|  |  |
| --- | --- |
| 스터디 날짜 | 22.05.18 |
| 제출일 | 22.06.29 |
| 학과 | 컴퓨터정보공학부 |
| 학번 | 21 |
| 이름 | 노지혜 |

2022 CAT-CERT Junior Web study 6

**목차**

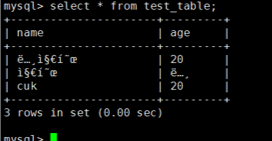
1. **실습**
2. **node.js 방명록 만들기**
3. **rest api 조사**
4. **실습**

-#1 정상 작동하게 만들기

소스코드

|  |
| --- |
| <?php  include('./dbinit.php');  $name=$\_POST['name'];  $age=$\_POST['age'];  $query="insert into test\_table (name,age) values ('$name','$age')";  if($result = mysqli\_query($conn,$query)){  while($row=mysqli\_fetch\_array($result)){  print\_r($row);  echo "<br>";  }  }  ?>  <html>  <head>  <title> ex1 </title>  </head>  <body>  <form action="" method="post">  <input type="text" name="name"/>  <input type="text" name="age"/>  <input type="submit"/>  </form>  </body>  </html> |

실행 결과



-#2 Old\_Member, New\_Member 모두 table에 출력

소스코드

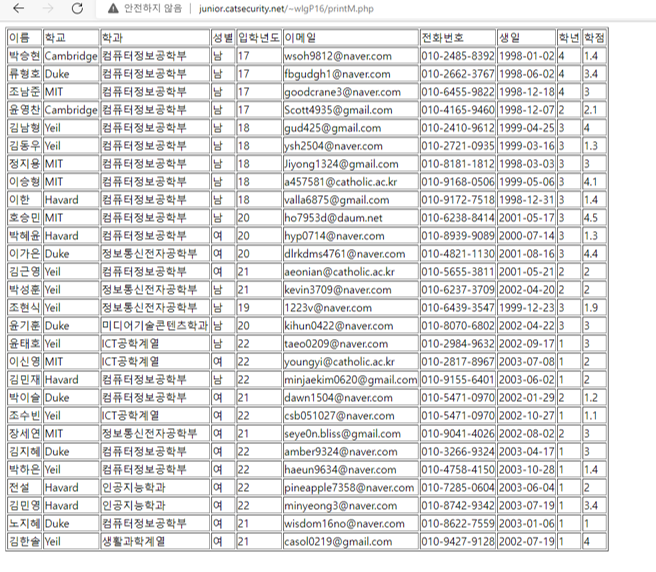
dbinit.php

|  |
| --- |
| <?php  $db\_Host="localhost";  $db\_Id="wlgP16";  $db\_Pass="wlgP0106!";  $db\_Name="wlgP16";  $conn=mysqli\_connect($db\_Host,$db\_Id,$db\_Pass,$db\_Name);  mysqli\_set\_charset($conn,"utf8");  if(mysqli\_connect\_errno())  die('Connect Error : '.mysqli\_connect\_errno());  ?> |

printM.php

|  |
| --- |
| <?php  include('./dbinit.php');  $query="select \* from old\_member union select \* from new\_member";  $result=mysqli\_query($conn,$query);    ?>  <html>  <head>  <title> printM </title>  </head>  <body>  <table border="1">  <tr> <td>이름</td>  <td>학교</td>  <td>학과</td>  <td>성별</td>  <td>입학년도</td>  <td>이메일</td>  <td>전화번호</td>  <td>생일</td>  <td>학년</td>  <td>학점</td>  </tr>  <?php  while($row=mysqli\_fetch\_array($result)){  echo "<tr>";  echo "<td>";  echo $row['Name'];  echo" </td>";  echo "<td>";  echo $row['School'];  echo "</td>";  echo "<td>";  echo $row['Major'];  echo "</td>";  echo "<td>";  echo $row['Gender'];  echo "</td>";  echo "<td>";  echo $row['EnterYear'];  echo "</td>";  echo "<td>";  echo $row['Email'];  echo "</td>";  echo "<td>";  echo $row['Phone'];  echo "</td>";  echo "<td>";  echo $row['Birth'];  echo "</td>";  echo "<td>";  echo $row['Grade'];  echo "</td>";  echo "<td>";  echo $row['Score'];  echo "</td>";  echo "</tr>";  }  ?>  </table>  </body>  </html> |

