**NodeJS express 모듈로 웹 프로젝트 구성하기**

1. mkdir Temp1

2. cd Temp1

3. express MyWeb --view=ejs

4. cd MyWeb

5. npm i

6. npm start

7. browser에서 http://localhost:3000 확인

**미션 1 : Temp1에 만든 MyWeb을 분석해서 http://localhost:3000/about 이 보이게~!**

**미션 2 : http://localhost:3000 화면이 부트스트랩 테마 2로 보이게~!**

**8. static 미들웨어 사용 :**

1. 미션2를 수행한 뒤 이 부트스트랩에서 필요로 하는 이미지들을 C:\Temp1\MyWeb\public\images 에 저장해 놓고
2. view/index.ejs에서 images/paris.jpg 와 같이 경로 수정
3. 브라우저로 이미지 보이게 하기

**9. 성능향상 팁 : static 미들웨어 설정을 logger 다음으로 승격 선언~!**

app.use(logger('dev'));

app.use(express.static(path.join(\_\_dirname, 'public')));

**10. 개발 편의성의 위해 nodemon 사용해보기**

1) npm i --save-dev nodemon

2) package.json 파일을 열어 다음과 같이 수정한다

"scripts": {

"start": "nodemon ./bin/www"

}

3) VSCode에서 Ctrl+` 으로 터미널을 열어 npm start 하여 서버를 다시 실행한다

4) 브라우저에서 about 을 요청한 뒤 about.ejs의 내용을 조금 바꾸고 저장하고, 서버 리스타트 없이 웹브라우저만 새로고침하면 변경된 내용이 반영되는 걸 볼 수 있다.

**11. html을 정적 폴더에 넣어보자**

1) public 폴더 밑에 html 폴더를 만들고 index.html을 간단히 만들어 놓는다

2) 웹브라우저로 [http://ip:port/html/index.html](about:blank) 했을 때 잘 보인다

**12. 인덱스 페이지를 / 요청만으로도 보이게 하자**

1) app.js에 다음 내용을 넣는다

app.set('view engine', 'ejs');

app.engine('html',require('ejs').renderFile);

2) view/index.ejs를 view/index.html이 되도록 파일 확장자 변경

3) routes/index.js를 다음과 같이 수정한다

router.get('/', function(req, res, next) {

//biz 코드...

res.render('index.html');

});

3) 웹브라우저로 [http://ip:port](about:blank) 했을 때 잘 보인다

**13. index.html에 my.js의 링크를 거는 것**

1) <script src="javascripts/my.js" ></script>

2) 서버쪽 MyWeb>public>javascripts>my.js 를 작성

14. my.js 를 다음과 같이 작성

|  |
| --- |
| $(document).ready(function () {  $('#member\_insert\_btn').click(function () {  const name=$('#name').val();  const email=$('#email').val();  const comments=$('#comments').val();  const send\_params={  name,  email,  comments  };    $.post("/member\_insert",send\_params,function (data,status) {  alert(data+":"+status);  });  });  }); |

15. 서버 app.js 에 라우팅 추가

app.use('/member\_insert', require('./routes/member\_insert'));

16. MyWeb>routes>member\_insert.js 를 작성

|  |
| --- |
| var express = require('express');  var router = express.Router();    /\* POST users listing. \*/  router.post('/', function(req, res, next) {  //biz  const result={msg:"가입되셨습니다"};  res.json(JSON.stringify(result));  });    module.exports = router; |

17. my.js에서 data를 자바스크립트 객체로 파싱해서 contact 구역에 msg값을 출력하기

|  |
| --- |
| $.post("/member\_insert",send\_params,function (data,status) {  //alert(data+":"+status);  const parsed\_data=JSON.parse(data);  $('#result\_div').html(`<h1>${parsed\_data.msg}</h1>`);  }); |

18. member\_insert.js에서 사용자 파라미터 파싱하기

|  |
| --- |
| var express = require('express');  var router = express.Router();  /\* POST users listing. \*/  router.post('/', function (req, res, next) {  // 일을 할 code를 넣기  // msg 쪽 ${req.body.name}으로 하는 부분이 사용자의 parameter를 가져오는 것  console.log('Session ID = ' + req.sessionID);  req.session.name = req.body.name;  req.session.email = req.body.email;  req.session.comments = req.body.comments;  const result = {  name : req.body.name,  email : req.body.email,  comments : req.body.comments,  msg : `${req.body.name}님이 가입되셨습니다. `  };  res.json(JSON.stringify(result));  });  module.exports = router; |

19. MyWeb>routes>login.js 를 작성

|  |
| --- |
| var express = require('express');  var router = express.Router();  router.post('/', function (req, res, next) {  // biz -> DB 연동 및 수행할 것들을 작성  const result = {  msg : ''  };  console.log('Session ID = ' + req.sessionID);  console.log(req.session.name);  console.log(req.session.email);  console.log(req.session.comments);  // member\_insert.js file에서 req.session 값을 지정하였기에 가능  // 해당 session 안에 담겨있는 email과 사용자가 입력한 email과 비교  if(req.session.email === req.body.email) {  result.msg = `${req.session.email}님이 로그인 되었습니다. `;  } else {  result.msg = '다시 로그인해주세요. ';  }  res.json(JSON.stringify(result));  });  module.exports = router; |

20. my.js에서 data를 자바스크립트 객체로 파싱해서 contact 구역에 msg값을 출력하기

|  |
| --- |
| $.post('/login', login\_params, (data, status) => {  const parsedLoginData = JSON.parse(data);  $('#login\_div').html(`<h1>${parsedLoginData.msg}</h1>`);  }); |

21. login.js에서 사용자 파라미터 파싱하기

|  |
| --- |
| if(req.session.email === req.body.email) {  result.msg = `${req.session.email}님이 로그인 되었습니다. `;  } else {  result.msg = '다시 로그인해주세요. ';  } |