2.0.tar node_modules.tar
gbusr99@prjsvr:~$ tar xvf node_modules.tar
node_modules/
node_modules/async/
node_modules/async/package.json
node_modules/async/README.md
node_modules/async/LICENSE
node_modules/async/component.json
node_modules/async/.travis.yml
node_modules/async/bower.json
node_modules/async/lib/
node_modules/async/lib/async.js
node_modules/async/support/
node_modules/async/support/sync-package-managers.js
node_modules/balanced-match/
node_modules/balanced-match/package.json
node_modules/balanced-match/.npmignore
node_modules/balanced-match/README.md
node_modules/balanced-match/index.js
node_modules/balanced-match/LICENSE.md
node_modules/color-name/
node_modules/color-name/package.json
node_modules/color-name/.npmignore
node_modules/color-name/README.md
node_modules/multer/lib/remove-uploaded-files.js
node_modules/multer/storage/
node_modules/multer/storage/disk.js
node_modules/multer/storage/memory.js
gbusr99@prjsvr:~$
  
```

```

[gbusr99@prjsvr ~]$ cd g-bridge2.1
[gbusr99@prjsvr g-bridge2.1]$ ls
public routes server.js views
[gbusr99@prjsvr g-bridge2.1]$ vi server.js
[gbusr99@prjsvr g-bridge2.1]$
  
```

반드시, 각 개인별로 포트주소를  
프로그램코드에서 다르게 변경하여야 함!  
그래야, 충돌이 일어나지 않음!!  
즉, 아직 실행하면 안됨!

```

const adminuser = require('./routes/admin');
const product = require('./routes/product');
^M
// 쇼핑물 전용 PORT주소 설정
const PORT = 65040;

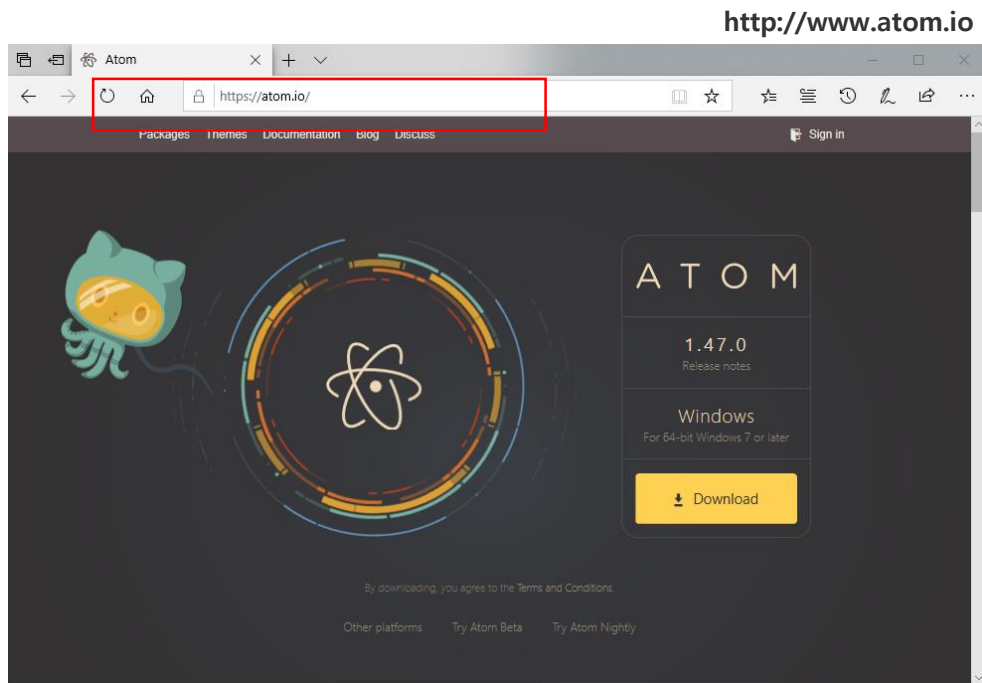
// 실행환경 설정부분
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'ejs');
  
```

```

[gbusr99@prjsvr g-bridge2.1]$
[gbusr99@prjsvr g-bridge2.1]$ node server.js
서버실행 : http://203.249.127.60:65040
서버실행 : http://192.9.80.96:65040
  
```

### ■ Atom 에디터 설치

- Windows 10 환경에서 설치

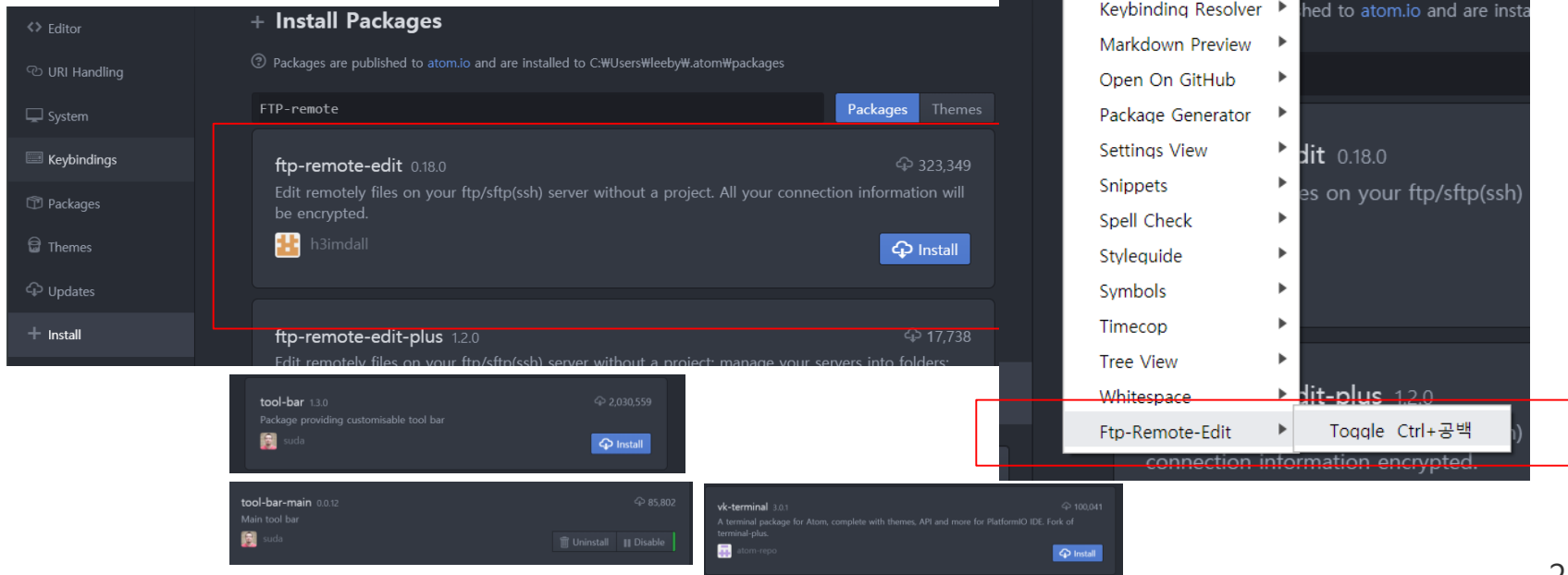




### ■ Atom 에디터 설치

- Atom 추가패키지 설치

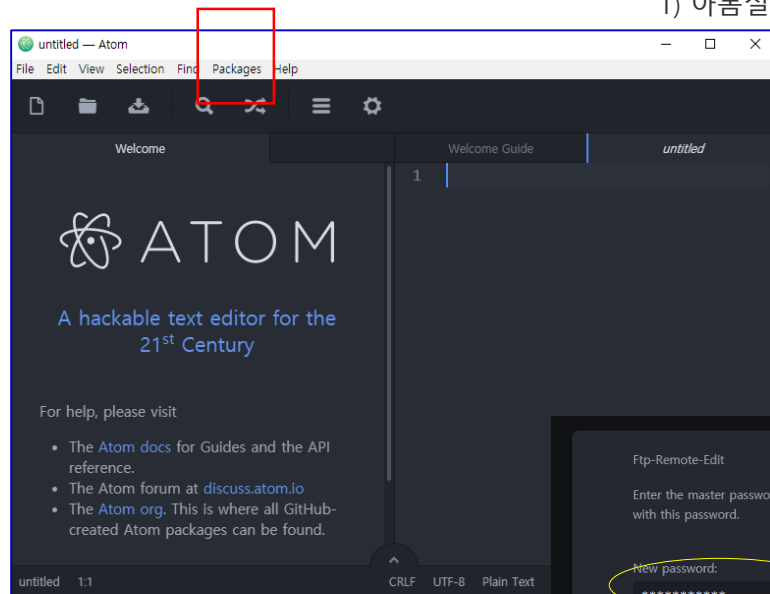
FTP-remote-edit, tool-bar, too-bar-main, vk-terminal 등  
설치: **^+, > Install >**



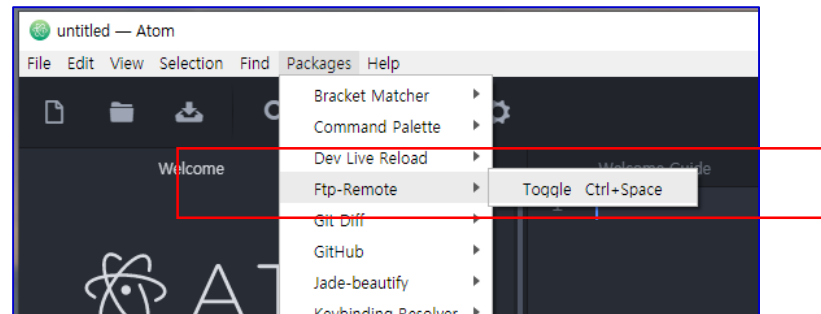
### ■ Atom 활용

#### ■ 프로그래밍

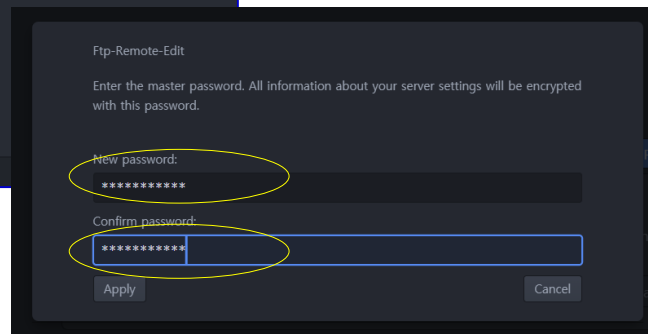
1) 아톰실행



2) FTP서버접속

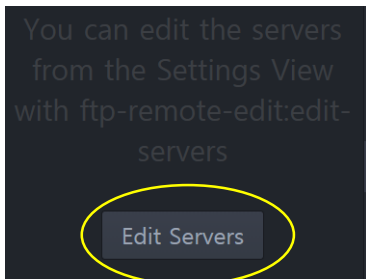


3) FTP세션 초기등록(암호설정)



## ■ Atom 활용

### ■ 프로그래밍



4) 서버 세션정보 등록/수정

5) 서버 세션정보 등록하여 저장함!

Ftp-Remote-Edit Server Settings

You can edit each connection at the time. All changes will only be saved by pushing the save button.

G브릿지서버 New Delete Duplicate Test

The hostname or IP address of the server. Port

XXX.XXX.XXX.XXX 21

Protocol

FTP - File Transfer Protocol

Encryption

Only use plain FTP (insecure)

Logon Type

Username / Password

Username for authentication.

gbusr99

Password/Passphrase for authentication.

\*\*\*\*\*

Initial Directory.

/home/gbusr99

Save Import Cancel

각자 입력!!

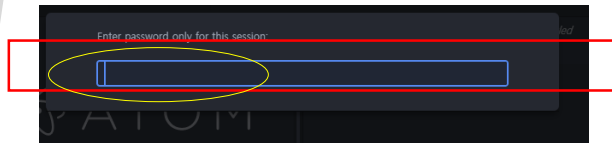
1) 계정ID

2) 암호

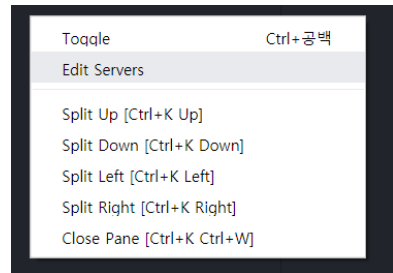
3) 홈디렉터리

1) 아톰실행후, ^ + <space>

암호입력하면, G브릿지 서버에 접속됨



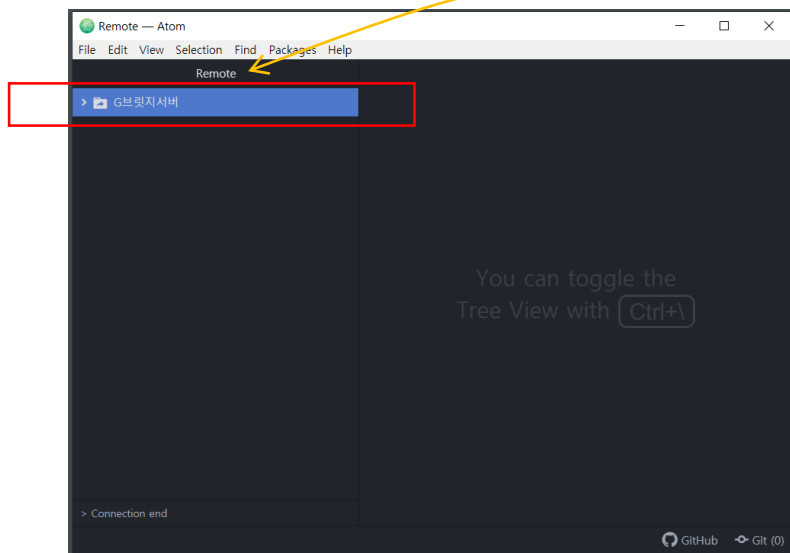
2) 세션을 다시 수정하려면 (마우스 우측버튼)



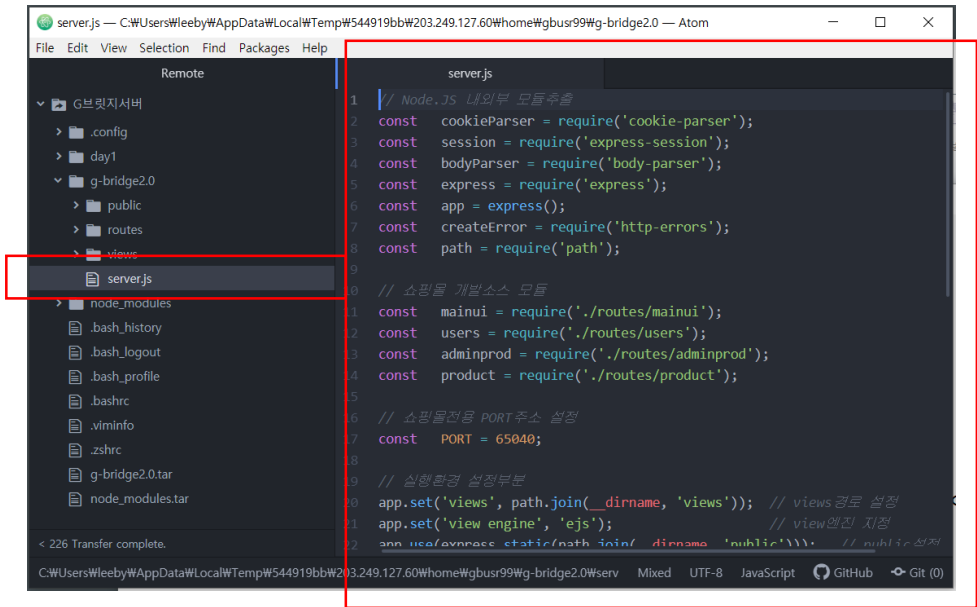
### Atom 활용

- 프로그래밍

FTP서버 접속세션



서버소스 디렉터리에 접속



## Atom 활용

### 프로그래밍

(서버로) 원격저장

The screenshot shows the Atom text editor with a project named 'g-bridge2.0'. The left sidebar displays the project structure, and the main editor shows the code for 'server.js'.

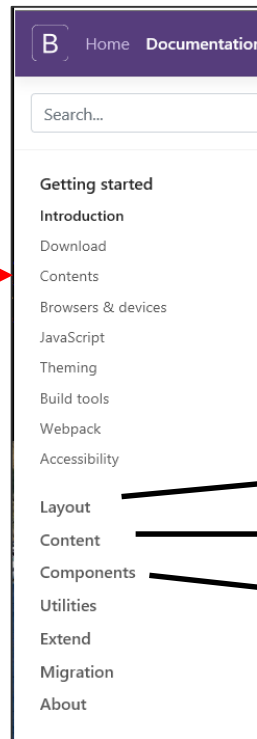
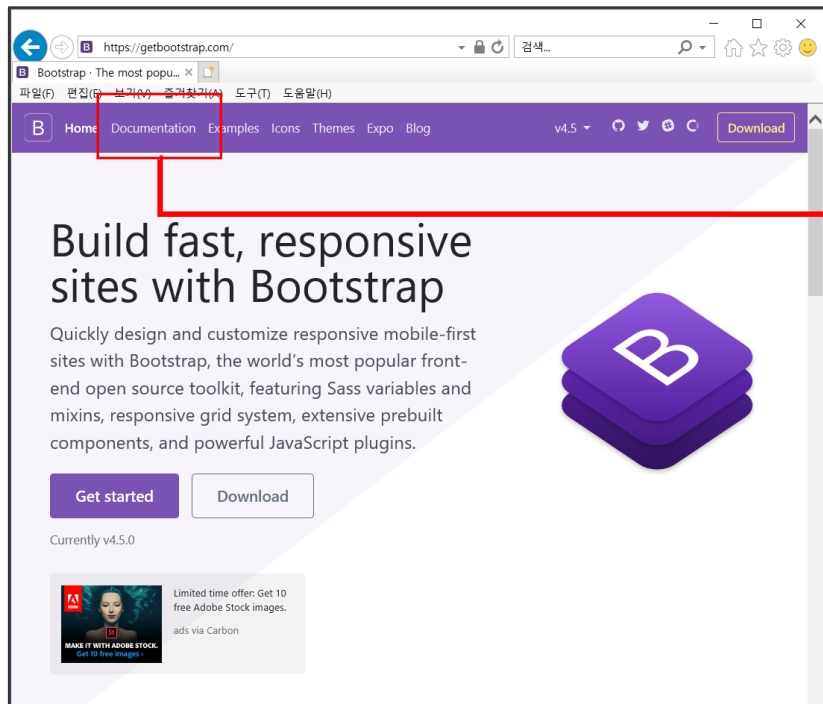
**Annotations:**

- 브라우저 Resources:** Points to the 'public' folder in the project structure.
- 백엔드 서비스 기능:** Points to the 'routes' folder in the project structure.
- 브라우저 View (EJS페이지):** Points to the 'views' folder in the project structure.
- Main Controller:** Points to the 'server.js' file in the project structure.
- (서버로) 원격저장:** Points to the 'File' menu in the top toolbar.

**server.js Code:**

