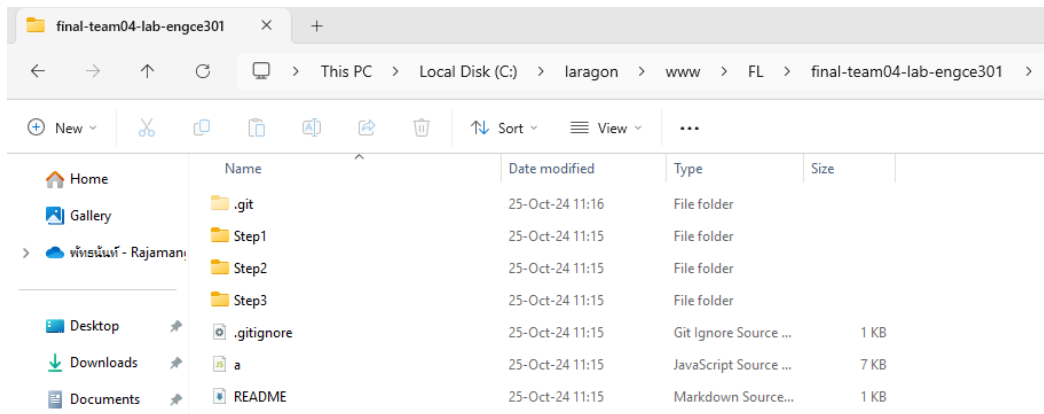


Term-Project (1-67) for ENGCE301 and ENGCE112 (งานกลุ่ม 4)



Step1

เพื่อให้ใช้คำสั่ง npm ของโปรแกรมได้ ** คำสั่งนี้จะใช้ในทุก step

On PowerShell:

```
$env:NODE_OPTIONS = "--openssl-legacy-provider"
```

เข้าไปยัง step1/client เพื่อติดตั้ง node_modules ด้วยคำสั่ง npm i

```
PS C:\laragon\www\FL\final-team04-lab-engce301\Step1\client> npm i
up to date, audited 2010 packages in 10s

207 packages are looking for funding
  run `npm fund` for details

146 vulnerabilities (4 low, 105 moderate, 30 high, 7 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
PS C:\laragon\www\FL\final-team04-lab-engce301\Step1\client> |
```

เข้าไปยัง step1/server เพื่อติดตั้ง node_modules ด้วยคำสั่ง npm i

```
PS C:\laragon\www\FL\final-team04-lab-engce301\Step1\server> npm i
up to date, audited 116 packages in 1s

17 packages are looking for funding
  run `npm fund` for details

6 vulnerabilities (1 low, 2 moderate, 3 high)

To address all issues, run:
  npm audit fix

Run `npm audit` for details.
PS C:\laragon\www\FL\final-team04-lab-engce301\Step1\server> |
```

RUN Client และ server

Client จะใช้คำสั่ง npm start เพื่อเปิดใช้งานหน้าเว็บ โดยจะทำงานที่พอร์ต 3000

```
PS C:\laragon\www\FL\final-team04-lab-engce301\Step1\client> npm start

> client@1.0.0 start
> react-scripts start

i [wds]: Project is running at http://192.168.110.1/
i [wds]: webpack output is served from
i [wds]: Content not from webpack is served from C:\laragon\www\FL\final-team04-lab-engce301\Step1\client\public
i [wds]: 404s will fallback to /
Starting the development server...

Browserslist: caniuse-lite is outdated. Please run:
  npx update-browserslist-db@latest
  Why you should do it regularly: https://github.com/browserslist/update-db#readme
Compiled successfully!

You can now view client in the browser.

Local:      http://localhost:3000
On Your Network: http://192.168.110.1:3000

Note that the development build is not optimized.
To create a production build, use npm run build.
```

Server จะใช้คำสั่ง npm run server

เพื่อเปิดใช้งาน server แบบ nodemon ทำงานที่พอร์ต 8081 เว็บจะมีการรีโหลดตัวเองทุกครั้งที่มีการแก้ไขไฟล์