출력 결과



-#3 Node.js로 Old\_Member, New\_Member 모두 table에 출력

소스코드

db.js

|  |
| --- |
| const mysql = require("mysql");  const connection = mysql.createConnection({  host : 'localhost',  user : 'wlgp16',  password : '2022Wnsldj!',  port : 3306,  database :'wlgp16'  });  connection.connect(function(err){  if (err) console.log(err);  else console.log('Connected!');  });  module.exports=connection; |

App2.js

|  |
| --- |
| const express = require("express");  const ejs = require("ejs");  const path = require("path");  const app = express();  const conn=require("./db.js");  app.listen(8008,function(){  console.log("listening on port 8008");  });  app.use(express.urlencoded({extended: false}));  app.set("views",path.join(\_\_dirname, "views"));  app.set("view engine","ejs");  app.get("/",function(req,res){  var sql="SELECT \* FROM Old\_Member union SELECT \* FROM New\_Member";  conn.query(sql,function (err,rows){  if (err) console.log("query is not excuted. select fail!\n" + err);  else res.render("index.ejs",{list: rows});  });  }); |

Index2.ejs

|  |
| --- |
| <html>  <head>  <title>CAT-Security Node.js & MySQL</title>  <style>  table{  width: 100%;  text-align: center;  border-collapse: collapse;  border-top: 2px solid #ed0606;  }  th,  td{  border-bottom: 1px dashed #001763;  }  </style>  </head>  <body>  <h2>CAT-Security Node.js & MySQL</h2>  <div>  <table>  <thead>  <th> 이름 </th>  <th> 학교 </th>  <th> 전공 </th>  <th> 성별 </th>  <th> 입학년도 </th>  <th> 이메일 </th>  <th> 전화번호 </th>  <th> 생일 </th>  <th> 학년 </th>  </thead>  <tbody> <% for(var i of list) { %> <tr>  <td> <%= i.Name %></td>  <td> <%= i.School %></td>  <td> <%= i.Major %></td>  <td> <%= i.Gender %></td>  <td> <%= i.EnterYear %></td>  <td> <%= i.Email %></td>  <td> <%= i.Phone %></td>  <td> <%= i.Birth %></td>  <td> <%= i.Grade %></td>  </tr> <% } %> </tbody>  </table>  </div>  </body>  </html> |