```
// Node.js 내부 모듈 추출
1 const cookieParser = require('cookie-parser');
2 const session = require('express-session');
3 const bodyParser = require('body-parser');
4 const express = require('express');
5 const app = express();
6 const createError = require('http-errors');
7 const path = require('path');
8
9 // 소정들 개발소스 모듈
10 const mainui = require('./routes/mainui');
11 const users = require('./routes/users');
12 const adminprod = require('./routes/adminprod');
13 const adminuser = require('./routes/adminuser');
14 const product = require('./routes/product');
15
16 // 소정들전용 PORT 주소 설정
17 const PORT = 65040;
18
19 // 실행환경 설정부분
20 app.set('views', path.join(__dirname, 'views')); // views 경로 설정
21 app.set('view engine', 'ejs'); // view 엔진 지정
22 app.use(express.static(path.join(__dirname, 'public'))); // public 설정
23 app.use(express.urlencoded({ extended: false }));
24 app.use(cookieParser());
25 app.use(bodyParser.urlencoded({ extended: false }));
26 app.use(express.json());
27 app.use(session({ key: 'sid',
28                   secret: 'secret key', // 세션id 암호화할때 사용
29                   resave: false, // 접속할때마다 id 부여금지
30                   saveUninitialized: true })); // 세션id 사용전에는 발급금지
