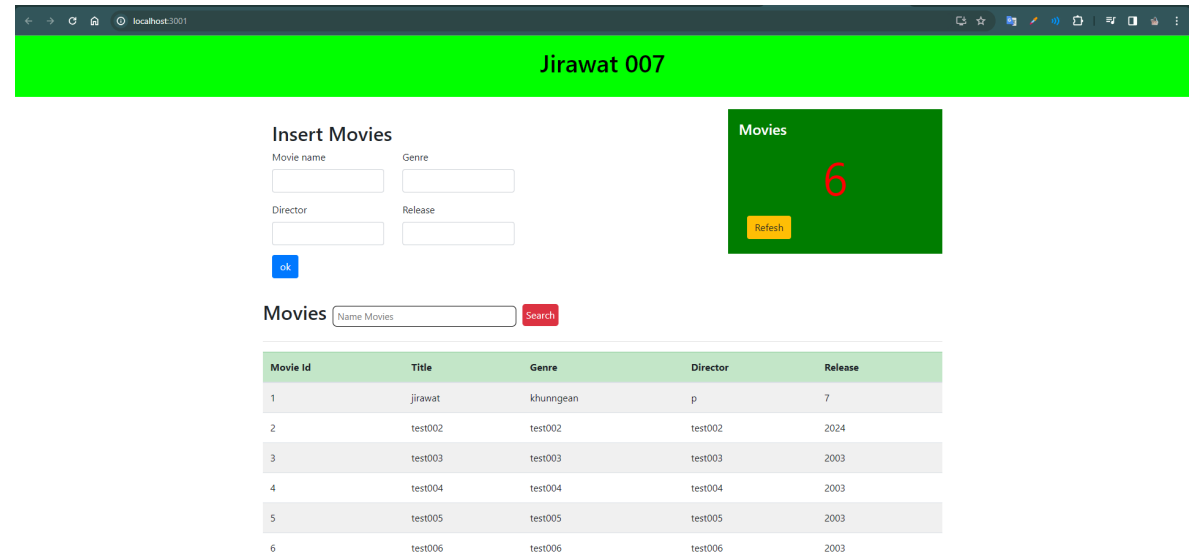
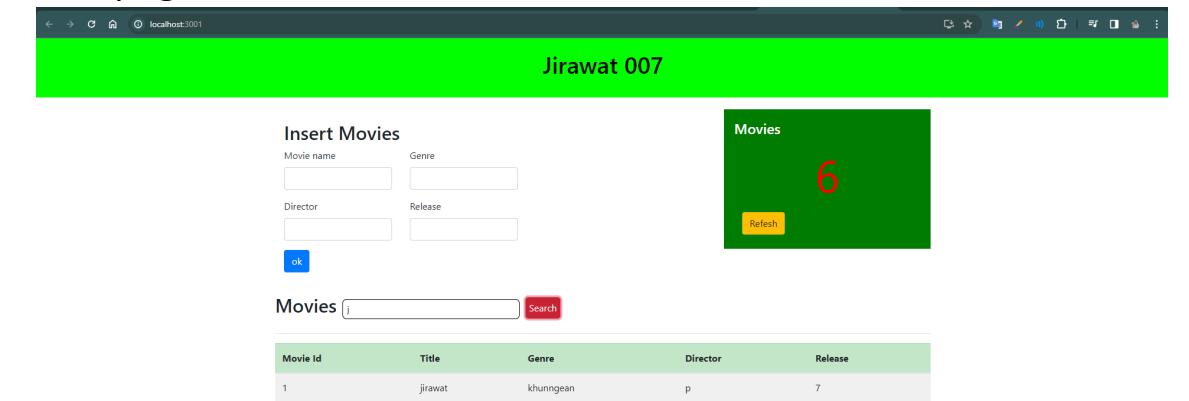