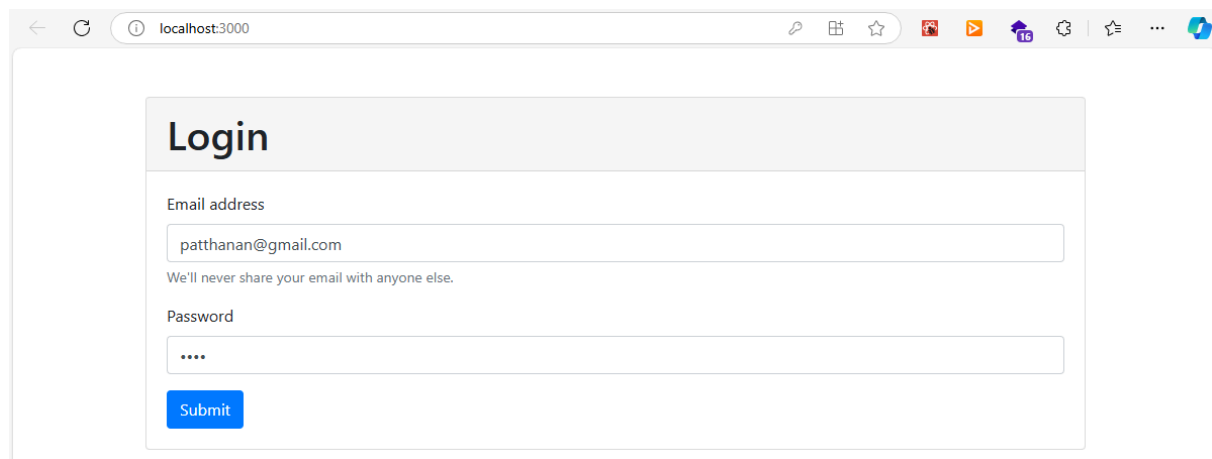
```
PS C:\laragon\www\FL\final-team04-lab-engce301\Step1\server> npm run server

> node-react@1.0.0 server
> nodemon

[nodemon] 3.1.4
[nodemon] to restart at any time, enter 'rs'
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting 'node app.js'
Server running on port: 8081
```

Result

จะมีหน้าล็อกอินขึ้นมาทำการใส่ email และ password เพื่อเข้าใช้งาน



The screenshot shows a web browser window with the address bar displaying 'localhost:3000'. The main content area features a login form with a light gray header containing the word 'Login'. Below the header, there is an 'Email address' label followed by a text input field containing 'patthanana@gmail.com'. Underneath the email field is a small line of text: 'We'll never share your email with anyone else.' Below this is a 'Password' label followed by a password input field with masked characters '....'. At the bottom of the form is a blue 'Submit' button.

สามารถใส่ keyword เพื่อค้นหารูปภาพได้


[←](#) [↻](#) [localhost:3000](#) [🔍](#) [🏠](#) [☆](#) [🔖](#) [🔧](#) [👤](#) [⚙️](#) [⋮](#) [🌐](#)

Log out


Welcome!

Enter one or multiple keywords below to search for artworks in the Art Institute of Chicago.


[Search artworks](#)





Knot
Japan, 2007
Honma Hideaki
Japanese, born 1959
Bamboo and rattan



Double Wind
Japan, 2005
Honma Hideaki
Japanese, born 1959
Madake and nemaqari bamboo and rattan



Corral, Hondo Valley
United States, 1972
William Clift
American, born 1944
Gelatin silver print, from the portfolio "New Mexico" (1993)



Step2

Cd เข้าไปยัง step 2

```
PS C:\laragon\www\FL\final-team04-lab-engce301\Step2> dir

Directory: C:\laragon\www\FL\final-team04-lab-engce301\Step2

Mode                LastWriteTime         Length Name
----                -
d-----          25-Oct-24      11:26             node_modules
d-----          25-Oct-24      11:15             public
d-----          25-Oct-24     13:44             server
d-----          25-Oct-24      11:15             src
-a-----          25-Oct-24      11:15              90 .babelrc
-a-----          25-Oct-24      11:15             333 .gitignore
-a-----          25-Oct-24      11:26          363952 package-lock.json
-a-----          25-Oct-24     13:48           920 package.json
-a-----          25-Oct-24      11:15           805 README.md
-a-----          25-Oct-24      11:15           533 webpack.config.js
```

ติดตั้ง node_modules ด้วยคำสั่ง npm i

```
PS C:\laragon\www\FL\final-team04-lab-engce301\Step2> npm i

up to date, audited 830 packages in 5s

52 packages are looking for funding
  run `npm fund` for details

40 vulnerabilities (3 low, 14 moderate, 23 high)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
PS C:\laragon\www\FL\final-team04-lab-engce301\Step2> |
```

ทำการ build ไฟล์ด้วยคำสั่ง npm run dev

```
PS C:\laragon\www\FL\final-team04-lab-engce301\Step2> npm run build

> react_starter@1.0.0 build
> webpack

Hash: 9c3e03053ed09754794c
Version: webpack 4.47.0
Time: 1201ms
Built at: 10/25/2024 2:51:01 PM
    Asset      Size  Chunks             Chunk Names
bundle.js  1.22 MiB    main  [emitted]  main
Entrypoint main = bundle.js
[./node_modules/css-loader/dist/cjs.js!./src/css/styles.css] 1.6 KiB {main} [built]
[./src/actions/users.js] 107 bytes {main} [built]
[./src/app.js] 4.42 KiB {main} [built]
[./src/css/styles.css] 526 bytes {main} [built]
[./src/reducers/users.js] 1.52 KiB {main} [built]
[./src/store/store.js] 225 bytes {main} [built]
+ 84 hidden modules
PS C:\laragon\www\FL\final-team04-lab-engce301\Step2> |
```