31
32
```

<https://getbootstrap.com/>



페이지 구성

HTML 태그 요소들

\*, 예) Cards 타입, Pagination 타입, ...

[경기도형 대학생 취업브리지 사업]

# 차세대 웹모바일 플랫폼 기반 응용SW 개발 교육과정

12주차: Node.js 서비스 개발

## 이번 강좌 에서는...

**Node.JS 기반 백엔드 서비스개발**은 쇼핑몰 판매 서비스를 주제로 웹사이트를 개발하는 과정을 프로젝트 형식으로 살펴보며, 각 단계마다 필요한 기술과 해결을 위한 접근방식을 소개합니다.

특히 이 강의를 통해 **Node.JS, express 웹프레임워크, ejs 미들웨어, Bootstrap 4.x 기술을 활용하는 기술을 배우며, 소스코드 템플릿과 샘플을 직접 다루어 보다 빠른 개발능력을 키우는데 혁신적인 학습방법을** 제공합니다.



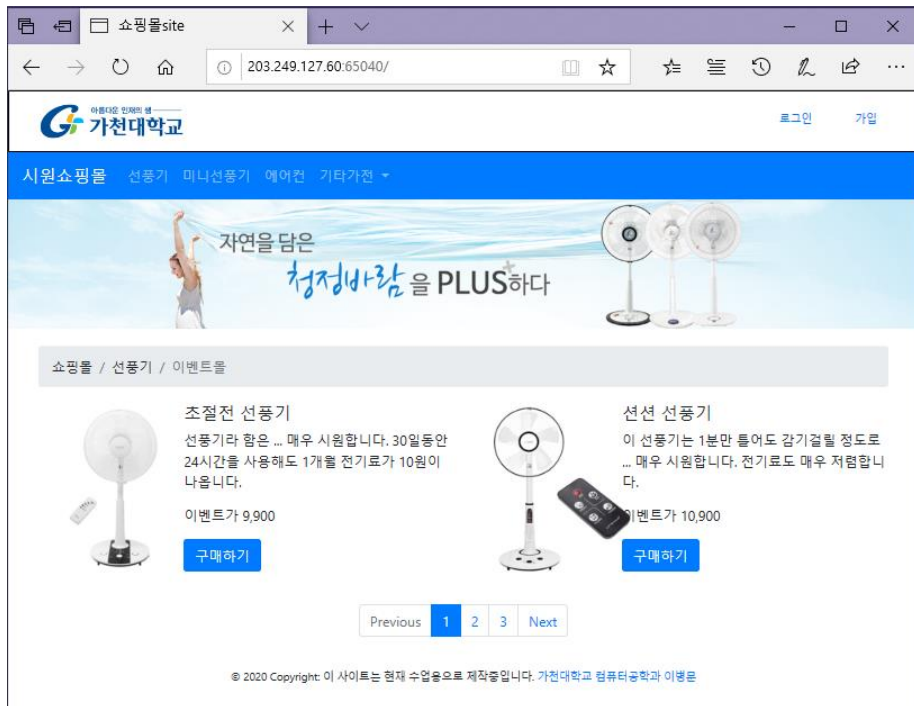
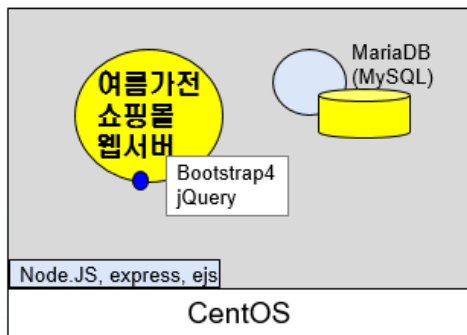
### ■ 목차

- 01. 개발사례 분석(샘플기능/코드)
- 02. 프로젝트 미션(서비스 완성)



## ■ 개발서비스

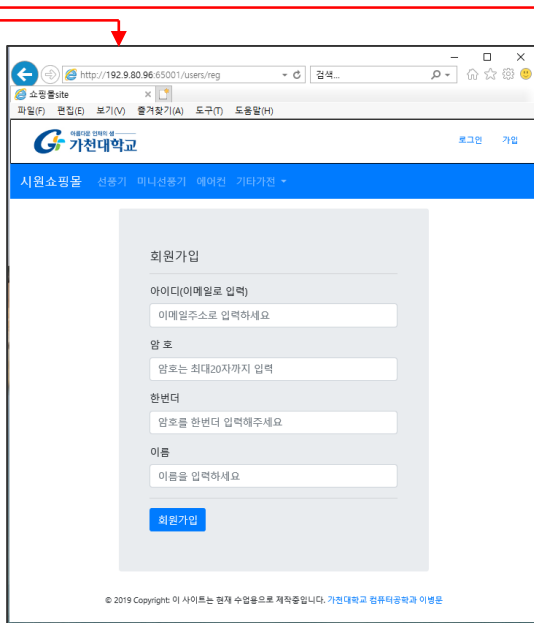
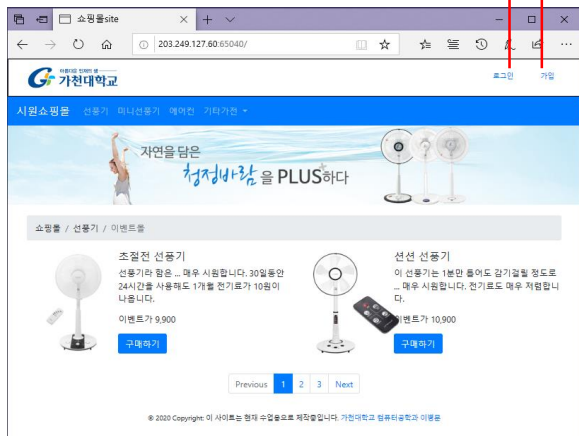
- 여름가전 쇼핑몰 서비스



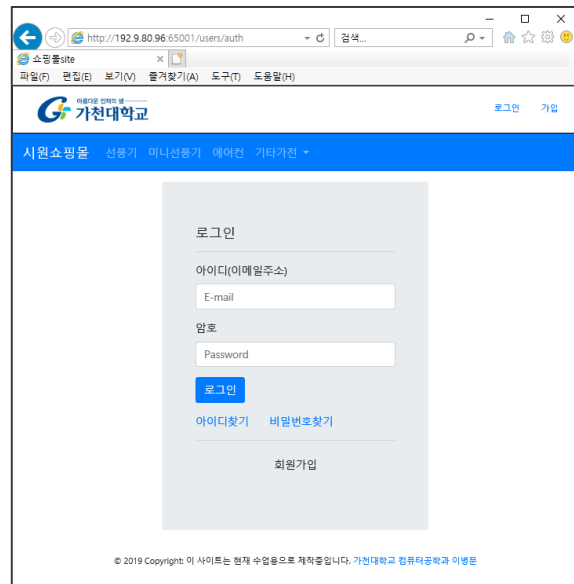
▲ (로그인 전) 메인화면

## ■ 개발서비스

### ■ 여름가전 쇼핑몰 서비스



▲ 회원가입화면

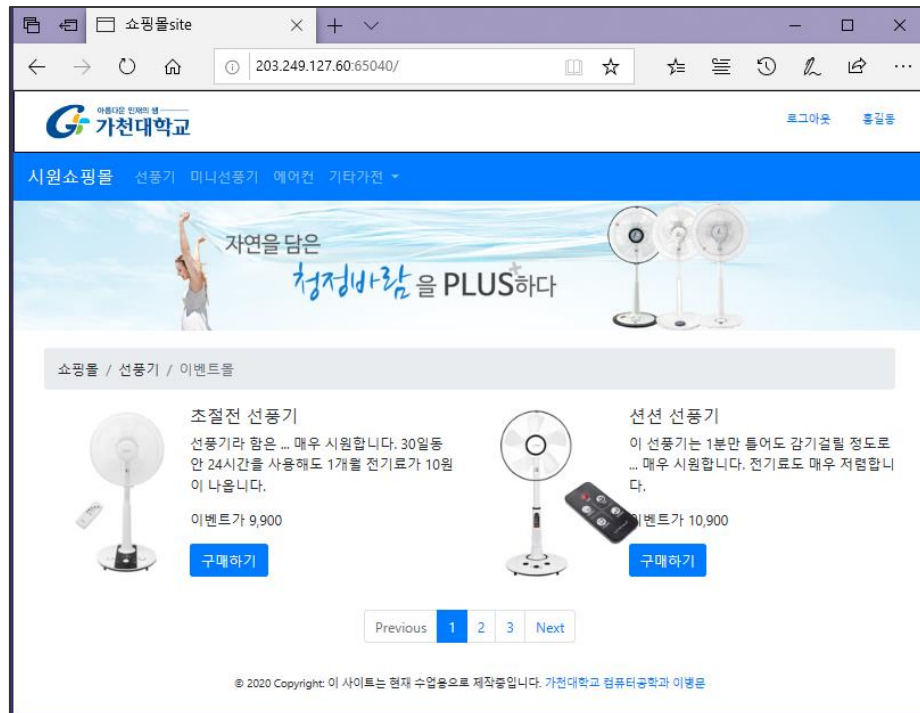


▲ 로그인화면

## ■ 개발서비스

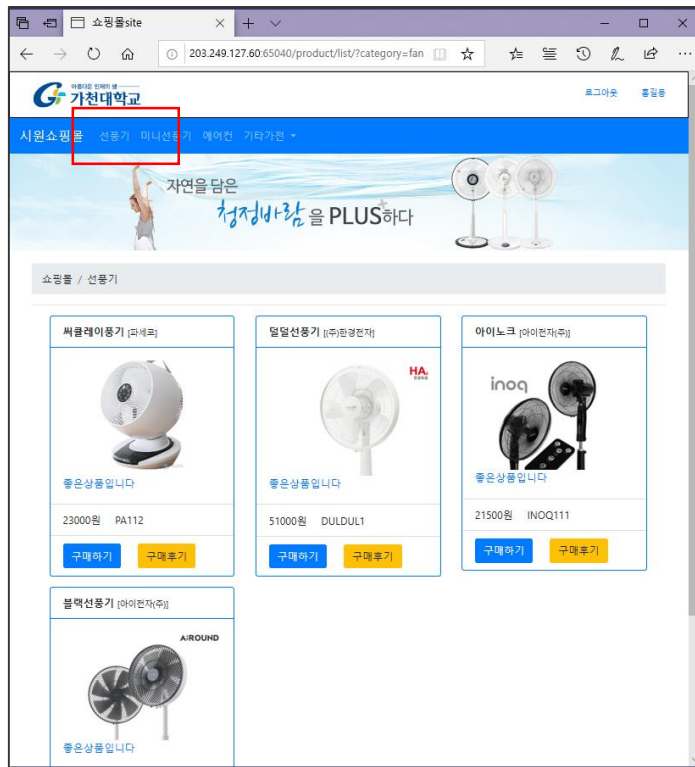
### ■ 여름가전 쇼핑몰 서비스

로그인후



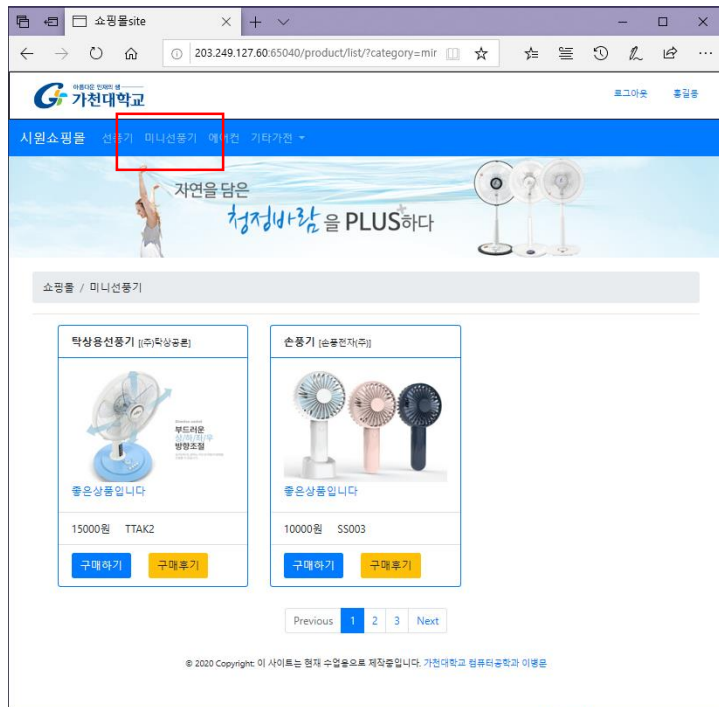
▲ (구매자로 로그인후 화면)

▶ “선풍기” 카테고리 선택후 화면 ▶

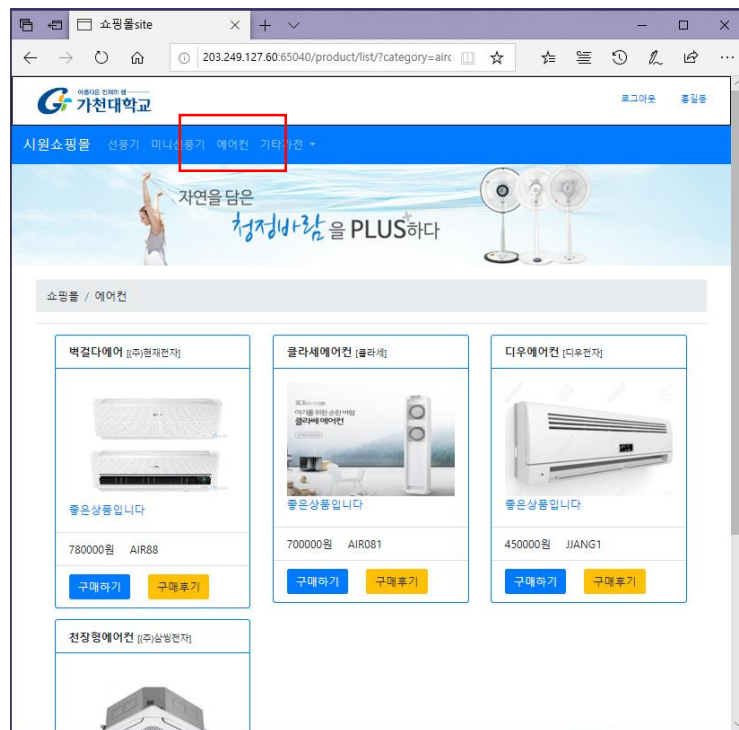


## ■ 개발서비스

### ■ 여름가전 쇼핑몰 서비스



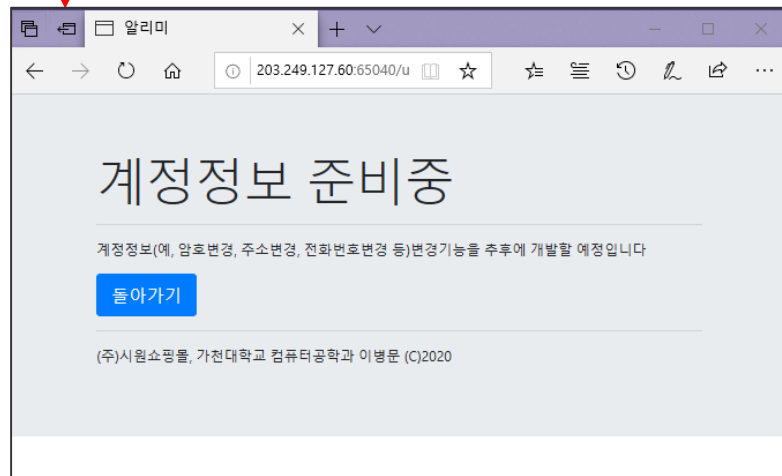
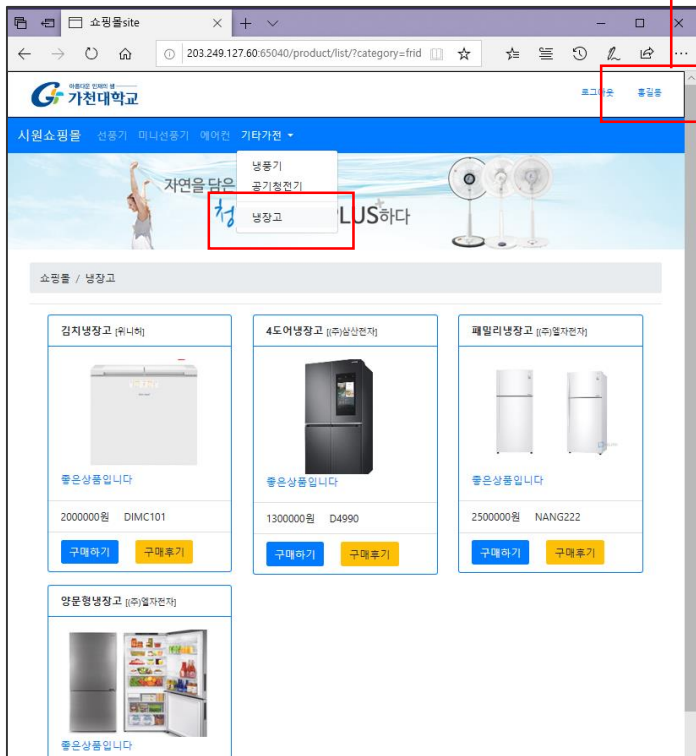