ENGSE203-2 Mid-term

Home page



Home page show some name



ap/index.js

```
1  const hapi = require('@hapi/hapi');
2  const env = require('./env.js');
3  const Movies = require('./respository/movie');
4
5  const express = require('express');
6  const app = express();
7
8  const path = require('path');
9      bodyParser = require("body-parser");
10
11  //-----
12  const api_port = 4000;
13  const web_port = 4001;
```

เฟรมเวิร์กและพอร์ต

```
1  //----- hapi -----
2  console.log('Running Environment: ' + env);
3
4  const init = async () => {
5      const server = hapi.Server({
6          port: api_port,
7          host: '0.0.0.0',
8          routes: {
9              cors: true
10          }
11      });
12
13  //-----
```

```
1 await server.register(require('@hapi/inert'));
2 server.route({
3   method: "GET",
4   path: "/",
5   handler: () => {
6     return '<h3> Welcome P</h3>';
7   }
8 });
9
10
11 //API: http://localhost:3001/api/movie/all
12 server.route({
13   method: 'GET',
14   path: '/api/movie/all',
15   config: {
16     cors: {
17       origin: ['*'],
18       additionalHeaders: ['cache-control', 'x-requested-width']
19     }
20   },
21   handler: async function (request, reply) {
22     //var param = request.query;
23     //const category_code = param.category_code;
24     try {
25       const responsedata = await Movies.MovieRepo.getMovieList();
26       if (responsedata.error) {
27         return responsedata.errMessage;
28       } else {
29         return responsedata;
30       }
31     } catch (err) {
32       server.log(["error", "home"], err);
33       return err;
34     }
35   }
36 });
37 });
```

```

1  server.route({
2      method: 'GET',
3      path: '/api/movie/search',
4      config: {
5          cors: {
6              origin: ['*'],
7              additionalHeaders: ['cache-control', 'x-requested-width']
8          }
9      },
10     handler: async function (request, reply) {
11         var param = request.query;
12         const search_text = param.search_text;
13         //const title = param.title;
14         try {
15             const respondedata = await Movies.MovieRepo.getMovieSearch(search_text);
16             if (respondedata.error) {
17                 return respondedata.errMessage;
18             } else {
19                 return respondedata;
20             }
21         } catch (err) {
22             server.log(["error", "home"], err);
23             return err;
24         }
25     }
26 });
27
28
29
30 server.route({
31     method: 'POST',
32     path: '/api/movie/insert',
33     config: {
34         payload: {
35             multipart: true,
36         },
37         cors: {
38             origin: ['*'],
39             additionalHeaders: ['cache-control', 'x-requested-width']
40         }
41     },
42     handler: async function (request, reply) {
43         const {
44             title,
45             genre,
46             director,
47             release_year
48         } = request.payload;
49         //const title = request.payload.title;
50         //const genre = request.payload.genre;
51         try {
52             const respondedata = await Movies.MovieRepo.postMovie(title, genre, director, release_year);
53             if (respondedata.error) {
54                 return respondedata.errMessage;
55             } else {
56                 return respondedata;
57             }
58         } catch (err) {
59             server.log(["error", "home"], err);
60             return err;
61         }
62     }
63 });
64
65 await server.start();
66 console.log('API Server running on %s', server.info.uri);
67 //-----
68 };

```



```
1 process.on('unhandledRejection', (err) => {  
2   console.log(err);  
3   process.exit(1);  
4 });  
5  
6 init();
```