RUN

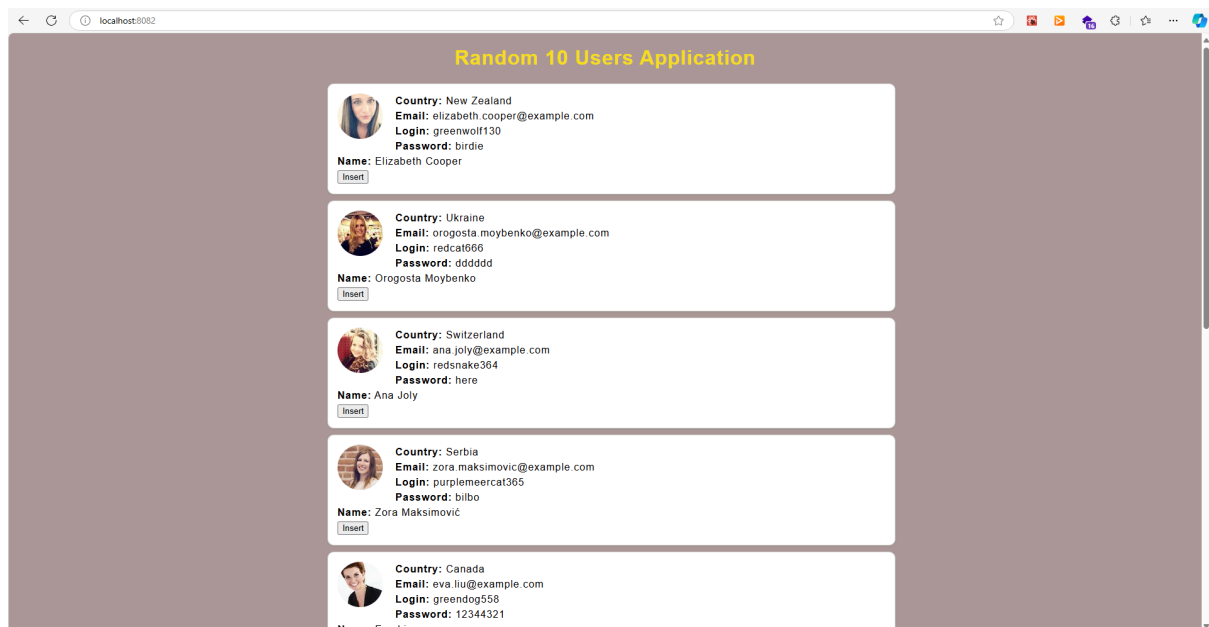
ใช้คำสั่ง npm run start-server แลดู nodemon ที่ port 8082

```
PS C:\laragon\www\FL\final-team04-lab-engce301\Step2> npm run start-server

> react_starter@1.0.0 start-server
> nodemon server/index.js

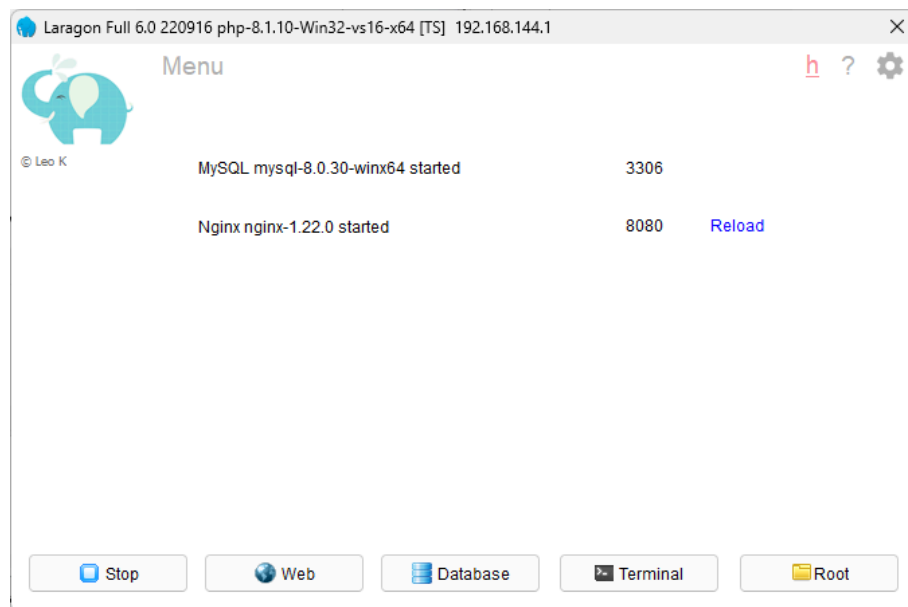
[nodemon] 2.0.22
[nodemon] to restart at any time, enter 'rs'
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting 'node server/index.js'
server started on port 8082
```