▲ “미니선풍기” 카테고리 선택후 화면



▲ “에어컨” 카테고리 선택후 화면

## ■ 개발서비스

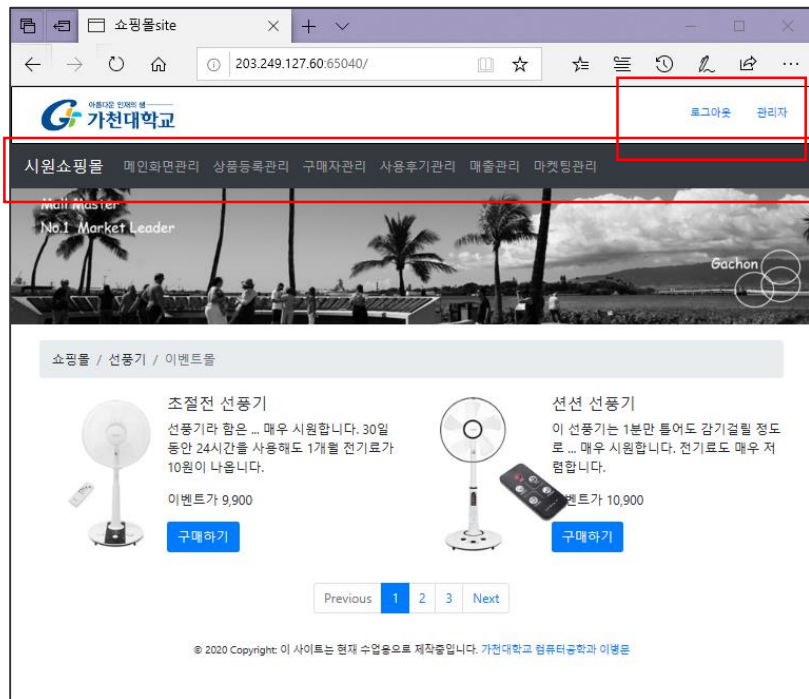
### ■ 여름가전 쇼핑몰 서비스



▲ (로그인 후) 이름선택후 “정보변경” 화면이 나와야 함.  
(현재는 준비중...)

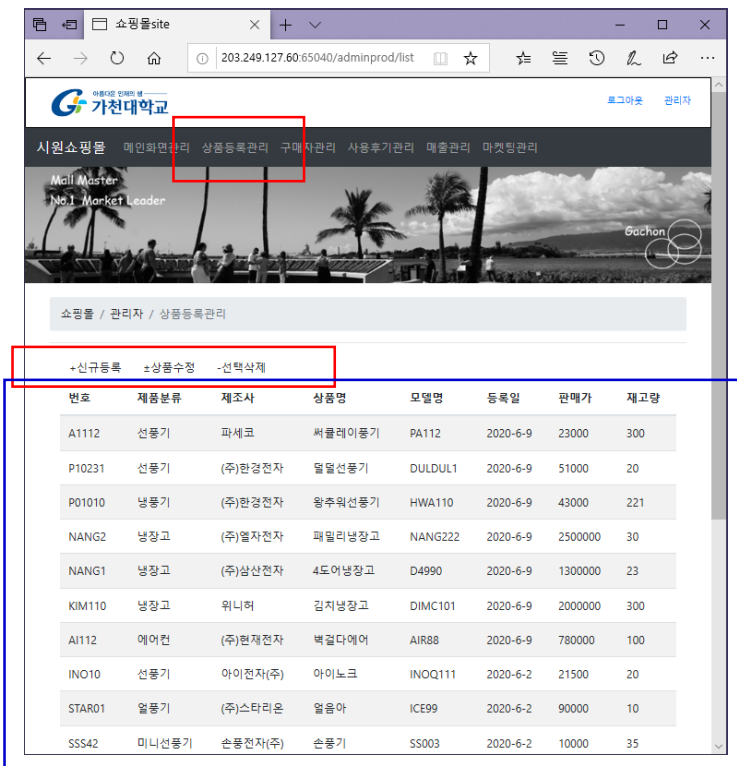
## ■ 개발서비스

### ■ 여름가전 쇼핑몰 서비스(관리자)



▲ (“관리자”로 로그인후) 화면

“상품등록관리” 메뉴 선택후 화면 ▶



## ■ 개발서비스

### ■ 여름가전 쇼핑몰 서비스(관리자)

상품등록

상품분류  
선택기

상품번호  
예) P0009

상품명  
최대20자까지 입력

제조사  
제조회사를 입력하세요

모델명  
최대20자까지 입력

수량  
수량을 입력하세요

판매형태  
일반세일, 이벤트세일, 특별세일

할인율(%)  
예) 10%의 경우에는 10 으로 입력하세요

판매가격(원)  
가격을 입력하세요

상품이미지 (직경사이즈:250x180)  
찾아보기...

상품등록

© 2020 Copyright: 이 사이트는 현재 수업용으로 제작됩니다. 가천대학교 컴퓨터공학과 이병문

▲ “상품등록” 선택후 화면

쇼핑몰 / 관리자 / 구매자가입목록

+승인처리 ±강제탈퇴

이름	아이디	암호	주후확장1	주후확장2	주후확장3	주후확장4	주후확장5
관리자	admin	654321	-	-	-	-	-
홍길동	bmllee@gachon.ac.kr	gachon654321	-	-	-	-	-

Previous 1 2 3 Next

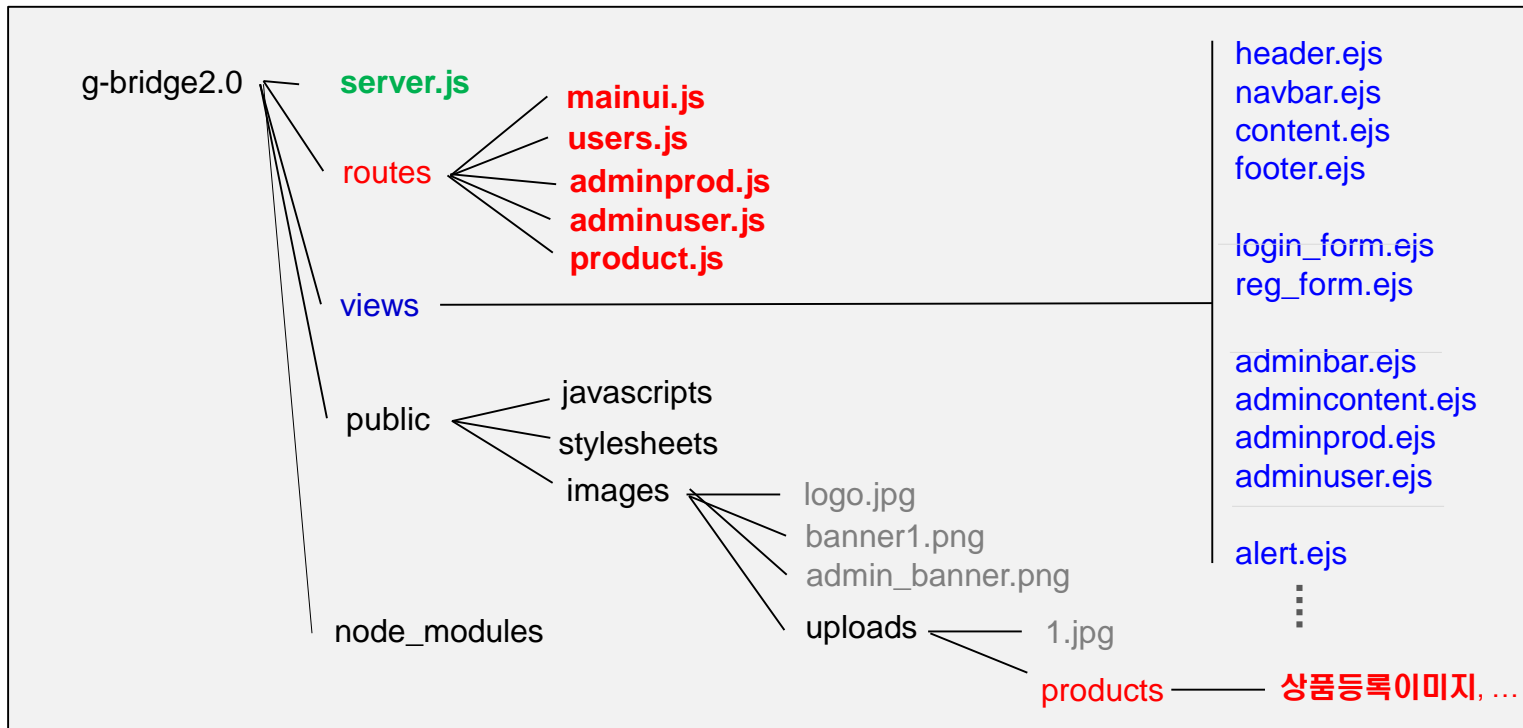
© 2020 Copyright: 이 사이트는 현재 수업용으로 제작됩니다. 가천대학교 컴퓨터공학과 이병문

## ■ 소스디렉터리 구조

### ■ MVC모델구조

front-end 프레임워크 : bootstrap jQuery

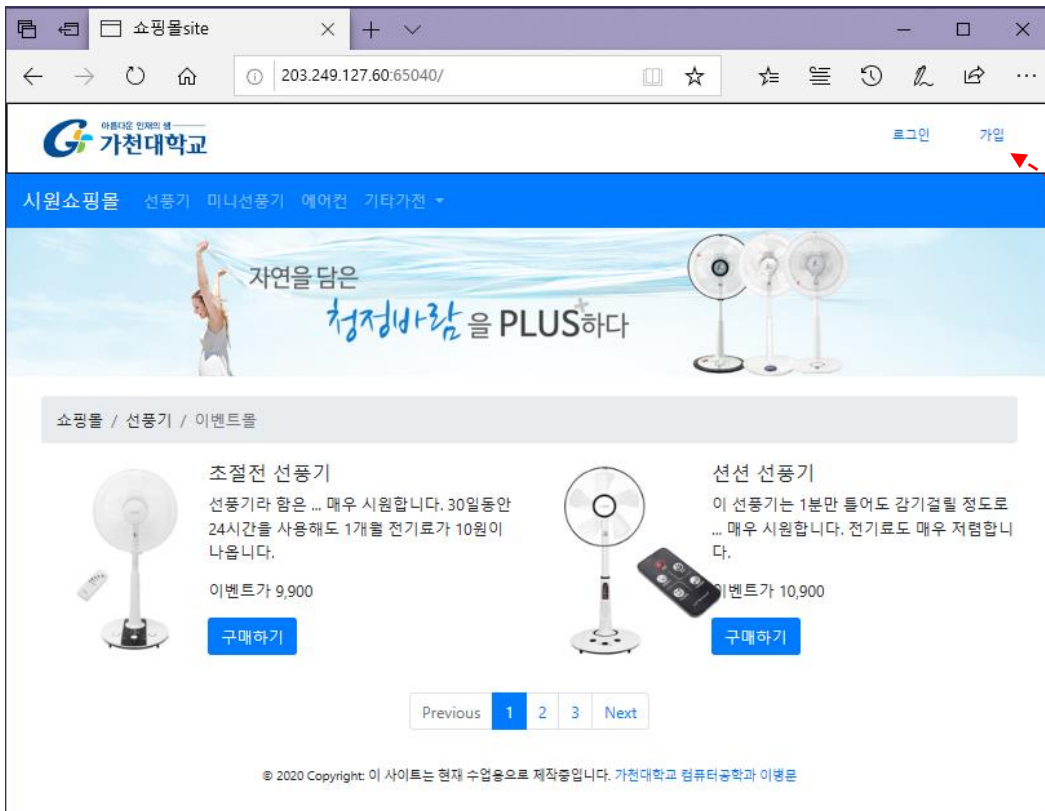
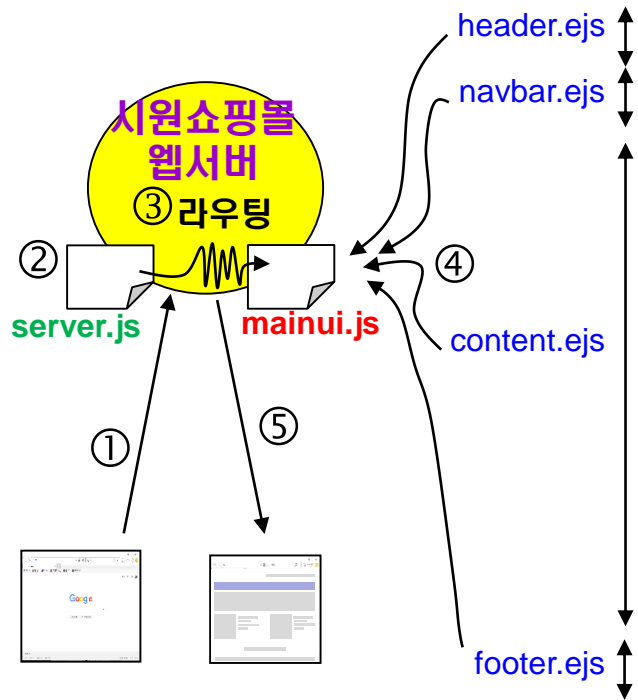
back-end 프레임워크 : Node.JS, express, **ejs**, mysql





## ■ 샘플소스 분석

- server.js / mainui.js



로그인전 / 로그인후 처리필요

## ■ 샘플소스 분석