api/respository/movie.js



```
1 var mysql = require('mysql');  
2 const env = require('../env.js');  
3 const config = require('../dbconfig.js')[env];
```



```
1  async function getMovieList() {
2      var Query;
3      var pool = mysql.createPool(config);
4      return new Promise((resolve, reject) => {
5          Query = `SELECT * FROM movies`;
6          pool.query(Query, function (error, results, fields) {
7              if (error) throw error;
8
9              if (results.length > 0) {
10                 pool.end();
11                 return resolve(results);
12             } else {
13                 pool.end();
14                 return resolve({
15                     statusCode: 404,
16                     returnCode: 11,
17                     message: 'No movie found',
18                 });
19             }
20         });
21     });
22 }
```



```
1  async function getMovieSearch(search_text) {
2      var Query;
3      var pool = mysql.createPool(config);
4      return new Promise((resolve, reject) => {
5          Query = `SELECT * FROM movies WHERE title LIKE '%${search_text}%'`;
6          pool.query(Query, function (error, results, fields) {
7              if (error) throw error;
8              if (results.length > 0) {
9                  pool.end();
10                 return resolve({
11                     statusCode: 200,
12                     returnCode: 1,
13                     data: results,
14                 });
15             } else {
16                 pool.end();
17                 return resolve({
18                     statusCode: 404,
19                     returnCode: 11,
20                     message: 'No movie found',
21                 });
22             }
23         });
24     });
25 }
```

```

1  async function postMovie(p_title,p_genre,p_director,p_release_year) {
2
3      var Query;
4      var pool = mysql.createPool(config);
5
6      return new Promise((resolve, reject) => {
7
8          var post = {
9              title: p_title,
10             genre: p_genre,
11             director: p_director,
12             release_year: p_release_year
13         };
14
15         console.log('post is: ', post);
16
17
18         Query = 'INSERT INTO movies SET ?';
19         pool.query(Query, post, function (error, results, fields) {
20             if (error){
21                 console.log('error_code_msg', error.code+':'+error.sqlMessage);
22                 pool.end();
23                 return resolve({
24                     statusCode: 450,
25                     returnCode: 9,
26                     message: error.code+':'+error.sqlMessage
27                 });
28             }else{
29                 console.log('results: ', results);
30                 if (results.affectedRows > 0) {
31                     pool.end();
32                     return resolve({
33                         statusCode: 200,
34                         returnCode: 1,
35                         message: 'Movie list was inserted',
36                     });
37                 }
38             }
39
40             });
41
42         });
43
44     }
45
46     module.exports.MovieRepo = {
47         getMovieList: getMovieList,
48         getMovieSearch: getMovieSearch,
49         postMovie: postMovie,
50
51     };

```


my-app/src/components/CreateMovie.js

เป็นหน้าการเพิ่มข้อมูลว่าเพิ่มอะไรไปบ้าง

```
1 import React from 'react'
2
3
4 const createMovie = ({onChangeForm, createMovie }) => {
5
6
7   return(
8     <div className="container">
9       <div className="row">
10         <div className="col-md-7 mrgnbtm">
11           <h2>Insert Movies</h2>
12
13           <form>
14
15             <div className="row">
16               <div className="form-group col-md-6">
17                 <label>Movie name</label>
18                 <input type="text" onChange={e => onChangeForm(e)} className="form-control" name="title" id="title"/>
19               </div>
20
21               <div className="form-group col-md-6">
22                 <label htmlFor="exampleInputPassword1">Genre</label>
23                 <input type="text" onChange={e => onChangeForm(e)} className="form-control" name="genre" id="genre"/>
24               </div>
25
26               <div className="form-group col-md-6">
27                 <label>Director</label>
28                 <input type="text" onChange={e => onChangeForm(e)} className="form-control" name="director" id="director"/>
29               </div>
30
31               <div className="form-group col-md-6">
32                 <label>Release</label>
33                 <input type="text" onChange={e => onChangeForm(e)} className="form-control" name="release_year" id="release_year"/>
34               </div>
35             </div>
36
37             <button type="button" onClick={e => createMovie()} className="btn btn-primary mb-3">ok</button>
38           </form>
39         </div>
40       </div>
41     </div>
42   )
43 }
44
45 export default createMovie
```