Result

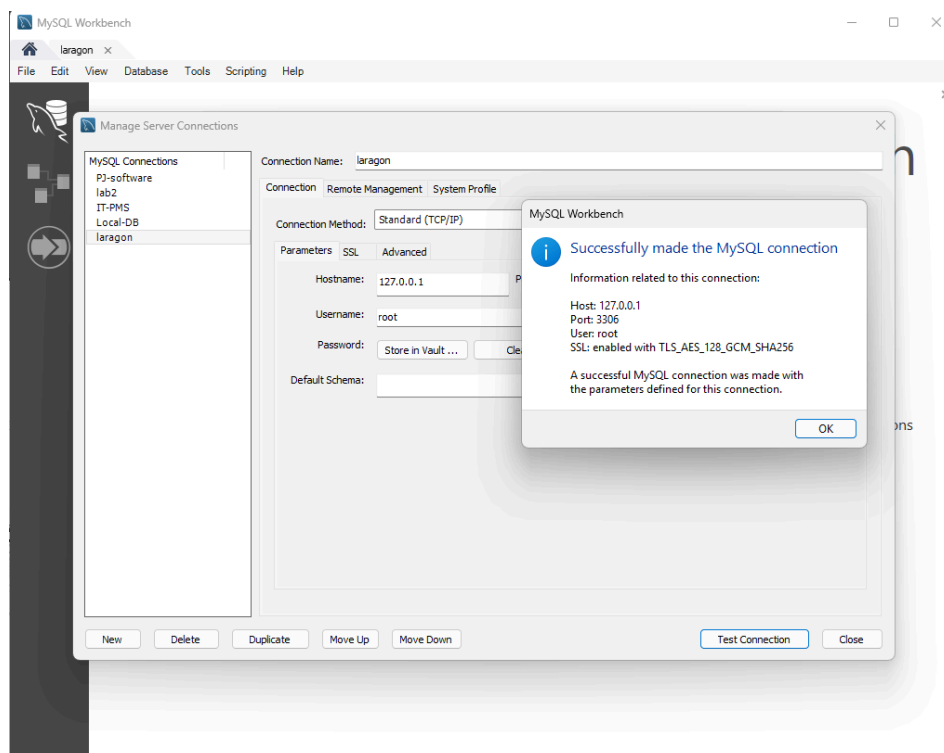


Step3

จะเป็นในส่วนของฐานข้อมูลจำต้องเปิดใช้ตัวจำลองฐานข้อมูล ในที่นี้จะใช้เป็น laragon



เชื่อมต่อฐานข้อมูลด้วย MySQL workbench 8.0 CE



สร้างฐานข้อมูลใน MySQL ให้มีโครงสร้างดังนี้

```
CREATE DATABASE term_project_db;  
SHOW DATABASES;  
  
USE term_project_db;  
  
SHOW TABLES;
```

```
CREATE TABLE users(
id INT NOT NULL AUTO_INCREMENT,
gender VARCHAR(30),
name_title VARCHAR(30),
name_first VARCHAR(100),
name_last VARCHAR(100),
country VARCHAR(100),
email VARCHAR(100),
dob DATETIME,
uuid VARCHAR(100),
username VARCHAR(100),
password VARCHAR(100),
md5 VARCHAR(100),
sha1 VARCHAR(100),
sha256 VARCHAR(100),
picture_large VARCHAR(100),
picture_medium VARCHAR(100),
picture_thumbnail VARCHAR(100),
PRIMARY KEY(id)
);
INSERT INTO users
(
    id,
    gender,
    name_title ,
    name_first,
    name_last,
    country,
    email,
    username,
    password ,
    picture_large,
    picture_medium,
    picture_thumbnail
)
VALUES (
    1,
    'female',
    'Mrs',
    'Pia',
    'Fossdal',
    'Norway',
    'pia.fossdal@example.com',
    'goldenwolf280',
    'preston',
    'https://randomuser.me/api/portraits/women/51.jpg',
    'https://randomuser.me/api/portraits/med/women/51.jpg',
    'https://randomuser.me/api/portraits/thumb/women/51.jpg'
);

SELECT * FROM users;
```

กด execute

The screenshot shows the MySQL Workbench interface. The 'Query' tab is active, displaying the following SQL code:

```
1 * CREATE DATABASE term_project_db;
2 * SHOW DATABASES;
3
4 * USE term_project_db;
5
6 * SHOW TABLES;
7
8 * CREATE TABLE users(
9   id INT NOT NULL AUTO_INCREMENT,
10  gender VARCHAR(30),
11  name_title VARCHAR(30),
12  name_first VARCHAR(100),
13  name_last VARCHAR(100),
14  country VARCHAR(100),
15  email VARCHAR(100),
16  dob DATETIME,
17  uuid VARCHAR(100),
18  username VARCHAR(100),
19  password VARCHAR(100),
20  mds VARCHAR(100))
```

The 'Schemas' pane on the left shows the 'term_project_db' database selected. The 'Table: users' pane shows the columns defined in the table. The 'Result Grid' shows the output of the queries, including the creation of the database and the table.