```
const express = require('express');
const cookieParser = require('cookie-parser');
const session = require('express-session');
const bodyParser = require('body-parser');
const createError = require('http-errors');
const path = require('path');
const app = express();
```

### // 쇼핑몰 개발소스 모듈

```
const mainui = require('./routes/mainui');
const users = require('./routes/users');
const adminprod = require('./routes/adminprod');
const adminuser = require('./routes/adminuser');
const product = require('./routes/product');
```

```
const PORT = 650XX; ← 개인별 포트주소로 수정!!
```

### // 실행환경 (초기화)설정

```
app.set('views', path.join(__dirname, 'views')); // views경로 설정
app.set('view engine', 'ejs'); // view엔진 지정
app.use(express.static(path.join(__dirname, 'public'))); // public 설정 (브라우저부분 설정)
app.use(express.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(bodyParser.urlencoded({ extended: false }));
app.use(express.json());
```

```
app.use(session({ key: 'sid',
  secret: 'secret key', // 세션id 암호화할때 사용
  resave: false, // 접속할때마다 id부여금지
  saveUninitialized: true })); // 세션id사용전에는 발급금지
```

### // URI와 핸들러를 매핑

```
app.use('/', mainui); // URI (/) 는 mainui 으로 라우팅
app.use('/users', users); // URI('/users') 은 users 로 라우팅
app.use('/adminprod', adminprod); // URI('/adminprod') 는 adminprod 로 라우팅
app.use('/adminuser', adminuser); // URI('/adminuser') 접속하면 adminuser 로 라우팅
app.use('/product', product); // URI('/product') 접속하면 product로 라우팅
```

### // 쇼핑몰 웹서버를 실행합니다.

```
app.listen(PORT, function () {
  console.log('서버실행: http://203.249.127.60:' + PORT);
  console.log('서버실행: http://192.9.80.96:' + PORT);
});
```

## ■ 샘플소스 분석

mainui.js

### // 메인화면 출력파일

```
const express = require('express');
const ejs = require('ejs');
const fs = require('fs');
const router = express.Router();
var loglevel = 1;

const GetMainUI = (req, res) => { // 메인화면을 출력합니다
let htmlstream = '';
```

```
logging(loglevel, ' GetMainUI() 호출 ! ');
```

```
htmlstream = fs.readFileSync(__dirname + '/../views/header.ejs', 'utf8');
```

// 헤더부분

```
if (req.session.auth && req.session.admin) { // 만약, 관리자가 로그인했다면
    htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/adminbar.ejs', 'utf8'); // 관리자메뉴
    htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/admincontent.ejs', 'utf8');
} else { // (로그인하지 않고) 처음 접속했을 경우
    htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/navbar.ejs', 'utf8'); // 일반사용자메뉴
    htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/content.ejs', 'utf8'); // Content
}
```

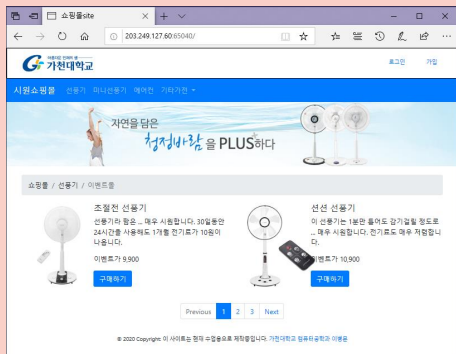
```
htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/footer.ejs', 'utf8');
```

// Footer

header.ejs  
navbar.ejs vs adminbar.ejs

content.ejs vs admincontent.ejs

footer.ejs



## ■ 샘플소스 분석

```
res.writeHead(200, {'Content-Type':'text/html; charset=utf8'});
if (req.session.auth) { // true :로그인된 상태, false : 로그인안된 상태
  res.end(ejs.render(htmlstream, { 'title' : '쇼핑몰site',
    'logurl': '/users/logout',
    'loglabel': '로그아웃',
    'regurl': '/users/profile',
    'reglabel':req.session.who })); // 세션에 저장된 사용자명표시
}
else {
  res.end(ejs.render(htmlstream, { 'title' : '쇼핑몰site',
    'logurl': '/users/auth',
    'loglabel': '로그인',
    'regurl': '/users/reg',
    'reglabel':'가입' }));
}
};

const logging = (level, logmsg) => {
  if (level != 0) {
    console.log(level, logmsg)
    loglevel++;
  }
}
```

mainui.js



```
// '/' get 메소드의 핸들러를 정의
router.get('/', GetMainUI); // URI (/) 는 GetMainUI 로 라우팅

// 외부로 뱉습니다.
module.exports = router
```

## ■ 샘플소스 분석

Ejs 바인딩 지점



```

<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <title> <%= title %> </title>
  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
    integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
    crossorigin="anonymous">

  <!-- 스크립트 코드링크 (CND)를 지정 -->
  <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-?????????XpG5KkN"
    crossorigin="anonymous"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js" integrity="sha384-??0b4Q"
    crossorigin="anonymous"></script>
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-???07jRM"
    crossorigin="anonymous"></script>
</head>
<body>
  <!-- 헤더부분 -->
  <div class="container-fluid" style="border: solid 1px black; border-radius: 1px">
    <div class="row align-items-center">
      <div class="col-2"></div>
      <div class="col-8"> </div>
      <div class="col-1"> <a href='<%=logurl%>'><h6><small><%=loglabel%></small></h6></a></div>
      <div class="col-1"> <a href='<%=regurl%>'><h6><small><%=reglabel%></small></h6></a></div>
    </div>
  </div>

```

header.ejs

Ejs 바인딩 지점

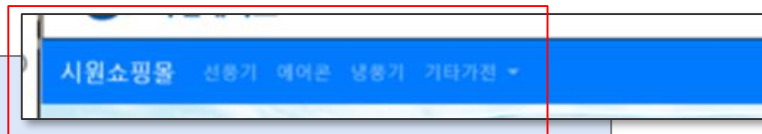
## ■ 샘플소스 분석

navbar.ejs

```
<!DOCTYPE html>
<!-- GNB(Global Navigation Bar) 부분 -->
<nav class="navbar navbar-expand-md navbar-dark bg-primary">

  <a class="navbar-brand" href="/">시원쇼핑몰</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent"
    aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item">
        <a class="nav-link" href="/product/list/?category=fan">선풍기<span class="sr-only">(current)</span></a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="/product/list/?category=minisun">미니선풍기</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="/product/list/?category=aircon">에어컨</a>
      </li>
    </ul>
  </div>
</nav>
```



## ■ 샘플소스 분석

navbar.ejs

```
<li class="nav-item dropdown">
  <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button"
    data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">기타가전 </a>
  <div class="dropdown-menu" aria-labelledby="navbarDropdown">
    <a class="dropdown-item" href="/product/list/?category=aircool">냉풍기</a>
    <a class="dropdown-item" href="#">공기청전기</a>
    <div class="dropdown-divider"></div>
    <a class="dropdown-item" href="/product/list/?category=fridge">냉장고</a>
  </div>
</li>
</ul>
</div>
</nav>
```



## ■ 샘플소스 분석

```

<!DOCTYPE html>
<!-- Content (배너이미지) 부분 -->
<div>  </div>

```



```

<!-- 컨텐츠(예, 쇼핑몰일 경우 상품리스트) 헤더부분 -->
<div class="container-fluid" style="border: solid 20px white; border-radius: 1px">
  <div class="panel panel-default">
    <div class="panel-heading">
      <nav aria-label="breadcrumb">
        <ol class="breadcrumb">
          <li class="breadcrumb-item">쇼핑몰</li>
          <li class="breadcrumb-item">선풍기</li>
          <li class="breadcrumb-item active" aria-current="page">이벤트몰</li>
        </ol>
      </nav>
    </div>
    <div class="panel-body">
      <div class="row align-items-center">
        <div class="col-2"></div>
        <div class="col-4">
          <h5 class="mt-0">초절전 선풍기</h5>
          <p>선풍기라 함은 ... 매우 시원합니다. 30일동안 24시간을 사용해도 1개월 전기료가 10원이 나옵니다.</p>
          <p>이벤트가 9,900</p>
          <button type="button" class="btn btn-primary">구매하기</button>
        </div>
      </div>
    </div>
  </div>
</div>

```



임시코드  
(추후,  
DB연동  
필요)



## ■ 샘플소스 분석

임시코드  
(추후,  
DB연동  
필요)

```
<div class="col-2">
  
</div>
<div class="col-4">
  <h5 class="mt-0">선선 선풍기</h5>
  <p class=xs>이 선풍기는 1분만 틀어도 감기걸릴 정도로 ... 매우 시원합니다. 전기료도 매우 저렴합니다.</p>
  <p>이벤트가 10,900</p>
  <button type="button" class="btn btn-primary">구매하기</button>
</div>
</div> </div> </div> </div>

<p> <p> <p> <p>
<nav aria-label="...">
  <ul class="pagination justify-content-center">
    <li class="page-item disabled">
      <a class="page-link" href="#" tabindex="-1" aria-disabled="true">Previous</a>
    </li>
    <li class="page-item active" aria-current="page">
      <a class="page-link" href="#">1 <span class="sr-only">(current)</span></a>
    </li>
    <li class="page-item"> <a class="page-link" href="#">2</a> </li>
    <li class="page-item"> <a class="page-link" href="#">3</a> </li>
    <li class="page-item"> <a class="page-link" href="#">Next</a> </li>
  </ul>
</nav>
```

content.ejs

Previous 1 2 3 Next

## ■ 샘플소스 분석

footer.ejs

```
<!doctype html>
<!-- Footer -->
<p>
  <footer class="page-footer font-small blue">
    <div class="footer-copyright text-center py-3">© 2020 Copyright: 이 사이트는 현재 수업용으로 제작중입니다.
      <a href="#"> 가천대학교 컴퓨터공학과 이병문 </a>
    </div>
  </footer>

