Node.js 프로그래밍

• Node.js란

Node.js는 구글의 Chrome Javascript Engine인 V8 Engine을 기반으로 구현된 일종의 Server Side Javascript 소프트웨어 시스템이다. Java언어가 각각의 OS 운영체제의 JVM(Java Virtual Machine)을 통해 구동되듯이 Node.js는 웹브라우저에 종속적인 Javascript를 각각 OS 운영체제에서 실행할 수 있도록 Javascript Runtime 환경을 제공하고 있다.

• 노드 설치하기

노드 공식 사이트에 접속하여 프로그램을 다운로드 받아 설치한다. 사이트는 영문 버전뿐만 아니라 한글 번전으로도 준비되어 있다.
https://nodejs.org/ko/

• 노드 환경 구성하기

노드를 설치했으므로 이제 노드의 패키지 관리자인 npm을 사용하여 익스프레스 프로젝트를 자동으로 생성해 주는 express-generator 모듈을 설치한다. 참고로 익스프레스는 노드 기반에서 웹 애플리케이션 개발을 지원하는 프레임워크이다. npm의 기본 사용법은 다음과 같다.

npm install [모듈명] -g
모듈을 전역으로 설치할 때 사용하는 방식이다. 여기서 설치 옵션인 g는 global을 뜻한다. 일반적으로 특정 프로젝트가 아닌 전체 프로젝트에서 공통으로 사용할 수 있는 익스프레스 템플릿 생성 모듈인 express-generator나 노드 프로젝트를 실행하는 nodemon 등을 전역으로 설치한다.

npm install [모듈명] --save
현재 작업 중인 프로젝트에 모듈을 설치할 때 사용하는 방식이다. --save 옵션으로 현재 프로젝트의 package.json에 모듈 이름과 버전을 추가한다. 이를 통해 현재 프로젝트가 사용하고 있는 모듈 이름과 버전을 알 수 있고 나중에 동일한 프로젝트를 다른 곳에 생성할 때 package.json을 복사해서 설치하면 되므로 매우 편리하다.

npm install
프로젝트 package.json에 작성된 모듈을 한 번에 설치하고 싶을 때 사용하는 명령어이다. package.json만 있다면 같은 버전의 모듈을 손쉽게 설치할 수 있다.

1) 프로젝트 디렉터리를 생성한 후 생성된 디렉터리로 이동한다.

```
C:₩project>
```

2) 익스프레스 프로젝트를 쉽게 생성해 주는 express-generator 모듈을 전역으로 설치한다. 전역으로 설치하면 다른 프로젝트에서도 사용할 수 있다. 사용법은 express -h로 살펴볼 수 있다.

```
C:₩project>npm install express-generator -g
```

3) express 명령어를 실행해서 프로젝트 디렉터리를 생성한다. 이때 템플릿 엔진은 ejs로 설치할 것이므로 -e 옵션을 추가한다. 이렇게 express-generator를 사용해서 프로젝트를 생성하면 기본적인 모듈들이 자동으로 pakage.json에 추가된다.

```
C:₩project>express -e web
```

4) package.json에 추가된 모듈을 설치한다. npm install 명령어를 사용하면 package.json에 선언된 모듈이 자동으로 설치된다. 이때 명령어는 web 디렉터리 아래에서 실행한다.

```
C:₩project₩web>npm install
```

5) 추가로 필요한 모듈을 설치한다. 이때는 설치 옵션에 --save를 주어 package.json에 설치된 모듈을 추가해야한다. 이렇게 추가해 놓으면 어떤 모듈을 사용하고 있는지 알 수 있으며 나중에 동일한 프로젝트를 다른 곳에 생성할 때 pakage.json을 복사해서 설치하면 매우 편리하다. 파일 업로드 모듈인 formidable과 mysql 연결 모듈인 mysql을 함께 설치하자.

```
C:₩project₩web>npm install formidable mysql --save
```

• 노드 실행하기

1) 노드는 npm start 명령어를 사용해서 실행할 수 있다. 참고로, 실행을 종료하고 싶다면 CTRL + C를 입력하면 된다.

```
C:₩project₩web>npm start  (npx port-kill 3000)
```

2) 결과는 브라우저에 http://localhost:3000으로 접속해서 확인할 수 있다. 노드는 기본적으로 3000번 포트에서 동작하도록 되어 있다. 다른 포트 번호로 변경하려면 C:₩project₩web₩bin 디렉터리의 www 파일의 내용을 수정해야 한다.

3) npm start로 노드를 실행하면 소스 코드를 수정한 것이 반영되지 않는다. 그래서 이런 경우를 위한 여러 모듈이 있는데 그중에서 간단히 사용할 수 있는 노드몬을 설치한다.

```
C:₩project₩web>npm install nodemon -g
```

4) 프로젝트 디렉터리에서 nodemon이라고 실행하면 소스 코드가 수정될 경우 수정을 감지하고 다시 시작하는 것을 확인할 수 있다. 따라서 소스 코드를 변경할 때마다 서버를 재 시작해야 하는 번거로움을 줄일 수 있다.

```
C:₩project₩web>nodemon
```

5) nodemon 실행 오류 시에는 관리자 권한으로 PowerShell을 실행한다.

```
C:₩windows₩system32>ExecutionPolicy  //현재정책확인
C:₩windows₩system32>Set-ExecutionPolicy Unrestricted //Resticted를 Unrestricted로 변경한다.
```

• 프로젝트의 기본 디렉터리

1) bin 디렉터리
bin 디렉터리에는 www 이름의 파일이 한 개 존재한다. 이 파일은 확장자가 없지만, 내부는 노드가 서버로서 동작하기 위한 기본적인 코드가 자바스크립트로 작성되어 있다. 또한 서버를 시작할 포트가 지정되어 있다.

2) node_modules 디렉터리
이 디렉터리는 npm install을 실행하면서 생긴 디렉터리이다. 그래서 package.json에 선언되어 있는 모듈과 이 모듈을 실행가기 위해서 필요한 의존관계의 모듈이 설치되어있다.

3) public 디렉터리
이 디렉터리에는 이미지, 자바스크립트, 스타일시트와 관련된 디렉터리로 구성되어있다.

4) routes 디렉터리
이 디렉터리에는 index.js와 users.js 파일이 존재한다. 기본적으로 생성되는 파일이며, 라우트를 처리하기 위한 코드가 작성되어 있다.

5) views 디렉터리
뷰를 처리하는 파일이 위치한 곳이며, 프로젝트를 생성할 때 템플릿 엔진으로 ejs를 지정했으므로 이 디렉터리에는 확장자가 ejs인 파일이 위치한다.

• 프로젝트의 기본 파일

1) bin/www에 위치한 www 파일
이 파일은 확장자가 존재하지 않지만 내부는 자바스크립트로 작성된 노드 파일이다. 이 파일에서는 기본적으로 노드를 실행할 포트를 설정하고 웹서버를 생성하는 코드가 작성되어있다.

```
/* 사용할 모듈지정 */
var app = require('../app');
var debug = require('debug')('web:server');
var http = require('http');

/* 환경 설정에 지정된 포트가 있다면 이를 사용하고 없다면 지정한 포트(3000)를 사용하도록 설정 */
var port = normalizePort(process.env.PORT || '3000');
app.set('port', port);

/*  HTTP 서버 생성 */
var server = http.createServer(app);

/* 지정된 포트 상에서 응답 대기하도록 설정 */
server.listen(port);
```

2) app.js

이 파일은 웹 애플리케이션을 위한 기본적인 설정을 가지고 있는 파일이다. 모듈을 로딩하고 템플릿 엔진을 설정하며, 라우트를 설정한다. 파일 상단 부분에는 사용할 모듈을 로딩 하는 코드가 작성되어있다. 외부 모듈을 해당 파일에서 사용하고 싶다면 require() 함수를 호출해야 한다.

```
var express = require('express');
var path = require('path');
var cookieParser = require('cookie-parser');
var logger = require('morgan');
```

라우트 코드를 로딩 하는 코드이다 require() 함수를 사용하며, 로딩 한 라우트 함수들을 지정된 변수로 사용할 수 있게 해 준다.

```
var indexRouter = require('./routes/index');
var usersRouter = require('./routes/users');
```

익스프레스 객체를 app 변수로 선언한다. 앞으로 app을 통해 익스프레스 함수를 호출할 수 있게 된다.

```
var app = express();
```

익스프레스에서 사용할 템플릿 엔진을 설정하는 코드이다. __dirname은 현재 디렉터리를 의미하며 path.join() 함수는 경로를 연결하는 기능을 한다. 그래서 현재 디렉터리에 있는 views 디렉터리를 의미한다. 그리고 app.set() 함수는 익스프레스의 환경을 설정하는 함수이다. 그래서 첫 번째 라인은 경로를 두 번째 라인은 템플릿 엔진의 종류를 설정한다. 참고로 app.get() 함수도 정의되어 있으며, 이 함수는 값을 반환한다.