#	Time	Action	Message	Duration / Feat
6	11:33:39	SHOW TABLES	0 row(s) returned	0.016 sec / 0.000 sec
7	11:33:39	CREATE TABLE users(id INT NOT NULL AUTO_INCREMENT, gender VARCHAR(30), name_title VARCHAR(30), name_first VARCHAR(100), name_...	0 row(s) affected	0.016 sec
8	11:33:39	INSERT INTO users (id, gender, name_title, name_first, name_last, country, email, username, password, picture_large, picture_medium, picture_thumbnail	1 row(s) affected	0.000 sec
9	11:33:39	SELECT * FROM users LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Result

จะได้ผลลัพธ์ดังนี้

The screenshot shows the MySQL Workbench interface. The 'Query' tab is active, displaying the following SQL code:

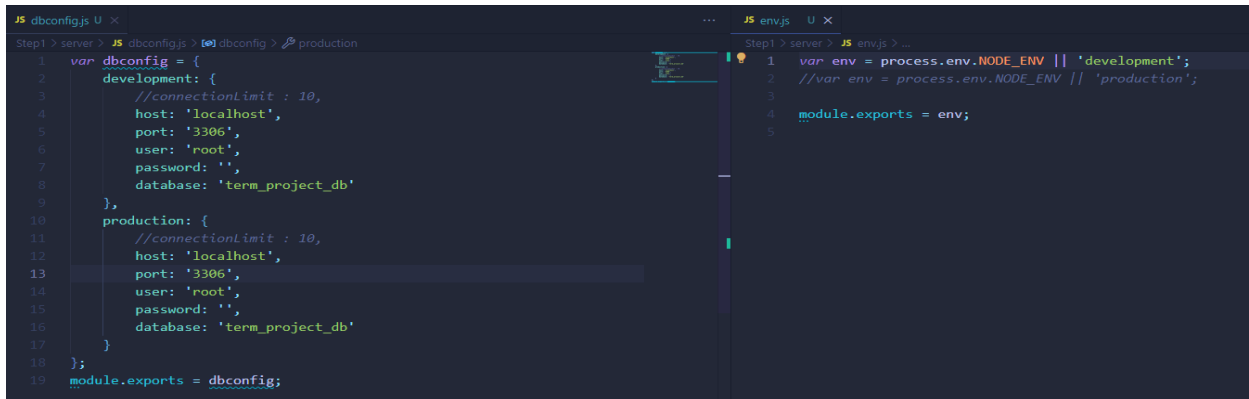
```
1 * INSERT INTO users (id, gender, name_title, name_first, name_last, country, email, username, password, picture_large, picture_medium, picture_thumbnail)
2 * VALUES (1, 'female', 'Mrs', 'Pia', 'Fossdal', 'Norway', 'pia.fossdal@example.com', 'poldmuvof280', 'preston', 'md5', 'sha1', 'sha256', 'https://randomuser.me/api/portraits/women/51...', 'https://randomuser.me/api/portraits/med/wom...', 'https://randomuser.me/api/portraits/thumb/51...');
```

The 'Result Grid' shows the output of the queries, including the insertion of data into the table.

#	Time	Action	Message	Duration / Feat
4	13:20:37	SELECT * FROM users LIMIT 0, 1000	6 row(s) returned	0.000 sec / 0.000 sec
5	13:50:44	SELECT * FROM users LIMIT 0, 1000	6 row(s) returned	0.000 sec / 0.000 sec
6	13:54:58	SELECT * FROM users LIMIT 0, 1000	7 row(s) returned	0.000 sec / 0.000 sec
7	15:01:37	SELECT * FROM users LIMIT 0, 1000	7 row(s) returned	0.000 sec / 0.000 sec

Step4

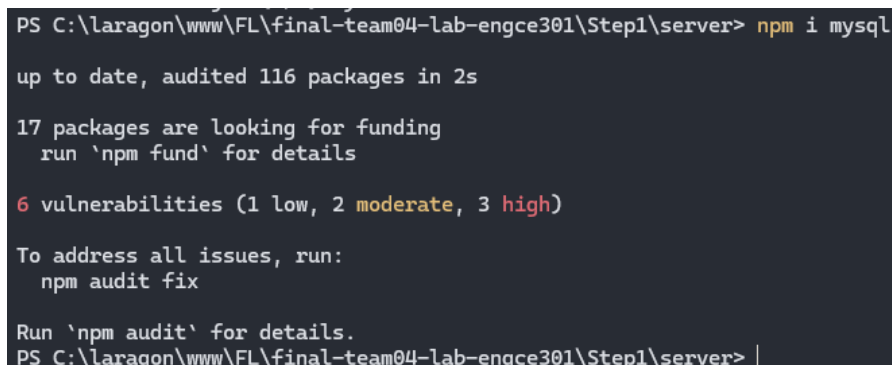
สร้างไฟล์ env.js , dbconfig.js ใน Step1/server/



```
Step1 > server > JS dbconfig.js U X
1 var dbconfig = {
2   development: {
3     //connectionLimit : 10,
4     host: 'localhost',
5     port: '3306',
6     user: 'root',
7     password: '',
8     database: 'term_project_db'
9   },
10  production: {
11    //connectionLimit : 10,
12    host: 'localhost',
13    port: '3306',
14    user: 'root',
15    password: '',
16    database: 'term_project_db'
17  }
18 };
19 module.exports = dbconfig;

Step1 > server > JS env.js U X
1 var env = process.env.NODE_ENV || 'development';
2 //var env = process.env.NODE_ENV || 'production';
3
4 module.exports = env;
5
```