</body>
</html>
```

© 2020 Copyright: 이 사이트는 현재 수업용으로 제작중입니다. 가천대학교 컴퓨터공학과 이병문

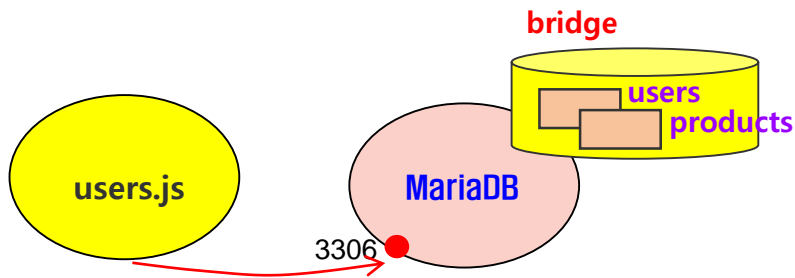
## ■ 샘플소스 분석

users.js

```
const fs = require('fs');
const express = require('express');
const ejs = require('ejs');
const mysql = require('mysql');
const bodyParser = require('body-parser');
const session = require('express-session');
const router = express.Router();

router.use(bodyParser.urlencoded({ extended: false }));

const db = mysql.createConnection({
  host: 'localhost',      // DB서버 IP주소
  port: 3306,            // DB서버 Port주소
  user: 'bmlee',         // DB접속 아이디
  password: 'bmlee654321', // DB암호
  database: 'bridge'     // 사용할 DB명
});
```



MariaDB [bridge]> desc users;

Field	Type	Null	Key	Default	Extra
uid	varchar(20)	NO	PRI	NULL	
pass	varchar(20)	YES		NULL	
name	varchar(20)	YES		NULL	

MariaDB [bridge]> desc products;

Field	Type	Null	Key	Default	Extra
itemid	varchar(20)	NO	PRI	NULL	
category	varchar(20)	YES		NULL	
maker	varchar(20)	YES		NULL	
pname	varchar(30)	YES		NULL	
modelnum	varchar(20)	YES		NULL	
rdate	date	YES		NULL	
price	int(11)	YES		NULL	
dcrate	int(11)	YES		NULL	
amount	int(11)	YES		NULL	
event	varchar(30)	YES		NULL	
pic	varchar(200)	YES		NULL	

## ■ 샘플소스 분석

```
// ----- 회원가입기능 -----  
// 회원가입 입력양식을 브라우저로 출력합니다.  
const PrintRegistrationForm = (req, res) => {  
  let htmlstream = "";  
  
  htmlstream = fs.readFileSync(__dirname + '/../views/header.ejs','utf8');  
  htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/navbar.ejs','utf8');  
  htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/reg_form.ejs','utf8');  
  htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/footer.ejs','utf8');  
  res.writeHead(200, {'Content-Type':'text/html'; charset=utf8});  
  
  if (req.session.auth) { // true :로그인된 상태, false : 로그인안된 상태  
    res.end(ejs.render(htmlstream, { 'title' : '쇼핑몰site',  
                                   'logurl': '/users/logout',  
                                   'loglabel': '로그아웃',  
                                   'regurl': '/users/profile',  
                                   'reglabel':req.session.who }));  
  }  
  else {  
    res.end(ejs.render(htmlstream, { 'title' : '쇼핑몰site',  
                                   'logurl': '/users/auth',  
                                   'loglabel': '로그인',  
                                   'regurl': '/users/reg',  
                                   'reglabel':'가입' }));  
  }  
};
```

users.js

## ■ 샘플소스 분석

```
// 회원가입 양식에서 입력된 회원정보를 신규등록(DB에 저장)합니다.
const HandleRegistration = (req, res) => { // 회원가입
  let   body = req.body;
  let   htmlstream="";

  // 임시로 확인하기 위해 콘솔에 출력해봅니다.
  console.log('회원가입 입력정보 :%s, %s, %s',body.uid, body.pw1, body.uname);

  if (body.uid == " || body.pw1 == ") {
    console.log("데이터입력이 되지 않아 DB에 저장할 수 없습니다.");
    res.status(561).end('<meta charset="utf-8">데이터가 입력되지 않아 가입을 할 수 없습니다');
  }
  else {
    db.query('INSERT INTO users (uid, pass, name) VALUES (?, ?, ?)',
      [body.uid, body.pw1, body.uname], (error, results, fields) => {

      if (error) {
        htmlstream = fs.readFileSync(__dirname + '/../views/alert.ejs','utf8');
        res.status(562).end(ejs.render(htmlstream, { 'title': '알리미',
          'warn_title':'회원가입 오류',
          'warn_message':'이미 회원으로 등록되어 있습니다. 바로 로그인을 하시기 바랍니다.',
          'return_url':'/' }));
      } else {
        console.log("회원가입에 성공하였으며, DB에 신규회원으로 등록하였습니다.!!");
        res.redirect('/');
      }
    })
  }
};
```

users.js

```
// REST API의 URI와 핸들러를 매핑합니다.
router.get('/reg', PrintRegistrationForm); // 회원가입화면을 출력처리
router.post('/reg', HandleRegistration); // 회원가입내용을 DB에 등록처리
router.get('/', function(req, res) { res.send('respond with a resource 111'); });
```

## ■ 샘플소스 분석

users.js

// ----- 로그인기능 -----

// 로그인 화면을 웹브라우저로 출력합니다.

```
const PrintLoginForm = (req, res) => {  
  let htmlstream = '';
```

```
  htmlstream = fs.readFileSync(__dirname + '/../views/header.ejs', 'utf8');  
  htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/navbar.ejs', 'utf8');  
  htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/login_form.ejs', 'utf8');  
  htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/footer.ejs', 'utf8');  
  res.writeHead(200, {'Content-Type': 'text/html; charset=utf8'});
```

```
  if (req.session.auth) { // true : 로그인된 상태, false : 로그인안된 상태  
    res.end(ejs.render(htmlstream, { 'title' : '쇼핑몰site',  
      'logurl': '/users/logout',  
      'loglabel': '로그아웃',  
      'regurl': '/users/profile',  
      'reglabel': req.session.who }));  
  }  
  else {  
    res.end(ejs.render(htmlstream, { 'title' : '쇼핑몰site',  
      'logurl': '/users/auth',  
      'loglabel': '로그인',  
      'regurl': '/users/reg',  
      'reglabel': '가입' }));  
  }  
};
```

## ■ 샘플소스 분석

// 로그인을 수행합니다. (사용자인증처리)

```
const HandleLogin = (req, res) => {
  let body = req.body;
  let userid, userpass, username;
  let sql_str;
  let htmlstream = "";

  console.log('로그인 입력정보: %s, %s', body.uid, body.pass);

  if (body.uid == "" || body.pass == "") {
    console.log("아이디나 암호가 입력되지 않아서 로그인할 수 없습니다.");
    res.status(562).end('<meta charset="utf-8">아이디나 암호가 입력되지 않아서 로그인할 수 없습니다.');
  }
  else {
    sql_str = "SELECT uid, pass, name from users where uid =" + body.uid + " and pass=" + body.pass + " ";
    console.log("SQL: " + sql_str);
    db.query(sql_str, (error, results, fields) => {
      if (error) { res.status(562).end("Login Fail as No id in DB!"); }
      else {
        if (results.length <= 0) { // select 조회결과가 없는 경우 (즉, 등록계정이 없는 경우)
          htmlstream = fs.readFileSync(__dirname + '/../views/alert.ejs', 'utf8');
          res.status(562).end(ejs.render(htmlstream, { 'title': '알리미',
            'warn_title': '로그인 오류',
            'warn_message': '등록된 계정이나 암호가 틀립니다.',
            'return_url': '/' })));
        }
      }
    });
  }
}
```

users.js

## ■ 샘플소스 분석

users.js

```
else { // select 조회결과가 있는 경우 (즉, 등록사용자인 경우)
  results.forEach((item, index) => {
    userid = item.uid; userpass = item.pass; username = item.name;
    console.log("DB에서 로그인성공한 ID/암호:%s/%s", userid, userpass);
    if (body.uid == userid && body.pass == userpass) {
      req.session.auth = 99;           // 임의 수(99)로 로그인성공했다는 것을 설정함
      req.session.who = username;       // 인증된 사용자명 확보 (로그인후 이름출력용)
      if (body.uid == 'admin')          // 만약, 인증된 사용자가 관리자(admin)라면 이를 표시
        req.session.admin = true;
      res.redirect('/');
    }
  }); /* foreach */
} // else
} // else
});
}
}

// REST API의 URI와 핸들러를 매핑합니다.
// URI: http://xxxx/users/auth
router.get('/auth', PrintLoginForm); // 로그인 입력화면을 출력
router.post('/auth', HandleLogin);  // 로그인 정보로 인증처리
```

```
session.auth
session.who
session.admin
```



## ■ 샘플소스 분석

users.js

```
// ----- 로그아웃기능 -----
const HandleLogout = (req, res) => {
  req.session.destroy(); // 세션을 제거하여 인증오작동 문제를 해결
  res.redirect("/");      // 로그아웃후 메인화면으로 재접속
}

// REST API의 URI와 핸들러를 매핑합니다.
router.get('/logout', HandleLogout); // 로그아웃 기능

// ----- 정보변경 기능을 개발합니다 -----
const PrintProfile = (req, res) => {
  let htmlstream = "";

  htmlstream = fs.readFileSync(__dirname + '/../views/alert.ejs', 'utf8');
  res.status(562).end(ejs.render(htmlstream, { 'title': '알리미',
    'warn_title': '계정정보 준비중',
    'warn_message': '계정정보(예, 암호변경, 주소변경, 전화번호변경 등)변경기능을 추후에 개발할 예정입니다',
    'return_url': '/' })));
}

router.get('/profile', PrintProfile); // 정보변경화면을 출력

module.exports = router;
```

## ■ 샘플소스 분석

reg\_form.ejs

```
<!DOCTYPE html>
```

```
<script language="javascript">
```

```
function validate() {
```

```
    let pw1 = document.getElementById("pw1").value;
```

```
    let pw2 = document.getElementById("pw2").value;
```

```
    let uid = document.getElementById("uid").value;
```

```
    if (pw1 != pw2) {
```

```
        alert("암호가 서로 다릅니다. 다시입력하세요");
```

```
        return false;
```

```
    }
```

```
    return true;
```

```
}
```

```
</script>
```

```
<p>
```

```
<div class="container"> <!-- container class 가 페이지의 콘텐츠영역을 지정 -->
```

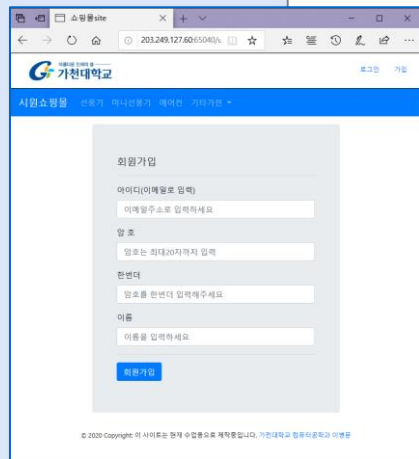
```
<div class="jumbotron" style="width:500px; margin:auto">
```

```
<div class="bm-login-form" style='width:400px; margin-left:auto; margin-right:auto; border:1px'>
```

```
    <div class="starter-template">
```

```
        <h5> 회원가입 </h5> <hr>
```

```
    </div>
```

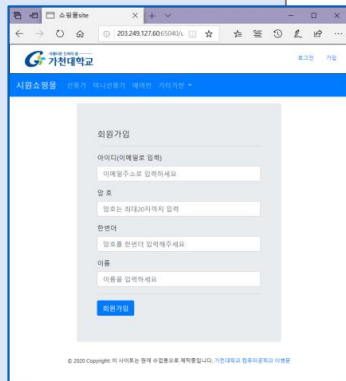


## ■ 샘플소스 분석

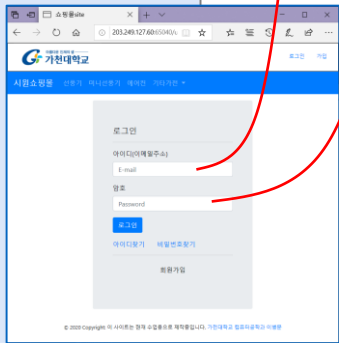
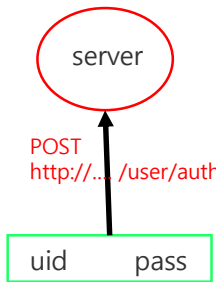
reg\_form.ejs