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

app.use() 함수는 지정된 인자를 실행하는 함수이다. 여기서는 각각의 모듈을 사용하도록 설정하고 있다.

```
app.use(logger('dev'));
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));
```

라우트를 설정하는 코드이다. 여기서 말하는 라우트는 url 경로의 뒷부분을 의미한다. 그래서 '/'와 관련된 라우트는 routes 폴더의 index 파일에 설정된 라우트 함수를 통해 처리되며 '/users'와 관련된 라우트는 users 파일에 작성된 라우트 함수를 통해 처리된다.

```
app.use('/', indexRouter);
app.use('/users', usersRouter);
```

3) /routes/index.js

app.js에서 라우트를 처리할 때는 app.get()나 app.post() 함수를 사용하며 별도 파일에서 라우트 함수를 작성할 때는 express.Router() 함수를 통해 호출하며 별도 파일에서 함수를 사용할 수 있도록 module.export=router를 추가해야 한다. index.js에 작성된 router.get() 함수는 URL 경로가 '/'일 때 호출되는 함수이며 실제로 호출되어 실행될 코드는 function(req, res, next){ } 함수에 작성하면 된다. req는 HTTP 요청 객체에 대한 정보이며, res는 HTTP 응답 객체에 대한 정보를 가지고 있다.

```
var express = require('express');
var router = express.Router();

/* get home page. */
router.get('/', function(req, res, next) {
    res.render('index', { title: 'Express' });
});

module.exports = router;
```

4) /db.js

데이터베이스를 연동하는 방법은 두 가지가 있다. 첫 번째는 매번 데이터베이스의 커넥션을 얻어서 사용하는 방법이고, 두 번째는 커넥션 풀을 생성해서 커넥션을 미리 생성한 후에 하나씩 꺼내서 사용하는 방법이다. 커넥션 생성 비용은 매우 크므로 가능하면 커넥션 풀을 사용하는 것이 좋다. 이 파일에는 두 개의 함수가 있다. connect는 데이터베이스 커넥션 풀을 생성하는 함수이고, get은 생성한 커넥션 풀을 반환하는 함수이다. get 함수를 통해 풀을 반환받은 후에 데이터베이스에 질의를 실행할 수 있다.

```
var mysql = require('mysql');
var conn;
exports.connect=function() {
    conn=mysql.createPool({
        connectionLimit:100,
        host:'localhost',
        user:'root',
        password:'1234',
        database:'webdb'
    });
}

exports.get=function(){
    return conn;
};
```

connect 함수는 커넥션 풀을 생성하는 함수이므로 한 번만 호출하면 된다. 그러므로 app.js에서 db.js 모듈을 로딩한 후에 connect 함수를 다음처럼 호출하면 된다.

```
var db = require('./db');
db.connect(function(err){
    if(err){
        console.log('Unable to connect to MySQL.');
        process.exit(1);
    }
});
```

라우트 파일에서 get함수를 호출한 후에 query 함수를 호출해서 질의를 실행한다.

```
var express = require('express');
var router = express.Router();
var db = require('../db');

router.get("/list.json", function(req, res) {
    var query="%" + req.query.keyword + "%";
    var sql="select * from userinfo where id like ?";
    db.get().query(sql, [query], function(err, rows){
        if(err) return res.sendStatus(400);
        res.status(200).json(rows);  //res.status(200).send(rows) 또는 res.send(rows)
    });
});

router.post("/insert", function(req, res){
    var id=req.body.id;
    var password=req.body.password;
    var name=req.body.name;
    var sql="insert into userinfo(id, password, name) values(?, ?, ?)";
    db.get().query(sql, [id, password, name], function(err, result){
        if(err) return res.sendStatus(400);
        res.sendStatus(200);
    });
});

module.exports = router;
```

• Oracle DB 접속

1) 오라클 사이트로 가서 Oracle Instant Client Downloads 페이지에서 2가지 종류의 파일을 받아야 한다.

```
https://www.oracle.com/database/technologies/instant-client.html
```

- SDK Package 라고 불리우는 압축 파일(파일 이름이 instantclient-sdk-...로 시작한다.)
- Basic Package 라고 불리우는 압축 파일(파일 이름이 instantclient-basic-..로 시작한다.)

2) c:/oraclexe/client라는 폴더를 생성한 후 압축이 풀린 2개의 디렉터리의 내용을 이곳으로 옮긴다.
3) 환경변수에 c:/oraclxe/client 폴더를 추가해준다. 추가 후 가장 위쪽으로 이동시킨다.
4) oracledb를 설치한다.

```
npm install oracledb
```

5) /db.js 파일을 생성후 아래 내용을 입력한다.

```
var oracledb = require('oracledb');
oracledb.outFormat = oracledb.OBJECT;

var conn;
exports.connect = function() {
    oracledb.getConnection({
        user:"system",
        password:"1234",
        connectString:"localhost/xe"
    },function(err,con){
        if(err){
            console.log("접속이 실패했습니다.",err);
        }
        conn = con;
    });
}

exports.get=function(){
    return conn;
};
```

6) /app.js 파일에 아래 내용을 추가한다.

```
var db = require('./db');
db.connect(function(err){
    if(err){
        console.log('Unable to connect to Oracle');
        process.exit(1);
    }
});
```

7) 라우트 파일에서 get함수를 호출한 후에 query 함수를 호출해서 질의를 실행한다.

```
var express = require('express');
var router = express.Router();
var db = require("../db");

router.get('/', function(req, res) {
    db.get().execute('select * from students', function(err, result){
        res.send(result.rows);
    });
});

module.exports = router;
```

- **사용자 관리 프로그램**

[app.js]

```
...
app.use('/', indexRouter);
app.use('/users', usersRouter);
app.use('/user', require('./routes/user'));
...
```

[routes]-[user.js]

```
var express = require('express');
var router = express.Router();
var db = require("../db");

router.get('/', function(req, res, next) {
  var sql="select * from userinfo";
  db.get().query(sql, function(err, rows){
      if(err) return res.sendStatus(400);
      console.log(rows);
      res.render('user/list', {users: rows});
  });
});

router.get('/insert', function(req, res, next) {
    res.render('user/insert', {title: "[사용자 등록]"});
});

//사용자 등록
router.post("/insert", function(req, res){
    var id=req.body.id;
    var password=req.body.password;
    var name=req.body.name;
    console.log("id:" + id + ",password:" + password + ",name:" + name);
    var sql="insert into userinfo(id, password, name) values(?, ?, ?)";
    db.get().query(sql, [id, password, name], function(err, result){
        if(err) return res.sendStatus(400);
        res.redirect('/user')
    });
});

module.exports = router;
```

[views]-[user]-[list.ejs]

```
<html>
    <head>
        <title>사용자목록</title>
    </head>
    <boyd>
        <hl>[사용자 목록]</hl>
        <ul>
        <% if(users == null || users.length == 0){ %>
            <li>등록된 사용자가 없습니다.</li>
        <% } %>
        <% users.forEach(function(user) {%>
            <li><%=user.id%> (<%=user.name%>) </li>
        <% }); %>
        </ul>
        <button onClick="location.href='/user/insert'">입력</button>
    </boyd>
</html>
```

[views]-[user]-[inser.ejs]

```
<html>
    <head>
        <title><%=title%></title>
    </head>
    <body>
        <hl><%=title%></hl>
        <form action="insert" method="post">
            <table border=l width=500>
                <tr> <td width=l00>아이디</td><td><input type="text" name="id"></td></tr>
                <tr><td width=l00>비밀번호</td><td><input type="password" name="password"></td></tr>
                <tr><td width=l00>이름</td><td><input type="name" name="name"></td></tr>
            </table>
            <input type="submit" value="저장"/>
            <input type="reset" value="취소"/>
            <input type="button" value="목록" onclick="location.href='/user'"/>
        </form>
    </body>
</html>
```