my-app/src/components/DisplayBoard.js

```
1 import React from 'react'
2
3 export const DisplayBoard = ({numberOfMovies, getAllMovies}) => {
4
5   const headerStyle = {
6
7     width: '100%',
8     padding: '2%',
9     backgroundColor: "red",
10    color: 'white',
11    textAlign: 'center'
12  }
13
14  return(
15    <div style={{backgroundColor:'green'}} className="display-board">
16      <h4 style={{color: 'white'}}>Movies</h4>
17      <div className="number">
18        {numberOfMovies}
19      </div>
20      <div className="btn">
21        <button type="button" onClick={e => getAllMovies()} className="btn btn-warning">Refresh</button>
22      </div>
23    </div>
24  )
25 }
26
```

my-app/src/components/Header.js

```
1 import React from 'react'
2
3 export const Header = () => {
4
5     const headerStyle = {
6
7         width: '100%',
8         padding: '20px',
9         backgroundColor: "lime",
10        color: 'black',
11        textAlign: 'center',
12        height: '100px'
13    }
14
15    return(
16        <div style={headerStyle}>
17            <h1>Jirawat 007</h1>
18        </div>
19    )
20 }
21 }
```

my-app/src/components/Movie.js

```
1 import React, { useState } from 'react';
2
3 export const Movie = ({ movies }) => {
4     const [searchData, setSearchData] = useState('');
5     const [searching, setSearching] = useState(false);
6     const [movieSearch, setMovieSearch] = useState('');
7     const [movie, setMovie] = useState('');
8
9     const handleSubmit = (e) => {
10         e.preventDefault();
11         if (searchData.trim() !== '') {
12             setSearching(true);
13             setMovieSearch(searchData);
14             setMovie('');
15             setSearching(true);
16         }
17     };
18
19     const handleInputChange = (e) => {
20         setSearchData(e.target.value);
21     };
22
23     const handleMovieSearch = (e) => {
24         setMovieSearch(e.target.value);
25     };
26
27     const handleMovieChange = (e) => {
28         setMovie(e.target.value);
29     };
30
31     const handleKeyPress = (e) => {
32         if (e.key === 'Enter') {
33             handleSubmit(e);
34         }
35     };
36
37     const handleKeyPressMovie = (e) => {
38         if (e.key === 'Enter') {
39             setMovieSearch(movieSearch + e.target.value);
40         }
41     };
42
43     return (
44         <div>
45             <div>
46                 <input type="text" value={searchData} onChange={handleInputChange} />
47                 <button type="button" value="Search" onClick={handleSubmit} />
48             </div>
49             <div>
50                 <input type="text" value={movieSearch} onChange={handleMovieSearch} />
51                 <button type="button" value="Search" onClick={handleSubmit} />
52             </div>
53             <div>
54                 <input type="text" value={movie} onChange={handleMovieChange} />
55                 <button type="button" value="Search" onClick={handleSubmit} />
56             </div>
57             <div>
58                 <div>
59                     <div>
60                         <div>
61                             <div>
62                                 <div>
63                                     <div>
64                                         <div>
65                                             <div>
66                                                 <div>
67                                                     <div>
68                                                         <div>
69                                                             <div>
70                                                                 <div>
71                                                                     <div>
72                                                                         <div>
73                                                                             <div>
74                                                                                 <div>
75                                                                                     <div>
76                                                                                         <div>
77                             </div>
78                         </div>
79                     </div>
80                 </div>
81             </div>
82         </div>
83     );
84 }
```