```
<form name=join onsubmit="return validate()" method=post action="/users/reg">
  <div class="form-group">
    <label for="exampleInputEmail1">아이디(이메일로 입력)</label>
    <input type="email" class="form-control" name="uid" id="uid" aria-describedby="emailHelp" placeholder="이메일주소로 입력하세요">
  </div>
  <div class="form-group">
    <label for="exampleInputPassword1">암 호</label>
    <input type="password" class="form-control" name="pw1" id="pw1" placeholder="암호는 최대20자까지 입력">
  </div>
  <div class="form-group">
    <label for="exampleInputPassword2">한번더</label>
    <input type="password" class="form-control" name="pw2" id="pw2" placeholder="암호를 한번더 입력해주세요">
  </div>
  <div class="form-group">
    <label for="exampleInputEmail1">이름</label>
    <input type="text" class="form-control" name="uname" id="uname" placeholder="이름을 입력하세요">
  </div>
  <hr>
  <button type="submit" class="btn btn-primary">회원가입</button>
</form>
</div>
</div>
</div><!-- /.container -->
```

<!-- Footer 영역 -->



# login\_form.ejs



## ■ 샘플소스 분석

```
const fs = require('fs');
const express = require('express');
const ejs = require('ejs');
const url = require('url');
const mysql = require('mysql');
const bodyParser = require('body-parser');
const session = require('express-session');
const multer = require('multer');
const router = express.Router();

const db = mysql.createConnection({
  host: 'localhost',      // DB서버 IP주소
  port: 3306,             // DB서버 Port주소
  user: 'bmlee',          // DB접속 아이디
  password: 'bmlee654321', // DB암호
  database: 'bridge'      //사용할 DB명
});
```

product.js

```
// ----- 상품리스트 기능 -----
// 등록된 상품리스트를 브라우저로 출력합니다.
const PrintCategoryProd = (req, res) => {
  let htmlstream = "";
  let htmlstream2 = "";
  let sql_str, search_cat;
  const query = url.parse(req.url, true).query;

  console.log('상품카테고리 조회 ' + query.category);

  if (req.session.auth) { // (로그인된 경우에만) 상품리스트를 출력합니다

    switch (query.category) {
      case 'fan' : search_cat = "선풍기"; break;
      case 'aircon': search_cat = "에어컨"; break;
      case 'aircool': search_cat = "냉풍기"; break;
      case 'fridge': search_cat = "냉장고"; break;
      case 'minisun': search_cat = "미니선풍기"; break;
      default: search_cat = "선풍기"; break;
    }

    htmlstream = fs.readFileSync(__dirname + '/../views/header.ejs','utf8'); // 헤더부분
    htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/navbar.ejs','utf8');
    htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/product.ejs','utf8');
    htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/footer.ejs','utf8');
```

## ■ 샘플소스 분석

product.js

```
// 상품조회용 SQL작성
sql_str = "SELECT maker, pname, modelnum, rdate, price, pic from products where category='" + search_cat + "' order by rdate desc;";

res.writeHead(200, {'Content-Type': 'text/html; charset=utf8'});

db.query(sql_str, (error, results, fields) => { // 상품조회용 SQL실행
  if (error) { res.status(562).end("AdminPrintProd: DB query is failed"); }
  else if (results.length <= 0) { // 조회된 상품이 없다면, 오류메시지 출력
    htmlstream2 = fs.readFileSync(__dirname + '/../views/alert.ejs', 'utf8');
    res.status(562).end(ejs.render(htmlstream2, { 'title': '알리미',
      'warn_title': '상품조회 오류',
      'warn_message': '조회된 상품이 없습니다.',
      'return_url': '/' })));
  }
  else { // 조회된 상품이 있다면, 상품리스트를 출력
    res.end(ejs.render(htmlstream, { 'title': '쇼핑몰site',
      'logurl': '/users/logout',
      'loglabel': '로그아웃',
      'regurl': '/users/profile', // 로그인명 클릭시 URL
      'reglabel': req.session.who, // 로그인 명
      'category': search_cat,
      'prodata': results }))); // 조회된 상품정보
  } // else
}); // db.query()
} // if (req.session.auth) 여기까지가 로그인되었을때 상품을 출력하는 부분
```

## ■ 샘플소스 분석

product.js

```
else { // (로그인하지 않고) 본 페이지를 참조하면 오류를 출력
  htmlstream = fs.readFileSync(__dirname + '/../views/alert.ejs','utf8');
  res.status(562).end(ejs.render(htmlstream, { 'title': '알리미',
                                              'warn_title': '로그인 필요',
                                              'warn_message': '상품검색을 하려면, 로그인이 필요합니다.',
                                              'return_url': '/' }));
}
};

// REST API의 URI와 핸들러를 매핑합니다.
router.get('/list', PrintCategoryProd); // 상품리스트를 화면에 출력

module.exports = router;
```

product.ejs

```

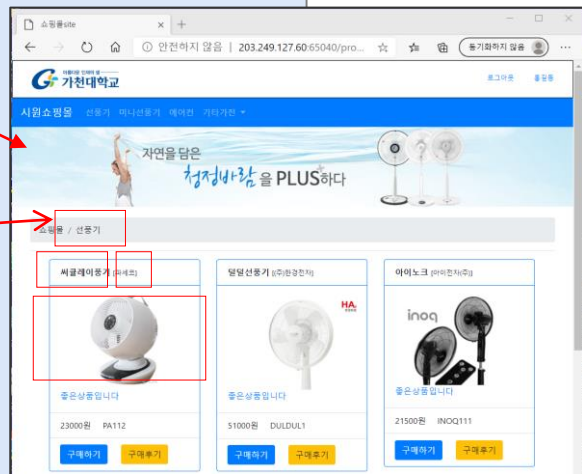
<!DOCTYPE html>
<!-- Content (배너이미지) 부분 -->
<div>  </div>

<!-- 컨텐츠(예, 쇼핑몰일경우 상품리스트) 헤더부분 -->
<div class="container-fluid" style="border: solid 20px white; border-radius: 1px">
  <div class="panel panel-default">
    <div class="panel-heading">
      <nav aria-label="breadcrumb">
        <ol class="breadcrumb">
          <li class="breadcrumb-item">쇼핑몰</a></li>
          <li class="breadcrumb-item"><%= category %> </a></li>
        </ol>
      </nav>
    </div>
  </div>

  <hr>
  <div class="container">
    <div class="row align-items-start">
      <% prodata.forEach(function (item, index) { %>
        <div class=col-4>
          <div class="card border-primary mb-3" style="max-width: 18rem; max-height: 26rem">
            <div class="card-header bg-transparent border-primary"><strong><%= item.pname %> </strong>
              <small> [<%= item.maker %>] </small></div>

            <div class="card-body text-primary">
              <div> <img class="img-fluid" src=<%= item.pic %> > </div>

```





product.ejs

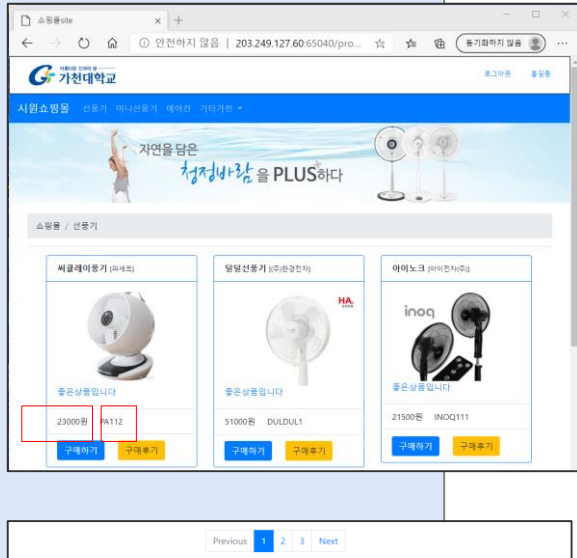
```

    <p class="card-text">좋은상품입니다
  </div>
  <div class="card-footer bg-transparent"><%= item.price %>원 &nbsp;&nbsp;&nbsp;<%= item.modelnum %> </div>
  <div class="card-footer bg-transparent"><button type=button class="btn btn-primary">구매하기</button> &nbsp;&nbsp;&nbsp;
    <button type=button class="btn btn-warning">구매하기</button> </div>

</div>
</div>
<% }); %>
</div>
</div>

<p>
<nav aria-label="...">
  <ul class="pagination justify-content-center">
    <li class="page-item disabled">
      <a class="page-link" href="#" tabindex="-1" aria-disabled="true">Previous</a>
    </li>
    <li class="page-item active" aria-current="page">
      <a class="page-link" href="#">1 <span class="sr-only">(current)</span></a>
    </li>
    <li class="page-item"><a class="page-link" href="#">2</a></li>
    <li class="page-item"><a class="page-link" href="#">3</a></li>
    <li class="page-item">
      <a class="page-link" href="#">Next</a>
    </li>
  </ul>
</nav>

```

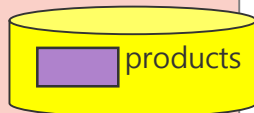


관리자 서비스

```
const fs = require('fs');
const express = require('express');
const ejs = require('ejs');
const mysql = require('mysql');
const bodyParser = require('body-parser');
const session = require('express-session');
const multer = require('multer');
const upload = multer({dest: __dirname + '/../public/images/uploads/products'}); // 업로드 디렉토리를 설정한다.
const router = express.Router();
```

adminprod.js

bridge



```
const db = mysql.createConnection({
  host: 'localhost',          // DB서버 IP주소
  port: 3306,                 // DB서버 Port주소
  user: 'bmlee',              // DB접속 아이디
  password: 'bmlee654321',    // DB암호
  database: 'bridge'          // 사용할 DB명
});
// ----- 상품리스트 기능 -----
// (관리자용) 등록된 상품리스트를 브라우저로 출력합니다.
```

```
const AdminPrintProd = (req, res) => {
  let htmlstream = "";
  let htmlstream2 = "";
  let sql_str;
```

```
  if (req.session.auth && req.session.admin) { // 관리자로 로그인된 경우에만 처리한다
    htmlstream = fs.readFileSync(__dirname + '/../views/header.ejs','utf8'); // 헤더부분
    htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/adminbar.ejs','utf8'); // 관리자메뉴
    htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/adminproduct.ejs','utf8'); // 관리자메인화면
    htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/footer.ejs','utf8'); // Footer
    sql_str = "SELECT itemid, category, maker, pname, modelnum, rdate, price, amount from products order by rdate desc;"; // 상품조회SQL
```

MariaDB [bridge]&gt; desc products;

Field	Type	Null	Key	Default	Extra
itemid	varchar(20)	NO	PRI	NULL	
category	varchar(20)	YES		NULL	
maker	varchar(20)	YES		NULL	
pname	varchar(30)	YES		NULL	
modelnum	varchar(20)	YES		NULL	
rdate	date	YES		NULL	
price	int(11)	YES		NULL	
dcrate	int(11)	YES		NULL	
amount	int(11)	YES		NULL	
event	varchar(30)	YES		NULL	
pic	varchar(200)	YES		NULL	

adminprod.js

```

res.writeHead(200, {'Content-Type': 'text/html; charset=utf8'});

db.query(sql_str, (error, results, fields) => { // 상품조회 SQL실행
  if (error) { res.status(562).end(" AdminPrintProd: DB query is failed "); }
  else if (results.length <= 0) { // 조회된 상품이 없다면, 오류메시지 출력
    htmlstream2 = fs.readFileSync(__dirname + '/../views/alert.ejs', 'utf8');
    res.status(562).end(ejs.render(htmlstream2, { 'title': '알리미',
                                              'warn_title': '상품조회 오류',
                                              'warn_message': '조회된 상품이 없습니다.',
                                              'return_url': '/' })));
  }
  else { // 조회된 상품이 있다면, 상품리스트를 출력
    res.end(ejs.render(htmlstream, { 'title': '쇼핑몰site',
                                    'logurl': '/users/logout',
                                    'loglabel': '로그아웃',
                                    'regurl': '/users/profile',
                                    'reglabel': req.session.who,
                                    prodata : results }))); // 조회된 상품정보
  } // else
}); // db.query()
}
else { // (관리자기능이기 때문에, 관리자로 로그인하지 않고) 본 페이지를 참조하면 오류를 출력
  htmlstream = fs.readFileSync(__dirname + '/../views/alert.ejs', 'utf8');
  res.status(562).end(ejs.render(htmlstream, { 'title': '알리미',
                                              'warn_title': '상품등록기능 오류',
                                              'warn_message': '관리자로 로그인되어 있지 않아서, 상품등록 기능을 사용할 수 없습니다.',
                                              'return_url': '/' })));
}
};

```

adminprod.js

```
// ----- 상품등록기능 -----
// 상품등록 입력양식을 브라우저로 출력합니다.

const PrintAddProductForm = (req, res) => {
  let htmlstream = "";

  if (req.session.auth && req.session.admin) { // 관리자로 로그인된 경우에만 처리한다
    htmlstream = fs.readFileSync(__dirname + '/../views/header.ejs','utf8'); // 헤더부분
    htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/adminbar.ejs','utf8'); // 관리자메뉴
    htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/product_form.ejs','utf8'); // 관리자메인화면
    htmlstream = htmlstream + fs.readFileSync(__dirname + '/../views/footer.ejs','utf8'); // Footer

    res.writeHead(200, {'Content-Type':'text/html; charset=utf8'});
    res.end(ejs.render(htmlstream, { 'title' : '쇼핑몰site',
                                   'logurl': '/users/logout',
                                   'loglabel': '로그아웃',
                                   'regurl': '/users/profile',
                                   'reglabel': req.session.who })));
  }
  else {
    htmlstream = fs.readFileSync(__dirname + '/../views/alert.ejs','utf8');
    res.status(562).end(ejs.render(htmlstream, { 'title': '알리미',
                                                'warn_title': '상품등록기능 오류',
                                                'warn_message': '관리자로 로그인되어 있지 않아서, 상품등록 기능을 사용할 수 없습니다.',
                                                'return_url': '/' })));
  }
};
```

adminprod.js

// 상품등록 양식에서 입력된 상품정보를 신규로 등록(DB에 저장)합니다.