[app.js]

```
...
app.use('/userinfo', require('./routes/userinfo'));
...
```

[routes]-[userinfo.js]

```
var express = require('express');
var router = express.Router();
var db = require('../db');

router.get('/', function(req, res, next) {
    res.render('userinfo', { title: '사용자관리' });
});

//전체사용자 출력
router.get("/list.json", function(req, res) {
    var query="%" + req.query.keyword + "%";
    console.log("query:" + query);
    var sql="select * from userinfo where id like ?";
    db.get().query(sql, [query], function(err, rows){
        if(err) return res.sendStatus(400);
        //console.log("rows:" + JSON.stringify(rows) + ":" + row.length);
        res.status(200).json(rows);
    });
});

//사용자 등록
router.post("/insert", function(req, res){
    var id=req.body.id;
    var password=req.body.password;
    var name=req.body.name;
    console.log("id:" + id + ",password:" + password + ",name:" + name);
    var sql="insert into userinfo(id, password, name) values(?, ?, ?)";
    db.get().query(sql, [id, password, name], function(err, result){
        if(err) return res.sendStatus(400);
        res.sendStatus(200);
    });
});
```

[views]-[userinfo.js]

```
<html>
    <head>
        <title><%=title%></title>
        <script src="http://code.jquery.com/jquery-3.1.1.min.js"></script>
    </head>
    <body>
        <h1>[사용자등록]</h1>
        <form>
            <table border=1 width=500>
                <tr>td>아이디</td><td><input type="text" id="id"></td></tr>
                <tr><td>비밀번호</td><td><input type="password" id="password"></td></tr>
                <tr><td>이름</td><td><input type="text" id="name"></td></tr>
            </table>
            <input type="button" value="저장" id="btnSave"/>
            <input type="reset"  value="취소"/>
        </form>
        <hr/>
        <h1>[사용자목록]</h1>
        <table border=1 width=500 id="tbl"></table>
    </body>
    <script>
        getList();
        $("#btnSave").on("click", function(){
            var id=$("#id").val();
            var password=$("#password").val();
            var name=$("#name").val();
            $.ajax({
                type: "post",
                url:"/userinfo/insert",
                data:{"id": id, "password":password, "name":name},
                success:function(){
                    alert("저장되었습니다!");
                    getList();
                },
                error:function(){
                    alert("저장을 실패했습니다!");
                }
            });
        });
        function getList(){
            $.ajax({
                type:"get",
                url:"/userinfo/list.json",
                datatype:"json",
                data:{"keyword": ""},
                success:function(data){
                    var str="";
                    $(data).each(function(){
                        var id=this.id;
                        var name=this.name;
                        var password=this.password;
                        str += "<tr>";
                        str += "<td>" + id + "</td>";
                        str += "<td>" + name + "</td>";
                        str += "<td>" + password + "</td>";
                        str += "</tr>";
                    });
                    $("#tbl").html(str);
                }
            });
        }
    </script>
</html>
```

- **상품관리 프로그램**

[상품목록]

[상품등록]

[상품등록]

상품코드	
상품이름	
상품가격	원
상품이미지	150 x 120

저장 취소 목록

[상품정보]

상품코드	P001
상품이름	5단서랍장
상품가격	50000
상품이미지	

수정 삭제 취소 목록

[productinfo] 테이블 생성

```
create table productinfo(
    code char(4) primary key,
    pname varchar(100) not null,
    price int,
    image varchar(50)
);
```

[productinfo] Sample 데이터 입력

```
insert into productinfo(code, pname, price, image) values('P001', '5단서랍장', 50000, 'img01.jpg');
insert into productinfo(code, pname, price, image) values('P002', '원목싱글침대', 800000, 'img02.jpg');
insert into productinfo(code, pname, price, image) values('P003', '싱글침대', 800000, 'img03.jpg');
insert into productinfo(code, pname, price, image) values('P004', '4인용 가죽 쇼파', 1800000, 'img04.jpg');
```

1) 상품목록 출력 프로그램

[app.js] 추가

```
app.use('/product', require('./routes/product'));
```

[routes]-[product.js] 생성

```
var express = require('express');
var router = express.Router();
var db = require('../db');

router.get('/', function(req, res, next) {
    res.render('product/list', { title: '상품관리' });
});

router.get("/list.json", function(req, res) {
    var sql="select * from productinfo";
    db.get().query(sql, function(err, rows){
        if(err) return res.sendStatus(400);
        res.status(200).json(rows);
    });
});
module.exports = router;
```

[views]-[product]-[list.ejs] 생성

```
<html>
    <head>
        <title>상품목록</title>
        <script src="https://code.jquery.com/jquery-3.1.1.min.js"></script>
        <script src="https://cdnjs.cloudflare.com/ajax/libs/handlebars.js/3.0.1/handlebars.js"></script>
        <style>
            #container {width:680px;background:gray; padding:10px; overflow:hidden;}
            .box {width:150px; background:pink; padding:5px; margin:5px; float:left;}
        </style>
    </head>
    <body>
        <h1>[상품목록]</h1>
        <a href="/product/insert">상품등록</a>
        <div id="container"></div>
        <script id="temp" type="text/x-handlebars-template">
            {{#each .}}
            <div class="box">
                <img src="/upload/{{image}}" width=150/>
                <div>{{code}}</div>
                <div>{{pname}}</div>
                <div>{{price}}원</div>
            </div>
            {{/each}}
        </script>
    </body>
    <script>
        getList();
        function getList(){
            $.ajax({
                type:"get",
                url:"/product/list.json",
                dataType:"json",
                success:function(data){
                    var temp=Handlebars.compile($("#temp").html());
                    $("#container").html(temp(data));
                }
            });
        }
    </script>
</html>
```

```
var formData = new FormData();
formData.append('title', title);
formData.append('price', price);
formData.append('image', $(frm.image)[0].files[0]);

$.ajax({
    type: 'post',
    url : '/product/insert',
    data: formData,
    processData: false,
    contentType: false,
    success:function(){
        alert('상품등록 성공!');
        location.href="/product";
    }
});
```

---

2) 상품등록 프로그램

```
C:₩project₩web>npm install multer
```

[routes]-[product.js] 수정

```
var multer  = require('multer')
var fs = require('fs');
var path = require('path');
/* ./public 현재프로젝트 폴더의 public 폴더 */
var uploadPath = './public/upload';
if (!fs.existsSync(uploadPath)) fs.mkdirSync(dir);

var upload = multer({
    storage: multer.diskStorage({
        destination: (req, file, done) => {
            done(null, uploadPath);
        },
        filename: (req, file, done) => {
            var ext = path.extname(file.originalname);
            done(null, path.basename(file.originalname, ext) + Date.now() + ext);
        },
    }),
    limits: { fileSize: 5 * 1024 * 1024 },
});
```

```
var multer  = require('multer')
var uploadPath = './public/upload';
var upload = multer({
    storage: multer.diskStorage({
        destination: (req, file, done) => {
            done(null, uploadPath);
        },
        filename: (req, file, done) => {
            done(null, Date.now()+'_' + file.originalname);
        },
    })
});
```

```
router.get('/insert', function(req, res, next) {
    res.render('product/insert', { title: '상품입력' });
});

router.post('/insert', upload.single('image'),function(req, res) {
    var code=req.body.code;
    var pname=req.body.pname;
    var price=req.body.price;
    var image='';
    if(req.file != null) var image=req.file.filename;
    var sql="insert into productinfo(code,pname,price,image) values(?,?,?,?)";
    db.get().query(sql, [code, pname, price, image], function(err, result){
        if(err) return res.sendStatus(400);
        res.status(200).redirect('/product');
    });
});
```

[views]-[product]-[insert.ejs] 생성

```
<html>
<head>
    <script src="http://code.jquery.com/jquery-3.1.1.min.js"></script>
    <title>상품등록</title>
    <style>
        table {border-collapse:collapse;}
        td {border:solid 1px black; height:30px;}
        .title {background:gray; color:white; text-align:center;}
    </style>
</head>
<body>
    <h1>[상품등록]</h1>
    <form name="frm" action="insert" method="post" enctype="multipart/form-data">
        <table>
            <tr><td width=100 class="title">상품코드</td><td width=500><input type="text" name="code"></td></tr>
            <tr><td width=100 class="title">상품이름</td><td width=500><input type="text" name="pname" size=50></td></tr>
            <tr><td width=100 class="title">상품가격</td><td width=500><input type="text" name="price">원</td></tr>
            <tr>
                <td width=100 class="title">상품이미지</td>
                <td width=500>
                    <img src="http://placehold.it/150x120" id="image" width=150>
                    <input type="file" name="image" accept="image/*" style="visibility:hidden;">
                </td>
            </tr>
        </table>
        <input type="submit" value="저장">
        <input type="reset" value="취소">
        <input type="button" value="목록" onClick="location.href='/product/'">
    </form>
</body>
<script>
    $("#image").on("click", function(){
        $(frm.image).click();
    });
    $(frm.image).on("change", function(e){
        var reader = new FileReader();
        reader.onload=function(e){
            $("#image").attr("src", e.target.result);
        }
        reader.readAsDataURL(this.files[0]);
    });
</script>
</html>
```