ติดตั้ง mysql ใน Step1/server/ ด้วยคำสั่ง npm install mysql



```
PS C:\laragon\www\FI\final-team04-lab-engce301\Step1\server> npm i mysql

up to date, audited 116 packages in 2s

17 packages are looking for funding
  run `npm fund` for details

6 vulnerabilities (1 low, 2 moderate, 3 high)

To address all issues, run:
  npm audit fix

Run `npm audit` for details.
PS C:\laragon\www\FI\final-team04-lab-engce301\Step1\server> |
```

ปรับปรุงไฟล์ auth.js ใน Step1/server/controllers/

```
const { response } = require("express");

let mysql = require("mysql");
const env = require("../env.js");
const config = require("../dbconfig.js")[env];
const login = async (req, res = response) => {
  const { email, password } = req.body;
  let dbcon = mysql.createConnection(config);

  const userDetails = "SELECT * FROM users where email = '" + email + "'";
  console.log(userDetails);

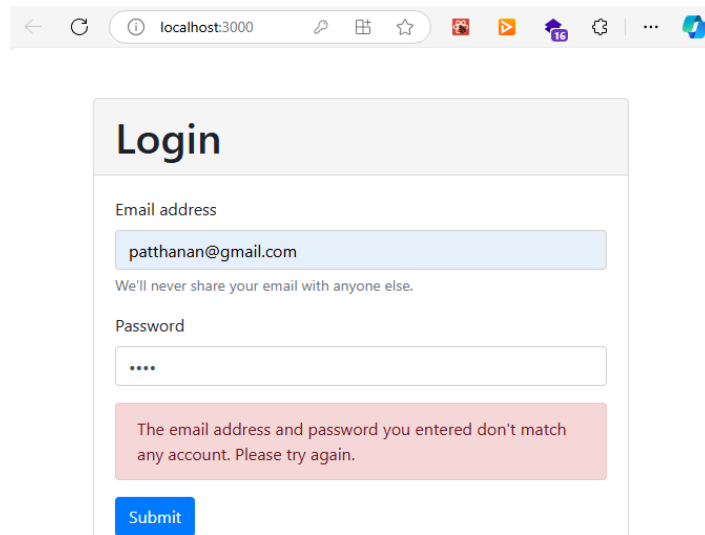
  dbcon.query(userDetails, function (err, user) {
    console.log(user);

    if (user.length > 0) {
      if (password !== user[0].password) {
        return res.status(400).json({
          msg: "User / Password are incorrect",
        });
      }
    }

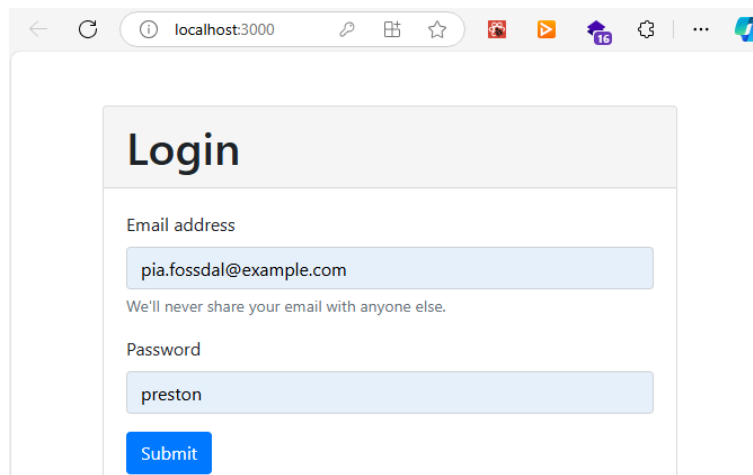
    res.status(200).json({ user });
  } else {
    // if (user.length > 0)
    return res.status(401).json({ message: "User not found !" });
  }
}
```

```
}  
});  
};  
  
module.exports = {  
  login,  
};
```

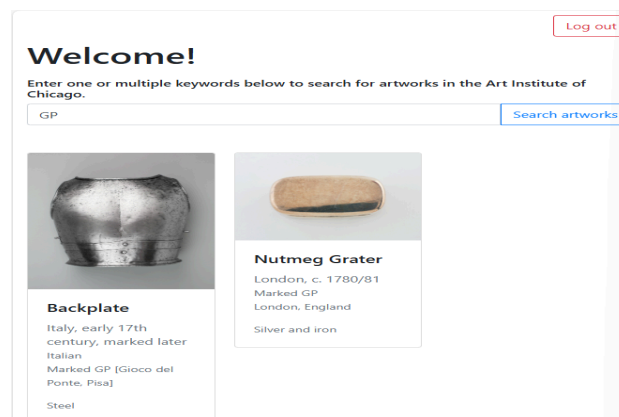
การเปลี่ยนแปลงนี้จะทำให้ไม่สามารถล็อกอินด้วย step1 ไม่ได้