```
const HandleAddProduct = (req, res) => { // 상품등록
```

```
  let body = req.body;
```

```
  let htmlstream = "";
```

```
  let datestr, y, m, d, regdate;
```

```
  let prodimage = '/images/uploads/products/'; // 상품이미지 저장디렉터리
```

```
  let picfile = req.file;
```

```
  let result = { originalName : picfile.originalname, size : picfile.size  }
```

```
  if (req.session.auth && req.session.admin) {
```

```
    if (body.itemid == "" || datestr == "") {
```

```
      console.log("상품번호가 입력되지 않아 DB에 저장할 수 없습니다.");
```

```
      res.status(561).end('<meta charset="utf-8">상품번호가 입력되지 않아 등록할 수 없습니다');
```

```
    }
```

```
    else {
```

```
      prodimage = prodimage + picfile.filename;
```

```
      regdate = new Date();
```

```
      db.query("INSERT INTO products (itemid, category, maker, pname, modelnum, rdate, price, dcrate, amount, event, pic) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)",
```

```
        [body.itemid, body.category, body.maker, body.pname, body.modelnum, regdate,
```

```
        body.price, body.dcrate, body.amount, body.event, prodimage], (error, results, fields) => {
```

```
      if (error) {
```

```
        htmlstream = fs.readFileSync(__dirname + '/../views/alert.ejs', 'utf8');
```

```
        res.status(562).end(ejs.render(htmlstream, { 'title': '알리미',
```

```
          'warn_title': '상품등록 오류',
```

```
          'warn_message': '상품으로 등록할때 DB저장 오류가 발생하였습니다. 원인을 파악하여 재시도 바랍니다',
```

```
          'return_url': '/' }));
```

```
      }
```

adminprod.js

```
else {
  console.log("상품등록에 성공하였으며, DB에 신규상품으로 등록하였습니다.!");
  res.redirect('/adminprod/list');
}
});
}
}
else {
  htmlstream = fs.readFileSync(__dirname + '/../views/alert.ejs','utf8');
  res.status(562).end(ejs.render(htmlstream, { 'title': '알리미',
    'warn_title': '상품등록기능 오류',
    'warn_message': '관리자로 로그인되어 있지 않아서, 상품등록 기능을 사용할 수 없습니다.',
    'return_url': '/' }));
}
};

// REST API의 URI와 핸들러를 매핑합니다.
router.get('/form', PrintAddProductForm);
router.post('/product', upload.single('photo'), HandleAddProduct);
router.get('/list', AdminPrintProd);

module.exports = router;
```

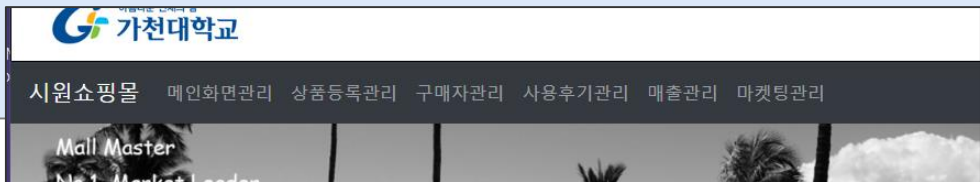
// 상품등록화면을 출력처리  
// 상품등록내용을 DB에 저장처리  
// 상품리스트를 화면에 출력

## ■ 샘플소스 분석

adminbar.ejs

```
<!DOCTYPE html>
<!-- GNB(Global Navigation Bar) 부분 -->
<nav class="navbar navbar-expand-md navbar-dark bg-dark">
  <a class="navbar-brand" href="/">시원쇼핑몰</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent"
    aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item"> <a class="nav-link" href="#">메인화면관리<span class="sr-only">(current)</span></a> </li>
      <li class="nav-item"> <a class="nav-link" href="/adminprod/list">상품등록관리</a> </li>
      <li class="nav-item"> <a class="nav-link" href="/adminuser/userlist">구매자관리</a> </li>
      <li class="nav-item"> <a class="nav-link" href="#">사용후기관리</a> </li>
      <li class="nav-item"> <a class="nav-link" href="#">매출관리</a> </li>
      <li class="nav-item"> <a class="nav-link" href="#">마케팅관리</a> </li>
    </ul>
  </div>
</nav>
```





## ■ 샘플소스 분석

adminprod.ejs

```
<!DOCTYPE html>
<!-- Content (배너이미지) 부분 -->
<div>  </div>

<!-- 컨텐츠(예, 쇼핑몰일 경우 판매제품리스트)헤더부분 -->
<div class="container-fluid" style="border: solid 20px white; border-radius: 1px">
  <div class="panel panel-default">
    <div class="panel-heading">
      <nav aria-label="breadcrumb">
        <ol class="breadcrumb">
          <li class="breadcrumb-item">쇼핑몰</li>
          <li class="breadcrumb-item">관리자</li>
          <li class="breadcrumb-item active" aria-current="page">상품등록관리</li>
        </ol>
      </nav>
    </div>
  </div>
```



```
<hr>
<div class="container">
  <a class="btn btn-default" href="/adminprod/form">+신규등록</a>
  <a class="btn btn-default" href="#">±상품수정</a>
  <a class="btn btn-default" href="#">-선택삭제</a>
  <table class="table table-striped">
    <thead>
      <tr>
        <th>번호 </th>
        <th>제품분류 </th>
        <th>제조사 </th>
        <th>상품명 </th>
        <th>모델명 </th>
        <th>등록일 </th>
        <th>판매가 </th>
        <th>재고량 </th>
      </tr>
    </thead>
```

## ■ 샘플소스 분석

```

<tbody>
  <% prodata.forEach(function (item, index) { %>
    <tr>
      <td> <%= item.itemid %> </td>
      <td> <%= item.category %> </td>
      <td> <%= item.maker %> </td>
      <td> <%= item.pname %> </td>
      <td> <%= item.modelnum %> </td>
      <td> <%= item.rdate.toLocaleDateString(); %> </td>
      <td> <%= item.price %> </td>
      <td> <%= item.amount %> </td>
    </tr>
  <% }); %>
</table>
</div>

```

번호	제품분류	제조사	상품명	모델명	등록일	현재가	재고량
A1112	선풍기	파세코	써클레이풍기	PA112	2020-6-9	23000	300
P10231	선풍기	(주)한경전자	덜덜선풍기	DULDUL1	2020-6-9	51000	20
P01010	냉풍기	(주)한경전자	왕추워선풍기	HWA110	2020-6-9	43000	221
NANG2	냉장고	(주)엘자전자	패밀리냉장고	NANG222	2020-6-9	2500000	30

```

<p>
<nav aria-label="...">
<ul class="pagination justify-content-center">
  <li class="page-item disabled">
    <a class="page-link" href="#" tabindex="-1" aria-disabled="true">Previous</a>
  </li>
  <li class="page-item active" aria-current="page">
    <a class="page-link" href="#">1 <span class="sr-only">(current)</span></a>
  </li>
  <li class="page-item"><a class="page-link" href="#">2</a></li>
  <li class="page-item"><a class="page-link" href="#">3</a></li>
  <li class="page-item">
    <a class="page-link" href="#">Next</a>
  </li>
</ul>
</nav>

```

Previous 1 2 3 Next

- 쇼핑몰서비스 세미프로젝트

## ■ 세미프로젝트 개발요구사항

요구사항1) 제공된 쇼핑몰서비스 소스코드를 분석하여, 요구사항에 맞도록 수정하고 완성한다.

요구사항2) 구매자 기능(메뉴)에 대해서 추가해야 할 기능은 다음과 같다.

- (2.1) 회원가입양식을 수정하여 1) 휴대폰 2) 포인트(기본값: 천만원) 항목을 입력/저장하여라.
- (2.2) 로그인한후 로그인명을 클릭하면, 회원정보(암호, 휴대폰번호)를 변경할 수 있어야 한다.
- (2.3) 암호는 반드시 암호화하여 DB에 저장하여야 한다. (crypto모듈이용해서 des 알고리즘으로)
- (2.4) products 테이블에 상품설명필드를 추가하여, 상품설명이 “카드형태” 내부에 보이도록 한다.
- (2.5) (카드스타일의) 상품목록화면에서 구매하기 버튼을 클릭하면, 보유한 포인트로 지정한 수량만큼 구매처리 한다. 즉, 구매완료창+구매처리+구매결과DB 모두 반영되도록 한다.

요구사항3) 관리자 기능(메뉴)에 대해서 추가 또는 수정해야할 기능은 다음과 같다.

- (3.1) 관리자도 로그인명을 클릭하면 관리자정보(암호, 휴대폰번호)를 변경할 수 있어야 한다.
- (3.2) 상품등록화면과 DB를 수정하여, “신규등록”시 상품설명을 추가하고, 입력/저장할 수 있어야 한다.  
또한, 이미지크기를
- (3.3) 등록된 상품정보를 “상품수정”할 수 있어야 한다.
- (3.4) 등록된 상품을 “선택삭제 ” 할 수 있어야 한다
- (3.5) “상품등록관리”의 페이지네이션을 완성한다. (즉, 1페이지에 10개만 나오도록 한다.)
- (3.6) “매출관리”기능을 완성한다. (매출관리를 클릭하면, 매출이 큰 순서로 상품목록이 나오도록 한다.  
(즉, 순번, 매출총액, 상품명, 제조사, 모델명, 남은 재고량 으로 구성된 목록이 출력된다.)

## ■ 개발방법/제출방법

- 1) g-bridge2.1.tar.gz 를 다운받아, 압축을 풀면, g-bridge2.1 디렉터리가 생성된다.
- 2) g-bridge2.1 디렉터리 전체를 g-bridge3.0 으로 모두 복사하고,
- 3) G-bridge3.0 디렉터리에서 요구사항이 반영되도록 코드를 수정/추가/삭제하여 개발한다.
- 4) DB는 "bridge" DB를 **반드시** 그대로 사용한다.
- 5) **반드시** 모든 테이블명에는 "u?\_" 를 앞에 붙인다. (예, 15번의 경우, **u15\_users** **u15\_products** )
- 6) 개발이 완료된 사람은 소스코드전체와 DB테이블을 덤프해서 **g-bridge4.0-학번-이름-src.tar.gz** 로 통합/압축한 뒤에 이메일로 첨부해서 보낸다.  
(DB테이블은 sql 디렉터리를 생성한 뒤에, u?.sql 형태로 저장한다)  
단, 이메일 제목은 "**g-bridge4.0**-학번-이름 " 으로 보낸다.
- 7) 제출마감은 6/22(월)까지 입니다.