[views]-[product]-[list.ejs] 수정

```
<script id="temp" type="text/x-handlebars-template">
    {{#each .}}
    <div class="box">
        <img src="/upload/{{image}}" code={{code}} width=150/>
        <div>{{code}}</div>
        <div>{{pname}}</div>
        <div>{{price}}원</div>
    </div>
    {{/each}}
</script>
...
<script>
    $("#container").on("click", ".box img", function(){
        var code=$(this).attr("code");
        location.href='/product/read?code=' + code;
    });
</scipt>
```

[routes]-[prduct.js] 추가

```
router.get('/read', function(req, res, next) {
    var code=req.query.code;
    var sql="select * from productinfo where code=?";
    db.get().query(sql, [code], function(err, rows){
        res.render('product/read', { product:rows[0] });
    });
});
```

[views]-[product]-[read.ejs] 생성

```
<html>
<head>
    <script src="http://code.jquery.com/jquery-3.1.1.min.js"></script>
    <title>상품정보</title>
    ...
</head>
<body>
    <h1>[상품정보]</h1>
    <form name="frm" action="update" method="post" enctype="multipart/form-data">
        <table>
            <tr>
                <td width=100 class="title">상품코드</td>
                <td width=500><input type="text" name="code" value="<%=product.code%>" disabled></td>
            </tr>
            <tr>
                <td width=100 class="title">상품이름</td>
                <td width=500><input type="text" name="pname" value="<%=product.pname%>"></td>
            </tr>
            <tr>
                <td width=100 class="title">상품가격</td>
                <td width=500><input type="text" name="price" value="<%=product.price%>"></td>
            </tr>
            <tr>
                <td width=100 class="title">상품이미지</td>
                <td width=500>
                    <img src="/upload/<%=product.image%>" id="image" width=150>
                    <input type="file" name="image" accept="image/*" style="visibility:hidden;">
                </td>
            </tr>
        </table>
        <input type="submit" value="수정">
        <input type="button" value="삭제" id="btnDelete">
        <input type="reset" value="취소">
        <input type="button" value="목록" onClick="location.href='list'">
    </form>
</body>
```

3) 상품삭제 프로그램

[views]-[product]-[read.ejs] 추가

```
<script>
    $("#btnDelete").on("click", function(){
        if(!confirm("삭제하실래요?")) return;
        var code = "<%=product.code%>";
        var image = "<%=product.image%>";
        location.href="/product/delete?code="+code + "&image=" + image;
    });
    ...
</script>
```

[routes]-[product.js] 추가

```
router.get('/delete', function(req, res){
    var code=req.query.code;
    var image=req.query.image;
    var sql="delete from productinfo where code=?";
    db.get().query(sql, [code], function(err, result){
        if (image != '') {
            fs.unlink(uploadPath +'/' + image, function(err){
                if( err ) throw err;
            });
        }
        res.status(200).redirect('/product');
    });
});
```

4) 상품수정 프로그램

[views]-[product]-[read.ejs] 추가

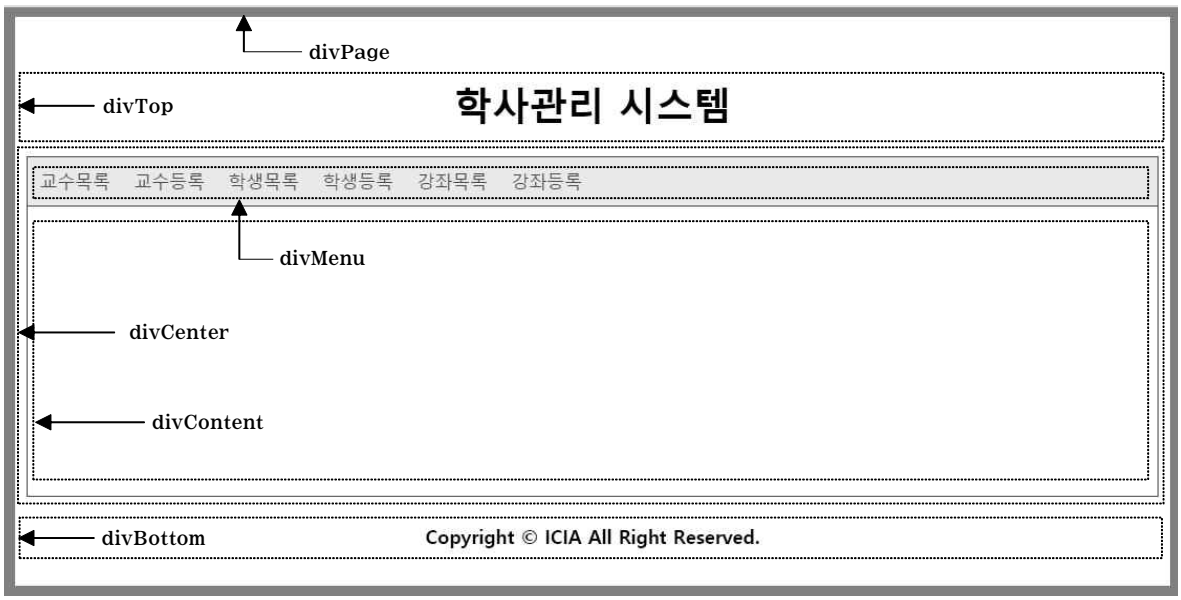
```
            <tr>
                <td width=100 class="title">상품이미지</td>
                <td width=500>
                    <img src="/upload/<%=product.image%>" id="image" width=150>
                    <input type="file" name="image" accept="image/*" style="visibility:hidden;">
                    <input type="text" name="oldImage" value="<%=product.image%>"/>
                </td>
            </tr>
<script>
    $(frm).on("submit", function(e){
        e.preventDefault();
        if(!confirm("수정하실래요?")) return;
        frm.submit();
    });
    ...
</script>
```

[routes]-[product.js] 추가

```
router.post('/update', upload.single('image'),function(req, res) {
    var code=req.body.code;
    var pname=req.body.pname;
    var price=req.body.price;
    var image=req.body.oldImage;
    if(req.file != null) {  //이미지가 변경된 경우
        fs.unlink(uploadPath +'/' + image, function(err){  //예전 이미지 삭제
            if(err) throw err;
        });
        image=req.file.filename;
    }
    var sql="update productinfo set pname=?,price=?,image=? where code=?";
    db.get().query(sql, [pname, price, image, code], function(err, result){
        if(err) return res.sendStatus(400);
        res.status(200).redirect('/product');
    });
});
```

- **로그인/로그아웃 프로그램**

1) 화면 레이아웃 프로그램



[public]-[stylesheets]-[style.css]

```
body{background:gray;}
#divPage{width:980px; background:white; box-shadow:10px 10px 10px blak; margin:auto;}
#divTop{padding:30px 0px 0px 0px; text-align:center;}
#divCenter{border:1px solid green; margin:0px 10px 0px 10px; padding-bottom:20px;}
#divMenu{overflow:hidden; background:#EAEAEA; margin:0px 0px 20px 0px; padding:10px; border-bottom:1px solid green;}
#divMenu .item{width:80px;float:left;}
#divContent{width:980px;height:500px;}
#divBottom{padding:0px 0px 10px 0px; text-align:center;}
#divHeader{width:980px; text-align:center;}
a {color: green; text-decoration: none;}
```

[views]-[index.ejs]

```
<html>
    <head>
        <title><%=title%></title>
        <link rel='stylesheet' href='/stylesheets/style.css' />
        <script src="https://code.jquery.com/jquery-3.1.1.min.js"></script>
        <script src="https://cdnjs.cloudflare.com/ajax/libs/handlebars.js/3.0.1/handlebars.js"></script>
    </head>
    <body>
        <div id="divPage">
            <div id="divTop">
                <h1>[<%=title%>]</h1>
            </div>
            <div id="divCenter">
                <div id="divMenu">
                    <div class="item"><a href="#">교수관리</a></div>
                    <div class="item"><a href="#">학생관리</a></div>
                    <div class="item"><a href="#">강좌관리</a></div>
                    <div class="item"><a href="#">수강신청</a></div>
                    <div style="float:right;"><a href="/user/login">로그인</a></div>
                </div>
                <div id="divContent">
                    <%-include(pageName)%>
                </div>
            </div>
            <div id="divBottom">
                <h4>Copyright © 인천일보아카데미 All Rights Reserved.</h4>
            </div>
        </div>
    </body>
</html>
```