출력 결과



1. **node.js 방명록 만들기**

소스코드

db.js /home/wlgp16/visiter/views

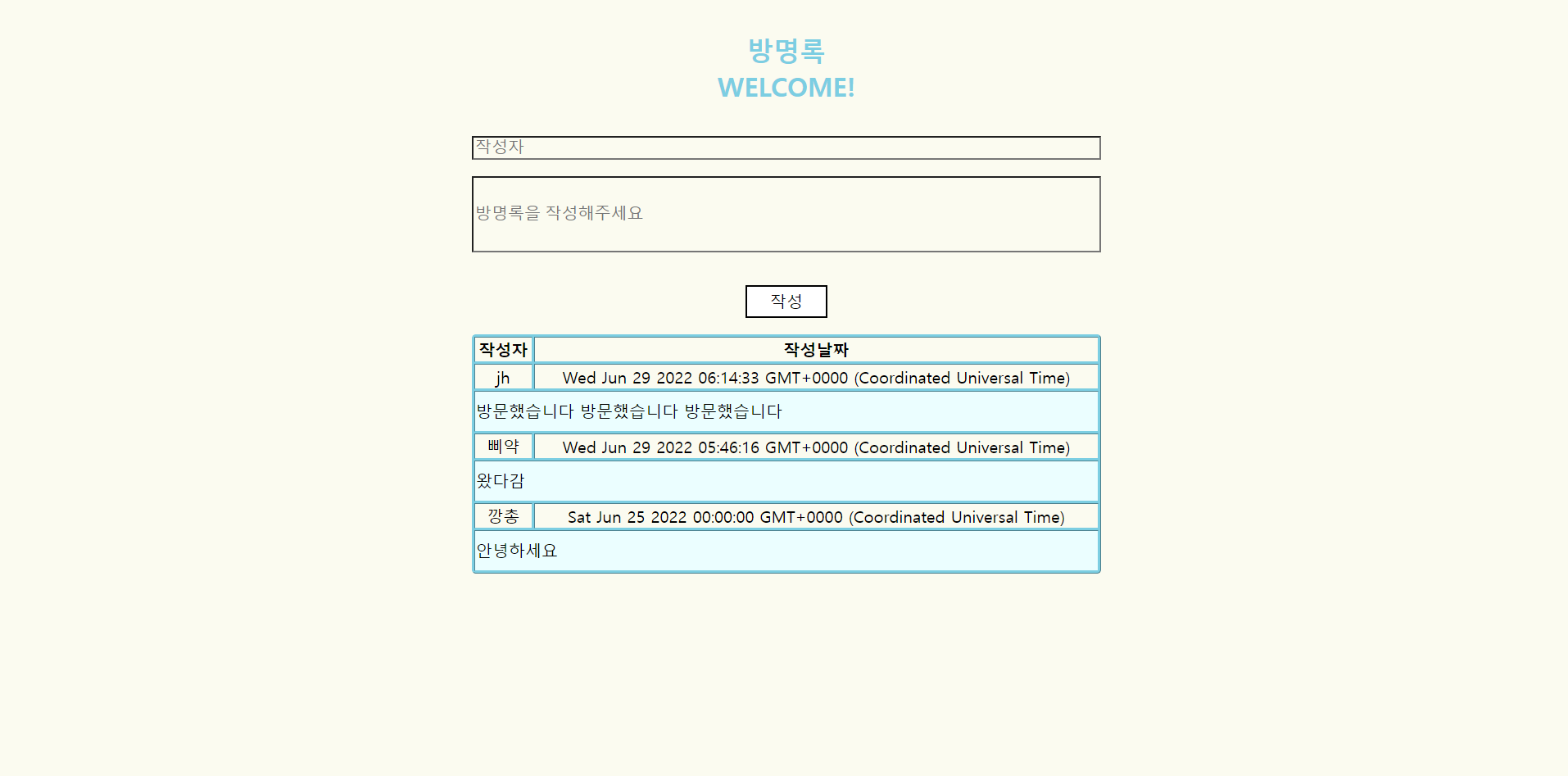
|  |
| --- |
| const mysql = require("mysql");  const connection = mysql.createConnection({  host : 'localhost',  user : 'wlgp16',  password : '2022Wnsldj!',  port : 3306,  database :'wlgp16'  });  connection.connect(function(err){  if (err) console.log(err);  else console.log('Connected!');  });  module.exports=connection; |

app.js /home/wlgp16/visiter/views

|  |
| --- |
| const express = require("express");  const ejs = require("ejs");  const app = express();  const path = require("path");  const conn=require("./db.js");  app.use(express.urlencoded({extended: true}));  app.set("view engine","ejs");  app.get('/',function(req,res){  const sql = "select \* from visit\_table order by seq desc";  conn.query(sql,function(err,result){  if (err) console.log("query is not excuted. select fail!\n"+err);  res.render('index.ejs',{list: result});  });  });  app.post ('/',function(req,res){  const sql = "insert into visit\_table set ?";  conn.query(sql,req.body,function(err,result){  if (err) console.log("error\n"+err);  else {console.log(result); res.redirect('/');}  });  });  app.listen(8008,function(){  console.log("listening on port 8008");  }); |