แต่จะล็อกอินด้วย email password ที่มีอยู่ในฐานข้อมูลได้



สามารถค้นหาได้ตามปกติ



Step5

ปรับปรุงไฟล์ใน step3/server เพื่อใช้เป็น api ของ insert และ login สร้างไฟล์ user.js ที่ repository และเพิ่มโค้ดดังนี้

```
var mysql = require("mysql");
const env = require("../env.js");
const config = require("../dbconfig.js")[env];

async function getUserSearch(username_text, password_text) {
  var Query;
  var pool = mysql.createPool(config);

  return new Promise((resolve, reject) => {
    Query = `SELECT username, password FROM users WHERE (username LIKE '%${username_text}%') AND (password LIKE '%${password_text}%')`;

    pool.query(Query, function (error, results, fields) {
      if (error) throw error;

      console.log("results: " + results);
      console.log("results: " + JSON.stringify(results));
      console.log("results.length: " + results.length);

      if (results.length > 0) {
        pool.end();
        return resolve({
          statusCode: 200,
          returnCode: 1,
          data: results,
        });
      } else {
        pool.end();
        return resolve({
          statusCode: 404,
          returnCode: 11,
          message: "No User found",
        });
      }
    });
  });
}

async function postUser(
  p_gender,
  p_name_title,
  p_name_first,
  p_name_last,
  p_country,
  p_email,
  p_username,
  p_password,
  p_picture_large,
  p_picture_medium,
  p_picture_thumbnail
) {
  var Query;
  var pool = mysql.createPool(config);
```

```

return new Promise((resolve, reject) => {
    var post = {
        gender: p_gender,
        name_title: p_name_title,
        name_first: p_name_first,
        name_last: p_name_last,
        country: p_country,
        email: p_email,
        username: p_username,
        password: p_password,
        picture_large: p_picture_large,
        picture_medium: p_picture_medium,
        picture_thumbnail: p_picture_thumbnail,
    };

    console.log("post is: ", post);

    Query = "INSERT INTO users SET ?";
    pool.query(Query, post, function (error, results, fields) {

        if (error) {
            console.log("error: " + JSON.stringify(error));
            pool.end();
            return resolve({
                error: true,
                statusCode: 404,
                returnCode: 0,
                errMsg: error.code + ":" + error.sqlMessage,
            });
        } else if (results.affectedRows > 0) {
            console.log("results: " + JSON.stringify(results));

            pool.end();
            return resolve({
                error: false,
                statusCode: 200,
                returnCode: 1,
                message: "User list was inserted",
            });
        }
    });
});
}

module.exports.UserRepo = {
    getUserSearch: getUserSearch,
    postUser: postUser,
};

```

แก้ไขไฟล์ dbconfig.js ใน Step3/server ให้ไปยังฐานข้อมูลของเรา

```

var dbconfig = {
    development: {
        //connectionLimit : 10,
        host: 'localhost',

```

```

    port: '3306',
    user: 'root',
    password: "",
    database: 'term_project_db'
  },
  production: {
    //connectionLimit : 10,
    host: 'localhost',
    port: '3306',
    user: 'root',
    password: "",
    database: 'term_project_db'
  }
};
module.exports = dbconfig;

```

จากนั้นเรียกใช้ใน index.js

```
const Users = require('./respository/user');
```

และเพิ่ม server route ของ api login และ insert

```

server.route({
  method: 'POST',
  path: '/api/user/login',
  config: {
    payload: {
      multipart: true,
    },
    cors: {
      origin: ['*'],
      additionalHeaders: ['cache-control', 'x-requested-width'],
      credentials: true
    }
  },
  handler: async function (request, reply) {
    const {
      username_text,
      password_text,
    } = request.payload;
    console.log("request.payload: " + JSON.stringify(request.payload));
    try {
      const responsedata = await Users.UserRepo.getUserSearch(username_text, password_text);
      if (responsedata.error) {
        return responsedata;
      } else {
        return responsedata;
      }
    } catch (err) {
      server.log(["error", "home"], err);
    }
  }
});

```

```

        return err;
    }
}
});

server.route({
  method: 'POST',
  path: '/api/user/insert',
  config: {
    payload: {
      multipart: true,
    },
    cors: {
      origin: ['*'],
      additionalHeaders: ['cache-control', 'x-requested-width'],
      credentials: true
    }
  },
  handler: async function (request, reply) {
    const {
      gender,
      name_title ,
      name_first,
      name_last,
      country,
      email,
      username,
      password ,
      picture_large,
      picture_medium,
      picture_thumbnail
    } = request.payload;
    console.log("request.payload: " + JSON.stringify(request.payload));
    try {
      const responsedata = await Users.UserRepo.postUser( gender,
                                                            name_title ,
                                                            name_first,
                                                            name_last,
                                                            country,
                                                            email,
                                                            username,
                                                            password ,
                                                            picture_large,
                                                            picture_medium,
                                                            picture_thumbnail);

      if (responsedata.error) {
        return responsedata;
      } else {
        return responsedata;
      }
    } catch (err) {
      server.log(["error", "home"], err);
      return err;
    }
  }
});