[views]-[info.ejs] 생성

```
<h2>[<%=title%>]</h2>
```

[views]-[info.ejs] 생성

```
var express = require('express');
var router = express.Router();

router.get('/', function(req, res, next) {
    res.render('index', { title: '학교소개', pageName: 'info.ejs' });
});
module.exports = router;
```

2) 로그인 프로그램

[routes]-[user.js] 추가

```
router.get('/login', function(req, res, next) {
    res.render('index', {title: "로그인", pageName: "user/login.ejs" });
});

router.post('/login', function(req, res) {
    var id = req.body.id;
    var password = req.body.password;
    var sql = "select * from userinfo where id=?";
    db.get().query(sql, [id], function(err, rows) {
        if(err) return res.sendStatus(400);
        var result=0;
        if(rows[0]!=null){
            if(rows[0].password == password){
                result = 1;
            }else{
                result = 2;
            }
        }
        res.status(200).send({result: result});
    });
});
```

[views]-[user]-[login.ejs] 생성

```
<form name="frm">
    <table width=300 border=1 style="margin:0px auto; margin-top:100px;" >
        <tr><td colspan="2"><hl>로그인</hl></td></tr>
        <tr><td width=100>아이디</td><td><input type="text" name="id"/></td></tr>
        <tr><td width=100>비밀번호</td><td><input type="password" name="password"/></td></tr>
        <tr><td colspan="2"><input type="submit" value="로그인"/></td></tr>
    </table>
</form>
<script>
    $(frm).on("submit", function(e){
        var id=$(frm.id).val();
        var password=$(frm.password).val();
        e.preventDefault();
        $.ajax({
            type: "post",
            url: "/user/login",
            data: {"id":id, "password":password},
            dataType:"json",
            success:function(data){
                if(data.result==0){
                    alert("아이디가 존재하지 않습니다!");
                }else if(data.result==1){
                    location.href='/';
                }else{
                    alert("비밀번호가 일치하지 않습니다!");
                }
            }
        });
    });
</script>
```

3) 아이디 세션 저장 및 로그아웃 프로그램

```
npm install express-session
```

[app.js] 추가

```
var session = require('express-session');
...
var app = express();
app.use(session({
    secret: 'mykey',
    resave: fals,  //세션 데이터가 바뀌기 전에는 세션 저장소에 값을 저장하지 않음
    saveUninitialized:true  //세션이 필요한 경우에만 구동
}));
```

[routes]-[user.js] 추가

```
router.post('/login', function(req, res) {
    var id = req.body.id;
    var password = req.body.password;
    var sql = "select * from userinfo where id=?";
    db.get().query(sql, [id], function(err, rows) {
        if(err) return res.sendStatus(400);
        var result=0;
        if(rows[0]!=null){
            if(rows[0].password == password){
                result = 1;
                req.session.userid=id;
            }else{
                result = 2;
            }
        }
        res.status(200).send({result: result});
    });
});
```

[routes]-[index.js]

```
router.get('/', function(req, res, next) {
    res.render('index', {title:"학교소개", pageName: "info.ejs", userid:req.session.userid });
});
```

[routes]-[user.js]

```
router.get('/login', function(req, res, next) {
    res.render('index', {title: "로그인", pageName: "user/login.ejs", userid:req.session.userid });
});
```

[views]-[index.ejs] 추가

```
<div id="divMenu">
    ...
    <%if(userid==null) {%>
        <div style="float:right;"><a href="/user/login">로그인</a></div>
    <%}else {%>
        <div style="float:right;">
            <a href="/user/logout">로그아웃</a>
            <span><b><%=userid%></b></span>
        </div>
     <%}%>
</div>
```

[routes]-[user.js] 추가

```
router.get('/logout', function(req, res){
    req.session.destroy();
    res.clearCookie('userid');
    res.redirect('/');
});
```

4) 쿠키 저장 프로그램

[views]-[user]-[login.js] 추가

```
    <tr>
        <td colspan="2">
            <input type="submit" value="로그인"/>
            <input type="checkbox" name="chkLogin"/>아이디저장
         </td>
    </tr>
    ...
    <script>
      $(frm).on("submit", function(e){
            var id=$(frm.id).val();
            var password=$(frm.password).val();
            var chkLogin = $(frm.chkLogin).is(":checked") ? 1 : 0;
            e.preventDefault();
            $.ajax({
                type: "post",
                url: "/user/login",
                data: {"id":id, "password":password, "chkLogin":chkLogin},
                dataType:"json",
                success:function(data){
                    if(data.result==0){
                        alert("아이디가 존재하지 않습니다!");
                    }else if(data.result==1){
                        location.href='/';
                    }else{
                        alert("비밀번호가 일치하지 않습니다!");
                    }
                }
            });
      });
    </script>
```

[routes]-[user.js] 추가

```
router.post('/login', function(req, res) {
    var id = req.body.id;
    var password = req.body.password;
    var chkLogin = req.body.chkLogin;

    var sql = "select * from userinfo where id=?";
    db.get().query(sql, [id], function(err, rows) {
        if(err) return res.sendStatus(400);
        var result=0;
        if(rows[0]!=null){
            if(rows[0].password == password){
                result = 1;
                req.session.userid=id;
                if(chkLogin==1) res.cookie('userid', id, { maxAge: 1000 * 60 * 60 *24 });
            }else{
                result = 2;
            }
        }
        res.status(200).send({result: result});
    });
});
```

[routes]-[index.js] 추가

```
router.get('/', function(req, res, next) {
    if(req.cookies.userid) req.session.userid=req.cookies.userid;
    res.render('index', {title:"학교소개", pageName: "info.ejs", userid:req.session.userid});
});
```

• Firebase를 이용한 당근마켓

• 프로젝트 생성 및 설정

1) 프로젝트를 생성하고 필요한 라이브러리를 설치한다.

```
C:₩project>express -e carrot
```

2) 기본 라이브러리를 설치한다.

```
C:₩project₩carrot>npm install
```

3) firebase에 필요한 라이브러리를 설치한다.

```
C:₩project₩carrot>npm install firebase@8.10.0
```

[carrot]-[app.js]

```
app.use('/', indexRouter);
app.use('/users', usersRouter);
app.use('/products', require('./routes/products'));
```

[carrot]-[routes]-[firebase.js]

```
var {initializeApp} = require('firebase/app');

const firebaseConfig = {
    apiKey: "AIzaSyC3eWXVR3Uds_vBvbmMBetbauY-zocpy_c",
    authDomain: "fir-ell45.firebaseapp.com",
    databaseURL: "https://fir-ell45-default-rtdb.firebaseio.com",
    projectId: "fir-ell45",
    storageBucket: "fir-ell45.appspot.com",
    messagingSenderId: "990122132919",
    appId: "1:990122132919:web:8bb74676dcb18006969432",
    measurementId: "G-43EW86YZ74"
};
firebase=initializeApp(firebaseConfig);

module.exports = firebase;
```

[carrot]-[routes]-[index.js]

```
var express = require('express');
var router = express.Router();
var loginCheck = require('./loginCheck');

router.get('/', function(req, res, next) {
    res.render('index', { title: '당근마켓', pageName: 'products/list.ejs', 'email':loginCheck.getEmail()} );
});

module.exports = router;
```

[carrot]-[routes]-[loginCheck.js]

```
var firebase = require('./firebase');
require('firebase/auth');

module.exports.getEmail = () =>{
    const user = firebase.auth().currentUser;
    var email='';
    if (user !== null) email = user.email;
    return email;
}
```

[carror]-[stylesheets]-[style.css]

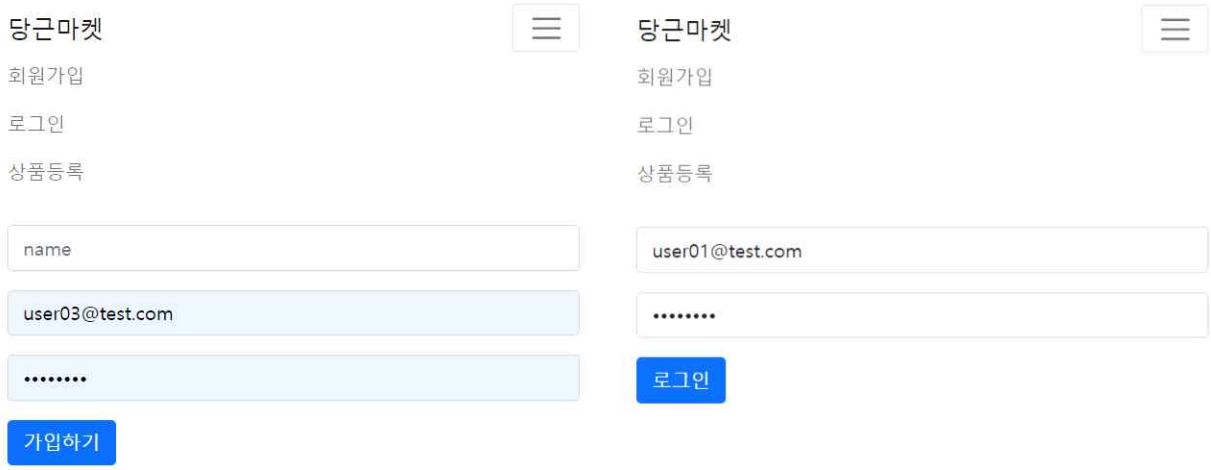
```
body {
    font: "Lucida Grande", Helvetica, Arial, sans-serif;
}

a {
    color: #00B7FF;
    text-decoration: none;
}
```