index.ejs : /home/wlgp16/visiter/views

|  |
| --- |
| <html>  <head>  <meta charset='UTF-8'>  <title>방명록</title>  <style>  \*{  background-color : #FBFBEF;  margin:auto;  margin-top:20px;  }  h1{  color: #5CD1E5;  }  table{  font-size:20px;  border-color: #5CD1E5;  background-color: #5CD1E5;  width:40%;  border-radius: 5px;  }  .in input{  width:40%;  border;pink;  font-size:20px;  }  span input{  height:10%;  font-size:20px;  }  button{  height:40px;  width:100px;  font-size:20px;  background-color:white;  }  </style>  </head>  <body>  <h1 align="center">방명록<br>WELCOME!</h1>  <form method="POST"><div align="center">  <div class="in"><input type="text" name="Name" placeholder="작성자"><br><span><input type="text" name="Story" placeholder="방명록을 작성해주세요"/></span><div>  <button type="submit"/>작성</button>  <table border="1">  <tr text-align="center">  <th> 작성자</th>  <th> 작성날짜</th>  </tr>  <% for(var i of list){%>  <tr>  <td align="center"><%= i.Name %></td>  <td align="center"><%= i.Upload %></td></tr>  <p><tr height="50">  <td colspan="2" style="background-color:#E6FFFF"><%= i.Story %></td>  </tr><p>  <%}%>  </table>  </body>  </html> |



1. **rest api 조사**

REST API : REST의 원리를 따르는 API

REST (Representational State Transfer) : HTTP 통신에서 어떤 자원에 대한 CRUD 요청을 Resource와 Method로 표현하여 특정한 형태로 전달하는 방식

API (Application Programming Interface) : 응용 프로그램에서 사용할 수 있도록 운영 체제나 프로그래밍 언어가 제공하는 기능을 제어할 수 있게 만든 인터페이스

구성 요소

Resource : 서버는 고유한 ID를 가지는 Resource를 가지고 있으며, 클라이언트는 Resource에 요청을 보낸다. 이러한 Resource는 URI에 해당한다.

Method : 서버에 요청을 보내기 위한 방식으로 GET, POST, PUT, PATCH, DELETE가 있다. CRUD 연산 중에서 처리를 위한 연산에 맞는 Method를 사용하여 서버에 요청을 보내야 한다.

\*CRUD :  대부분의 컴퓨터 소프트웨어가 가지는 기본적인 데이터 처리 기능인 Create(생성)-POST, Read(읽기)-GET, Update(갱신)-PUT, Delete(삭제)-DELETE를 묶어서 일컫는 말

REST의 조건

1. Uniform Interface (일관된 인터페이스) : 요청이 어디에서 오는지와 무관하게, 동일한 리소스에 대한 모든 API 요청은 동일하게 보여야 한다. 클라이언트가 필요로 하는 모든 정보를 포함해야 한다.
2. Stateless (무상태성) : 서버는 각각의 요청을 별개의 것으로 인식하고 처리해야하며, 이전 요청이 다음 요청에 연관되어서는 안된다. REST API는 세션정보나 쿠키 정보를 활용하여 작업을 위한 상태정보를 저장 및 관리하지 않는다.
3. Cacheable (캐시 가능) : 리소스를 클라이언트 또는 서버측에서 캐싱할 수 있어야한다. 서버 응답에는 전달된 리소스에 대해 캐싱이 허용되는지 여부에 대한 정보도 포함되어야 한다. 서버측의 확장성 증가와 클라이언트 측의 성능 향상을 동시에 얻는 것이다.
4. Client-Server Architecture (서버-클라이언트 구조) : 자원을 가지고 있는 쪽이 서버, 자원을 요청하는 쪽이 클라이언트에 해당한다. 서버는 API를 제공하며, 클라이언트는 사용자 인증, 로그인 정보등을 직접 관리하는 등 역할을 확실히 구분시킴으로써 서로 간의 의존성을 줄인다.
5. Self-Descriptiveness (자체 표현) : 요청 메시지만 보고도 쉽게 이해할 수 있는 자체 표현 구조로 되어 있다. Ex) JSON 형태의 REST 메시지
6. 계층 구조 : 서버는 다중 계층으로 구성될 수 있으며 보안, 암호화 등을 위한 계층을 추가하여 구조를 변경할 수 있다. 그러나 클라이언트는 서버와 직접 통신하는지, 중간 매체와 통신하는지 알 수 없다.

REST API 설계 규칙

1. URI는 동사보다 명사를, 대문자보다는 소문자를 사용한다.
2. 슬래시는 계층 관계를 나타내는데 사용되며 URI마지막은 슬래시를 포함하지 않는다.
3. 언더바 대신 하이픈을 사용한다.
4. 파일확장자는 URI에 포함하지 않는다.
5. URI 자원의 행위에 대한 표현이 들어가지 않는 대신 HTTP메소드를 통해 대신한다.