my-app/src/components/Movies.js

```
1 import React, { useState } from "react";
2
3 export const Movies = ({ movies }) => {
4   const [searchText, setSearchText] = useState('');
5   const [searchResults, setSearchResults] = useState([]);
6   const [isSearching, setIsSearching] = useState(false);
7   const handleSearch = async () => {
8     try {
9       const response = await fetch('http://localhost:3001/api/movie/search?search_text=${searchText}');
10      const data = await response.json();
11      if (data.returnCode === 1) {
12        setSearchResults(data.data);
13        setIsSearching(true);
14      } else {
15        setSearchResults([]);
16        setIsSearching(false);
17      }
18    } catch (error) {
19      console.error('Error searching for movies:', error);
20    }
21  };
22}
```

```
1 if (movies.length === 0) return null;
2
3 const MovieRow = (movie, index) => {
4   return (
5     <tr key={index} className={index % 2 === 0 ? "odd" : "even"}>
6       <td>{index + 1}</td>
7       <td>{movie.title}</td>
8       <td>{movie.genre}</td>
9       <td>{movie.director}</td>
10      <td>{movie.release_year}</td>
11    </tr>
12  );
13 };
14
15 const movieTable = isSearching ?
16   searchResults.map((movie, index) => MovieRow(movie, index)) :
17   movies.map((movie, index) => MovieRow(movie, index));
```

```
1 export default function App() {
2   const [searchText, setSearchText] = useState('');
3   const [searchResults, setSearchResults] = useState([]);
4   const [isSearching, setIsSearching] = useState(false);
5   const handleSearch = async () => {
6     try {
7       const response = await fetch('http://localhost:3001/api/movie/search?search_text=${searchText}');
8       const data = await response.json();
9       if (data.returnCode === 1) {
10        setSearchResults(data.data);
11        setIsSearching(true);
12      } else {
13        setSearchResults([]);
14        setIsSearching(false);
15      }
16    } catch (error) {
17      console.error('Error searching for movies:', error);
18    }
19  };
20
21   const movieTable = isSearching ?
22     searchResults.map((movie, index) => MovieRow(movie, index)) :
23     movies.map((movie, index) => MovieRow(movie, index));
24
25   return (
26     <div>
27       <input type="text" value={searchText} onChange={e => setSearchText(e.target.value)} />
28       <button onClick={handleSearch}>Search</button>
29     </div>
30     <div>
31       {movieTable}
32     </div>
33   );
34 }
```

my-app/src/services/MovieService.js

Change this ({movie:data}) -> body: JSON.stringify({...data}) || PORT 3001 -> 4000 -> 4000 is PORT backend

```
1 export async function getAllMovies() {
2   try{
3     //const response = await fetch('/api/users');
4     const response = await fetch('http://localhost:4000/api/movie/all');
5     //const response = await fetch('/api/movie/all');
6     return await response.json();
7   }catch(error) {
8     return [];
9   }
10
11 }
12
13
14 export async function createMovie(data) {
15   const response = await fetch('http://localhost:4000/api/movie/insert', {
16     method: 'POST',
17     headers: {'Content-Type': 'application/json'},
18     // body: JSON.stringify({movie:data})
19     body: JSON.stringify({...data})
20   })
21   return await response.json();
22   console.log(createMovie);
23 }
```