[carrot]-[views]-[index.ejs]

```
<head>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="...">
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="..."></script>
    <script src="https://code.jquery.com/jquery-3.1.1.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/handlebars.js/3.0.1/handlebars.js"></script>
    <link rel="stylesheet" href="/stylesheets/notice.css"/>
    <link rel="stylesheet" href="/stylesheets/style.css"/>
    <script src="https://www.gstatic.com/firebasejs/8.10.0/firebase.js"></script>
    <title><%= title %></title>
    <script>
        const firebaseConfig = {
            ...
        };
        firebase.initializeApp(firebaseConfig);
        const realdb=firebase.database();
        const db=firebase.database();
  </script>
</head>
<body>
    <nav class="navbar navbar-expand-lg navbar-light bg-light">
        <div class="container-fluid">
            <a class="navbar-brand" href="/"><%=title%></a>
            <a href="#" id="email" style="padding-right:80px;"><%=email%></a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav"
                    aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav">
                    <li class="nav-item"><a class="nav-link" href="/users/join">회원가입</a></li>
                    <li class="nav-item">
                        <% if(email != ""){ %>
                            <a class="nav-link" href="/users/logout">로그아웃</a>
                        <% }else{ %>
                            <a class="nav-link" href="/users/login">로그인</a>
                        <% } %>
                    </li>
                    <li class="nav-item"><a class="nav-link" href="#" id="register">상품등록</a></li>
                </ul>
            </div>
        </div>
    </nav>
    <div id="content">
        <%-include(pageName)%>
    </div>
</body>
<script>
    $("#register").on("click", function(e){
        e.preventDefault();
        var email=$("#email").html();
        if(email=="" || email==null) {
            location.href="/users/login";
        }else {
            location.href="/products/insert";
        }
    });
</script>
```

- **회원가입 및 로그인/로그아웃 작성**



[carrot]-[routes]-[users.js]

```
var express = require('express');
var router = express.Router();
var loginCheck = require('./loginCheck');
var firebase = require('./firebase');
require('firebase/auth');

router.get('/join', function(req, res, next) {
    res.render("index", {title: '당근마켓', pageName: 'users/join.ejs', email:loginCheck.getEmail()});
});

router.post('/join', function(req, res){
    var email = req.body.email;
    var password = req.body.password;
    firebase.auth().createUserWithEmailAndPassword(email, password).then((result) => {
        var user = result.user;
        res.send('success');
    })
    .catch((error) => {
        var errorMessage = error.message;
        res.send('fail');
    });
});

router.get('/login', function(req, res, next) {
    res.render("index", {title: '당근마켓', pageName: 'users/login.ejs', email:loginCheck.getEmail()});
});

router.post('/login', function(req, res){
    var email = req.body.email;
    var password = req.body.password;

    firebase.auth().signInWithEmailAndPassword(email, password).then((userCredential) => {
       var user = userCredential.user;
       res.send("success");
    })
    .catch((error) => {
       var errorMessage = error.message;
       res.send('fail');
    });
});

router.get('/logout', function(req, res){
    firebase.auth().signOut().then(()=>{
        res.redirect('/');
    });
});

module.exports = router;
```

[carrot]-[views]-[users]-[join.ejs]

```
<div class="container mt-3">
    <div class="mb-3">
        <input type="text" class="form-control" placeholder="name" id="txt_name"/>
    </div>
    <div class="mb-3">
        <input type="email" class="form-control" placeholder="email" id="txt_email"/>
    </div>
    <div class="mb-3">
        <input type="password" class="form-control" placeholder="password" id="txt_password"/>
    </div>
    <button type="submit" class="btn btn-primary" id="btn_join">가입하기</button>
</div>
<script>
    $("#btn_join").on("click", function(){
        var email=$("#txt_email").val();
        var password=$("#txt_password").val();
        var name=$("#txt_name").val();
        $.ajax({
            type: "post",
            url: "/users/join",
            data: { "email": email, "password": password, "name": name },
            success: function(){
                alert("회원가입성공!");
            }
        })
    });
</script>
```

[carrot]-[views]-[users]-[login.ejs]

```
<div class="container mt-3">
    <div class="mb-3">
        <input type="email" class="form-control" placeholder="email" id="txt_email" value="user01@test.com"/>
    </div>
    <div class="mb-3">
        <input type="password" class="form-control" placeholder="password" id="txt_password" value="12341234"/>
    </div>
    <button type="submit" class="btn btn-primary" id="btn_login">로그인</button>
</div>
<script>
    $("#btn_login").on("click", function(){
        var email=$("#txt_email").val();
        var password=$("#txt_password").val();
        $.ajax({
            type: "post",
            url: "/users/login",
            data: { "email": email, "password": password },
            success: function(data){
                if(data=="success"){
                    location.href='/';
                }else{
                    alert('아이디가 존재하지 않거나 비밀번호가 일치하지 않습니다!');
                    $("#txt_email").focus();
                }
            }
        });
    });
</script>
```

- **상품등록, 상품정보, 상품수정 작성**

[carrot]-[views]-[products]-[insert.ejs]

```
<div class="container mt-3">
    <input type="text" class="form-control mt-2" id="txt_name" placeholder="Title">
    <textarea class="form-control mt-2" id="txt_content" rows="3" placeholder="Content"></textarea>
    <input type="text" class="form-control mt-2" id="txt_price" placeholder="Price">
    <input type="file" class="form-control mt-2" id="image">
    <button class="btn btn-danger mt-3" id="btn_send">올리기</button>
</div>

<script>
    $("#btn_send").on("click", function(){
        var email = $("#email").html();
        var name = $("#txt_name").val();
        var price = $("#txt_price").val();
        var content =$("#txt_content").val();
        var file = $("#image")[0].files[0];

        if($("#image").val()=="") {
            alert("첨부할 이미지를 선택하세요!");
            return;
        }

        var formData = new FormData();
        formData.append("file", file);
        formData.append("email", email);
        formData.append("name", name);
        formData.append("price", price);
        formData.append("content", content);

        $.ajax({
            type: "post",
            url: "/products/insert",
            enctype: 'multipart/form-data',
            processData:false,
            contentType:false,
            data:formData,
            success: function(){
                alert("업로드성공!");
                location.href="/";
            }
        });
    });
</script>
```

C:₩project₩carrot>npm install multer

C:₩project₩carrot>npm install moment

[carrot]-[routes]-[products.js]

```
var express = require('express');
var router = express.Router();
var moment = require('moment');

var loginCheck = require('./loginCheck');
var firebase=require('./firebase');

require('firebase/firestore');
var db = firebase.firestore();

require('firebase/database');
var realdb = firebase.database();

//상품등록 페이지이동
router.get('/insert', function(req, res){
  res.render('index', {
      title:'당근마켓',
      pageName:'products/insert.ejs',
      email:loginCheck.getEmail()
  });
});

//파일업로드 설정
var multer = require('multer');
var upload = multer({ storage: multer.memoryStorage() });

require("firebase/storage");
var storage = firebase.storage();

//상품데이터 저장및 파일업로드
router.post('/insert', upload.single("file"), function(req, res){
    //파일업로드
    var file = req.file;
    global.XMLHttpRequest = require("xhr2");
    var fileName = Date.now() + '_' + file.originalname;
    var storageRef = storage.ref().child('image/' + fileName);
    storageRef.put(file.buffer).then((snapshot)=>{
        snapshot.ref.getDownloadURL().then(url => {
            //데이터 저장
            db.collection('product').doc().set({
                name: req.body.name,
                price: parseInt(req.body.price),
                date: moment(new Date()).format("YYYY-MM-DD HH:mm:ss"),
                content: req.body.content,
                cnt_chat: 0,
                url: url,
                image: fileName,
                email: loginCheck.getEmail()
            });
            res.send('success');
        });
    });
});
```

당근마켓 user01@test.com

4인용 가죽쇼파 : 1250000원

user01@test.com

등록일:2021-11-08 12:45:30

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

채팅으로 거래하기 상품정보수정

당근마켓 user01@test.com

4인용 가죽쇼파

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an

1250000

파일 선택 선택된 파일 없음

올리기

[carrot]-[views]-[products]-[read.ejs]

```
<link rel="stylesheet" href="/stylesheets/products.css"/>
<div id="product">
    <div class="name" id="title"><%=product.data.name%> : <%=product.data.price%>원</div>
    <div id="id" style="display: none;"><%=product.id%></div>
    <div><img src="<%=product.data.url%>"/></div>
    <div class="email"><%=product.data.email%></div>
    <div class="date">등록일:<%=product.data.date%></div>
    <div class="content"><%=product.data.content%></div>
    <div>
        <button class="btn btn-danger mt-3" id="btn_chat">채팅으로 거래하기</button>
        <% if(email==product.data.email){ %>
            <button class="btn btn-primary mt-3" id="btn_update">상품정보수정</button>
        <% } %>
    </div>
</div>
<script>
    var email=$("#email").html();
    var id=$("#id").html();
    var title=$("#title").html();

    //채팅으로 거래하기 버튼을 클릭한 경우
    $("#btn_chat").on("click", function(){
        if(email=="" || email==null){
            location.href="/users/login";
        }else{
            location.href='/products/chat?id=' + id + "&title=" + title;
        }
    });

    //상품수정 버튼을 클릭한 경우
    $("#btn_update").on("click", function(){
        location.href="/products/update?id=" + id;
    });
</script>
```

[carrot]-[routes]-[products.js]

```
//상품상세 페이지
router.get('/read', function(req, res){
    var id=req.query.id;
    db.collection("product").doc(id).get().then((doc) => {
        res.render('index', {
            title:'당근마켓',
            pageName:'products/read.ejs',
            email:loginCheck.getEmail(),
            product: {id: id, data: doc.data()}
        });
    });
});
```

[carrot]-[views]-[products]-[update.ejs]

```
<div class="container mt-3">
    <input type="hidden" id="id" value="<%=product.id%>"/>
    <input type="text" class="form-control mt-2" id="txt_name" value="<%=product.data.name%>">
    <textarea class="form-control mt-2" id="txt_content" rows="3"><%=product.data.content%></textarea>
    <input type="text" class="form-control mt-2" id="txt_price" value="<%=product.data.price%>">
    <img src="<%=product.data.url%>" style="width:300px;border-radius:10px;margin:10px;"/>
    <input type="file" class="form-control mt-2" id="image">
    <button class="btn btn-danger mt-3" id="btn_send">올리기</button>
</div>

<script>
    $("#btn_send").on("click", function(){
        if (!confirm('상품정보를 수정하실래요?')) return;

        var id = $("#id").val();
        var name = $("#txt_name").val();
        var price = $("#txt_price").val();
        var content =$("#txt_content").val();
        var file = $("#image")[0].files[0];

        var formData = new FormData();
        formData.append("file", file);
        formData.append("id", id);
        formData.append("name", name);
        formData.append("price", price);
        formData.append("content", content);

        $.ajax({
            type: "post",
            url: "/products/update",
            enctype: 'multipart/form-data',
            processData:false,
            contentType:false,
            data:formData,
            success: function(){
                alert("상품정보 수정 성공!");
                location.href="/";
            }
        });
    });
</script>
```

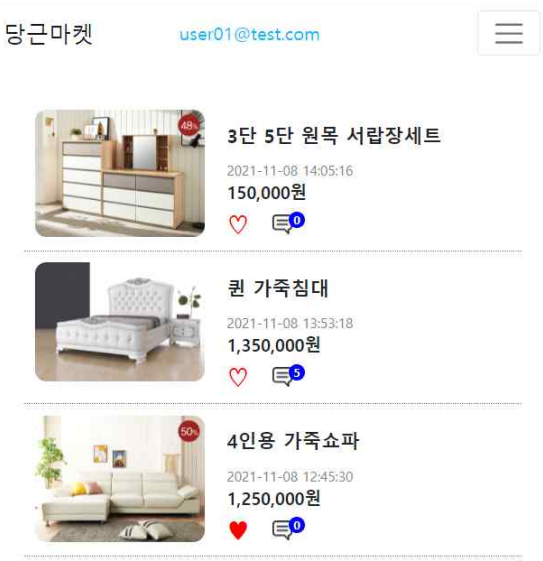
```
C:₩project₩carrot>npm install xhr2
```

[carrot]-[routes]-[products.js]

```
//상품수정 페이지
router.get("/update", function(req, res){
    var id = req.query.id;
    db.collection("product").doc(id).get().then((doc) => {
        res.render('index', {
            title:'당근마켓',
            pageName:'products/update.ejs',
            email:loginCheck.getEmail(),
            product: {id: id, data: doc.data()}
        });
    });
});

//상품데이터 수정 및 파일업로드
router.post('/update', upload.single("file"), function(req, res){
    var id = req.body.id;
    var file = req.file;
    var docRef=db.collection("product").doc(id);
    //수정할 이미지가 존재하는 경우
    if(file != null){
        global.XMLHttpRequest = require("xhr2");
        var fileName = Date.now() + '_' + file.originalname;
        var storageRef = storage.ref().child('image/' + fileName);
        storageRef.put(file.buffer).then((snapshot)=>{
            snapshot.ref.getDownloadURL().then(url => {
                var url = url;
                docRef.get().then(doc => {
                    //예전이미지삭제
                    storage.ref().child('image/' + doc.data().image).delete().then(()=>{
                        //상품정보 수정
                        docRef.update({
                            name:req.body.name,
                            price:req.body.price,
                            content:req.body.content,
                            image:fileName,
                            url:url
                        });
                        res.send('success');
                    });
                });
            });
        });
    //수정할 이미지가 없는 경우
    }else{
        docRef.get().then(doc => {
            //상품정보 수정
            docRef.update({
               name: req.body.name,
               price: req.body.price,
               content: req.body.content
           });
        });
    }
    res.send('success');
});
```

- **상품목록 작성 및 좋아요 기능 작성**



[carrot]-[routes]-[products.js]

```
//상품목록 데이터생성
router.get('/list.json', function(req,res){
    var rows = [];
    var loginEmail = loginCheck.getEmail().replace('.','');
    db.collection('product').orderBy("date", "desc").get().then(snapshot=>{
        snapshot.forEach((doc)=>{
            var id=doc.id;
            var row ={
                'id':id,
                'email':doc.data().email,
                'name':doc.data().name,
                'price':doc.data().price.toString().replace(/\B(?<!\.\d*)(?=(\d{3})+(?!\d))/g, ","),
                'date':doc.data().date,
                'url':doc.data().url,
                'cnt_chat':doc.data().cnt_chat,
                'like':0 };
            if(loginEmail == null || loginEmail == ''){
                rows.push(row);
            }else{
                realdb.ref('likes/' + id + '/' + loginEmail).on('value', snap => {
                    if(snap.exists()) row.like = 1;
                    rows.push(row);
                });
            }
        });
        res.send(rows);
    });
});

//상품 좋아요 추가
router.post("/like/insert", function(req, res){
    var id=req.body.id;
    var loginEmail=loginCheck.getEmail().replace('.','');
    var strDate=moment(new Date()).format('YYYY-MM-DD HH:mm:ss')
    realdb.ref('likes/' + id + '/' + loginEmail).set({ date: strDate });
    res.send('success');
});

//상품 좋아요 삭제
router.post("/like/delete", function(req, res){
    var id=req.body.id;
    var loginEmail=loginCheck.getEmail().replace('.','');
    realdb.ref('likes/' + id + '/' + loginEmail).remove();
    res.send('success');
});
```

[carrot]-[public]-[stylesheets]-[products.css]

```
#products {
    margin:10px; padding:20px; }

.product {
    overflow: hidden; border-bottom: 1px dotted gray; padding: 10px; }

.thumbnail {
    float: left; margin-right: 20px; }

.thumbnail img {
    width: 150px; border-radius: 10px; }

.info { float:left; }

.name {
    font-size: 17px; font-weight: bold; margin: 10px 0px; }

.price {
    font-size: 15px; font-weight: bold; }

.date {
    font-size: 12px; color: gray; }

.like{
    color:red; cursor: pointer; font-size: 20px; margin-right: 20px; }

#product { margin: 20px; }

#product div { margin: 10px 0px; }

#product img {
    width: 400px; border-radius: 20px; margin-bottom: 20px; }

.content {
    margin:20px 0px; padding: 20px 0px;
    border-top: 1px dotted gray; border-bottom: 1px dotted gray;
    font-size: 12px; }

.email { font-size: 15px; }
```