my-app/App.js

```
1 import React, { useState, useEffect } from 'react';
2 import 'bootstrap/dist/css/bootstrap.min.css';
3 import './App.css';
4 import { Header } from './components/Header'
5 import { Users } from './components/Users'
6 import { DisplayBoard } from './components/DisplayBoard'
7 import CreateUser from './components/CreateUser'
8 import { getAllUsers, createUser } from './services/UserService'
9 //-----
10 import { Movies } from './components/Movies'
11 import CreateMovie from './components/CreateMovie'
12 import { getAllMovies, createMovie } from './services/MovieService'
```



```
1 function App() {
2
3   const [user, setUser] = useState({})
4   const [users, setUsers] = useState([])
5   const [numberOfUsers, setNumberOfUsers] = useState(0)
6   //-----
7   const [movie, setMovie] = useState({})
8   const [movies, setMovies] = useState([])
9   const [numberOfMovies, setNumberOfMovies] = useState(0)
10
11   const userCreate = (e) => {
12
13     createUser(user)
14     .then(response => {
15       console.log(response);
16       setNumberOfUsers(numberOfUsers+1)
17     });
18   }
19
20   const movieCreate = (e) => {
21     createMovie(movie)
22     .then(response => {
23       console.log(response);
24       setNumberOfMovies(numberOfMovies+1)
25     });
26   }
```



```
1  const fetchAllMovies = () => {
2    getAllMovies()
3    .then(movies => {
4      console.log(movies)
5      setMovies(movies);
6      setNumberOfMovies(movies.length)
7    });
8  }
9
10  useEffect(() => {
11    getAllMovies()
12    .then(movies => {
13      console.log(movies)
14      setMovies(movies);
15      setNumberOfMovies(movies.length)
16    });
17  }, [])
18
19  const onChangeForm = (e) => {
20    if (e.target.name === 'title') {
21      movie.title = e.target.value;
22    } else if (e.target.name === 'genre') {
23      movie.genre = e.target.value;
24    } else if (e.target.name === 'director') {
25      movie.director = e.target.value;
26    } else if (e.target.name === 'release_year') {
27      movie.release_year = e.target.value;
28    }
29    setMovie(movie)
30  }
```

```

1  return (
2      <div className="App">
3          <Header></Header>
4          <div className="container mrgnbtm">
5              <div className="row">
6                  <div className="col-md-8">
7                      <CreateMovie
8                          user={user}
9                          onChangeForm={onChangeForm}
10                         createMovie={movieCreate}
11                     >
12                         </CreateMovie>
13                     </div>
14                     <div className="col-md-4">
15                         <DisplayBoard
16                             numberOfMovies={numberOfMovies}
17                             getAllMovies={fetchAllMovies}
18                         >
19                             </DisplayBoard>
20                         </div>
21                     </div>
22                     <div className="row mrgnbtm">
23                         <Movies movies={movies}></Movies>
24                     </div>
25                 </div>
26             </div>
27         );
28     }
29
30     export default App;

```

my-app ก่อน npm start

[\\$env:NODE_OPTIONS = "--openssl-legacy-provider"](#) (windows)

my-app/package.json

```
1  {
2    "name": "my-app",
3    "version": "0.1.0",
4    "private": true,
5    "dependencies": {
6      "@testing-library/jest-dom": "^4.2.4",
7      "@testing-library/react": "^9.5.0",
8      "@testing-library/user-event": "^7.2.1",
9      "bootstrap": "^4.5.0",
10     "react": "^16.13.1",
11     "react-bootstrap": "^1.0.1",
12     "react-dom": "^16.13.1",
13     "react-scripts": "3.4.1"
14   },
15   "scripts": {
16     "start": "set PORT=3001 && react-scripts start",
17     "build": "react-scripts build",
18     "test": "react-scripts test",
19     "eject": "react-scripts eject"
20   },
21   "proxy": "http://localhost:4000",
22   "eslintConfig": {
23     "extends": "react-app"
24   },
25   "browserslist": {
26     "production": [
27       ">0.2%",
28       "not dead",
29       "not op_mini all"
30     ],
31     "development": [
32       "last 1 chrome version",
33       "last 1 firefox version",
34       "last 1 safari version"
35     ]
36   }
37 }
38
```

["start": "set PORT=3001 && react-scripts start" \(windows\)](#)

DATABASES

movies

title | genre | director | releases_year
















✓ Showing rows 0 - 6 (7 total, Query took 0.0011 seconds.)

```
SELECT * FROM `movies`
```

☐ Profiling [[Edit inline](#)] [[Edit](#)] [[Explain SQL](#)] [[Create PHP code](#)] [[Refresh](#)]

☐ Show all | Number of rows: 25 ▼ Filter rows:

Extra options

<div>← T →</div>				title	genre	director	release_year
<input type="checkbox"/>	 Edit	 Copy	 Delete	test002	test002	test002	2024
<input type="checkbox"/>	 Edit	 Copy	 Delete	test003	test003	test003	2003
<input type="checkbox"/>	 Edit	 Copy	 Delete	test004	test004	test004	2003
<input type="checkbox"/>	 Edit	 Copy	 Delete	test005	test005	test005	2003
<input type="checkbox"/>	 Edit	 Copy	 Delete	test006	test006	test006	2003

จิรวุฒน์ ขุนเงิน 6654321007-1
ศุภวิชญ์ กะตะศิลา 66543210032-9