[carrot]-[public]-[stylesheets]-[notice.css]

```
#bell {
    display:inline-block;
    position:absolute;
    background-image: url('/images/chat.png');
    background-repeat: no-repeat;
    background-size: 100%;
    width: 18px;
    height: 18px;
    cursor: pointer;
    margin-top:8px;
}

#count {
    display:inline-block;
    position: relative;
    top: -10px;
    left: 15px;
    z-index: 3;
    padding:0px 4px;
    background: blue;
    border-radius: 15px;
    text-align: center;
    font-size: 10px;
    color: white;
    font-weight: bold;
}
```

[carrot]-[views]-[products]-[list.ejs]

```
<link rel="stylesheet" href="/stylesheets/products.css"/>
<div id="products"></div>
<script id="temp" type="text/x-handlebars-template">
    {{#each .}}
        <div class="product">
            <div class="thumbnail">
                <a href="/products/read?id={{id}}"><img src="{{url}}"/></a>
            </div>
            <div class="info">
                <div class="name">{{name}}</div>
                <div class="date" style="color:gray;">{{date}}</div>
                <div class="price">{{price}}원</div>
                <div>
                    <span class="like" pid={{id}}>{{printLike like}}</span>
                    <span id="bell" pid={{id}}><span id="count">{{cnt_chat}}</span></span>
                </div>
            </div>
        </div>
    {{/each}}
</script>
<script>
    Handlebars.registerHelper("printLike", function(like){
        if(like == 0){
            return "♡";
        }else{
            return "♥";
        }
    });
</script>
<script>
    var email=$("#email").html();
    getList();
    function getList(){
        $.ajax({
            type:"get",
            url:"/products/list.json",
            dataType:"json",
            success:function(data){
                var temp=Handlebars.compile($("#temp").html());
                $("#products").html(temp(data));
            }
        });
    }
    //좋아요 버튼을 클릭한 경우
    $("#products").on("click", ".product .like", function(){
        if(email=="" || email==null){
            location.href="users/login";
            return;
        }
        var id=$(this).attr("pid");
        if($(this).html()=="♡"){
            $(this).html("♥");
            $.ajax({
                type: "post",
                url: "/products/like/insert",
                data: {"id": id},
                success: function(){ alert("좋아요 추가!"); }
            });
        }else{
            $(this).html("♡");
            $.ajax({
                type: "post",
                url: "/products/like/delete",
                data: {"id": id},
                success: function(){ alert("좋아요 삭제!"); }
            });
        }
    });
</script>
```

- **채팅 기능 작성**


```
//채팅 페이지
router.get('/chat', function(req, res) {
    var id = req.query.id;
    db.collection("product").doc(id).get().then((doc) => {
        res.render('index', {
            title: '당근마켓',
            pageName: 'products/chat.ejs',
            email:loginCheck.getEmail(),
            id: id,
            data:doc.data()
        });
    });
});

//채팅카운트 증가
router.post("/chat/add_count", function(req, res){
    var id=req.body.id;
    var ref=db.collection('product').doc(id);
    ref.get().then(doc => {
        var count=doc.data().cnt_chat + 1;
        ref.update({ cnt_chat: count });
    });
});

//채팅카운트 감소
router.post("/chat/del_count", function(req, res){
    var id=req.body.id;
    var ref=db.collection('product').doc(id);
    ref.get().then(doc => {
        var count=doc.data().cnt_chat - 1;
        ref.update({ cnt_chat: count });
    });
});
```

[carrot]-[public]-[stylesheets]-[chat.css]

```
div.header {
   position: sticky;
   top: 0;
}

.chat_wrap .header{
    font-size: 14px;
    padding: 15px 0px;
    background: #F18C7E;
    color: white;
    text-align: center;
}

.chat_wrap #chat{
    padding-bottom: 80px;
    width: 100%;
    font-size: 12px;
}

.chat_wrap #chat .left{
    text-align: left;
}

.chat_wrap #chat .right{
    text-align: right;
}

.chat_wrap #chat .sender{
    margin: 10px 25px 0px 10px;
    font-weight: bold;
}

.chat_wrap #chat .message{
    display: inline-block;
    margin: 5px 20px 0px 10px;
    max-width: 75%;
    border: 1px solid gray;
    padding: 5px;
    border-radius: 5px;
    background-color: #FCFCFC;
    text-align: left;
}

.chat_wrap #chat .date {
    margin: 5px 20px 10px 10px;
    font-size: 10px;
}

.chat_wrap .input-div{
    position: fixed;
    bottom: 0px;
    width: 100%;
    background-color: #FFF;
    text-align: center;
    border-top: 1px solid #F18C7E;
}

#txtMessage {
    width: 100%;
    height: 80px;
    border: none;
    padding: 5px;
}
```

[carrot]-[views]-[products]-[chat.ejs]

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.18.0/moment.min.js"></script>
<link rel="stylesheet" href="/stylesheets/chats.css"/>
<style>
    .right sender {
        display: none;
    }
    .left .del {
        display: none;
    }
    img {
        width: 50px;
        border-radius: 5px;
    }
    .thumbnail {
        float: left;
        margin: 0px 20px;
    }
    .info {
        float: left;
    }
</style>
<div class="chat_wrap">
    <div class="header" style="overflow:hidden;">
        <span id="id" style="display:none;">
            <%=id%>
        </span>
        <div class="thumbnail">
            <img src="<%=data.url%>"/>
        </div>
        <div style="float:left;">
            <div>
                <%=data.name%>
                [ <%=data.price%>원 ]
            </div>
            <div><%=data.email%></div>
        </div>
    </div>
    <div id="chat"></div>
    <script id="temp" type="text/x-handlebars-template">
        {{#each .}}
        <div class="{{printLeftRight data.email}}">
            <div class="sender">{{data.email}}</div>
            <div class="message">
                {{data.message}}
                <a class="del" href="{{key}}"> X </a>
            </div>
            <div class="date">{{data.date}}</div>
        </div>
        {{/each}}
    </script>
    <script>
        Handlebars.registerHelper("printLeftRight", function(sender){
            var email=$("#email").html();
            if(email == sender){
                return "right";
            }else{
                return "left";
            }
        });
    </script>
    <div class="input-div"><textarea id="txtMessage" placeholder="Press Enter for send message."></textarea></div>
</div>
```

```
<script>
    var id=$("#id").html();
    var email=$("#email").html();
    realdb.ref("chats/" + id).on('value', (snap) => {  //채팅목록 출력
        var rows=[];
        snap.forEach(data=>{
            var row={"key":data.key, "data":data.val()};
            rows.push(row);
        });
        var temp=Handlebars.compile($("#temp").html());
        $("#chat").html(temp(rows));
        window.scrollTo(0, $("#chat").prop('scrollHeight'));
    });

    //채팅내용 입력 시 호출
    realdb.ref("chats/" + id).on('child_added', (snap) => {
        $("#chat").append(snap.val());
        window.scrollTo(0, $("#chat").prop('scrollHeight'));
    });

    //채팅내용 삭제 시 호출
    realdb.ref("/chats" + id).on('child_removed', (snap) => {
        $("#chat").remove(snap.val());
    });

    //채팅내용 입력
    $("#txtMessage").on('keydown', function(e){
        if(e.keyCode == 13 && !e.shiftKey) {
            e.preventDefault();
            var message = $(this).val();
            if(message=="") {
                alert("내용을 입력하세요!");
                return;
            }
            $("#txtMessage").val("");
            var strDate=moment(new Date()).format('YYYY-MM-DD HH:mm:ss')
            realdb.ref('chats/' + id).push().set({
                date: strDate,
                email: email,
                message: message
            });
            $.ajax({
                type: "post",
                url: "/products/chat/add_count",
                data: {"id": id},
                success: function(){}
            });
        }
    });

    //채팅내용 삭제 버튼을 클릭한 경우
    $("#chat").on("click", ".del", function(e) {
        e.preventDefault();
        var key=$(this).attr("href");
        if(!confirm(key + '내용을 삭제하실래요?')) return;
        realdb.ref("chats/" + id + "/" + key).remove();
        $.ajax({
            type: "post",
            url: "/products/chat/del_count",
            data: {"id": id},
            success: function(){}
        });
    });
</script>
```

[carrot]-[views]-[products]-[list.ejs]

```
    //채팅내용 입력 시 호출
    realdb.ref("chats/").on('child_changed', (snap) => {
        getList();
    });
```