```

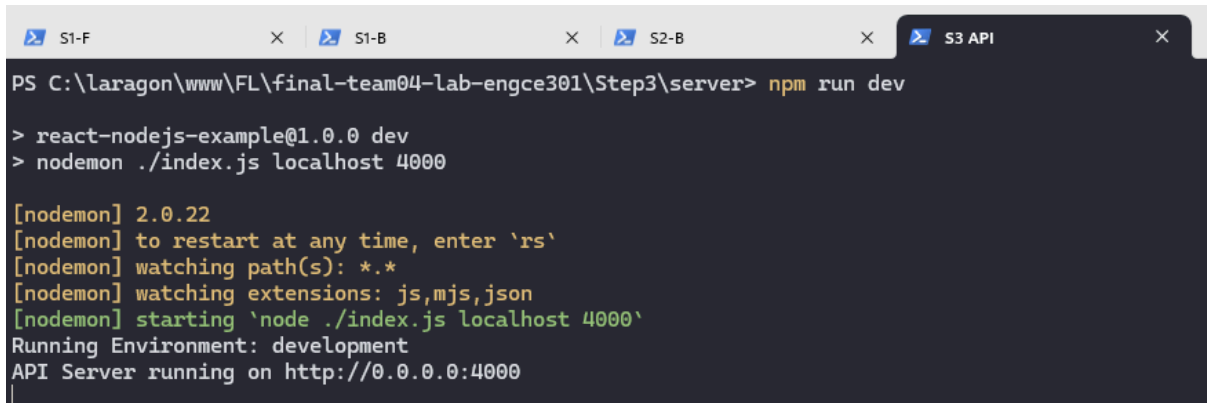
```

    await server.start();
    console.log('API Server running on %s', server.info.uri);

    //-----
};

```

ทำการ run api ใน step3./server ด้วยคำสั่ง npm run dev แบบ nodemon ที่พอร์ต 4000



```

PS C:\laragon\www\FL\final-team04-lab-engce301\Step3\server> npm run dev

> react-nodejs-example@1.0.0 dev
> nodemon ./index.js localhost 4000

[nodemon] 2.0.22
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node ./index.js localhost 4000`
Running Environment: development
API Server running on http://0.0.0.0:4000

```

จากนั้นทำการปรับปรุงไฟล์ index.js ที่ Step2/server เพิ่มโค้ดนี้

```

app.post("/api/user/insert", (req, res) => {
    const post_data = {
        records: [pre_post_data],
    };

    axios.post(updateCallTaskUrl, post_data).then(
        (response) => {
            if (response.data.results != null) {
                responseData = response.data.results[0];

                if (responseData.isSuccess) {
                    console.log("Success : " + responseData.isSuccess);
                } else {
                    console.log("Fail");
                }
            } else {
                console.log("");
            }
            console.log("----- END -----");
        },
        (error) => {
            console.log(error);
        }
    );
});

```

และปรับปรุง User.js ที่ Step2/src/components เพิ่มโค้ดนี้

```

import React from "react";

const User = ({gender,name,location,email,picture,login,createUser,}) => {
    async function createUser() {
        const token = "1234567890";
        const body = new FormData();
    }
}

```

```

body.set("gender", gender);
body.set("name_title", name.title);
body.set("name_first", name.first);
body.set("name_last", name.last);
body.set("country", location.country);
body.set("email", email);
body.set("username", login.username);
body.set("password", login.password);
body.set("picture_large", picture.large);
body.set("picture_medium", picture.medium);
body.set("picture_thumbnail", picture.thumbnail);

const response = await fetch('http://localhost:4000/api/user/insert', {
  method: "POST",
  headers: {
    Authorization: `Bearer ${token}`, // notice the Bearer before your token
  },
  body,
});
alert("User list was inserted");
}

return (
  <div className="random-user">
    <div className="user-image">
      <img src={picture.medium} alt={name.first} />
    </div>
    <div>
      <strong>Country:</strong> {location.country}
    </div>
    <div>
      <strong>Email:</strong> {email}
    </div>
    <div>
      <strong>Login:</strong> {login.username}
    </div>
    <div>
      <strong>Password:</strong> {login.password}
    </div>
    <div>
      <strong>Name:</strong> {name.first} {name.last}
    </div>
    <button
      type="button"
      onClick={(e) => createUser()}
      className="btn btn-danger"
    >
      Insert
    </button>
  </div>
);
};

export default User;

```


จากนั้นทำการ npm run build เพื่อให้หน้าเว็บเปลี่ยนแปลง และ npm run start-server ที่พอร์ต 8082

```
PS C:\laragon\www\FI\final-team04-lab-engce301\Step2> npm run build

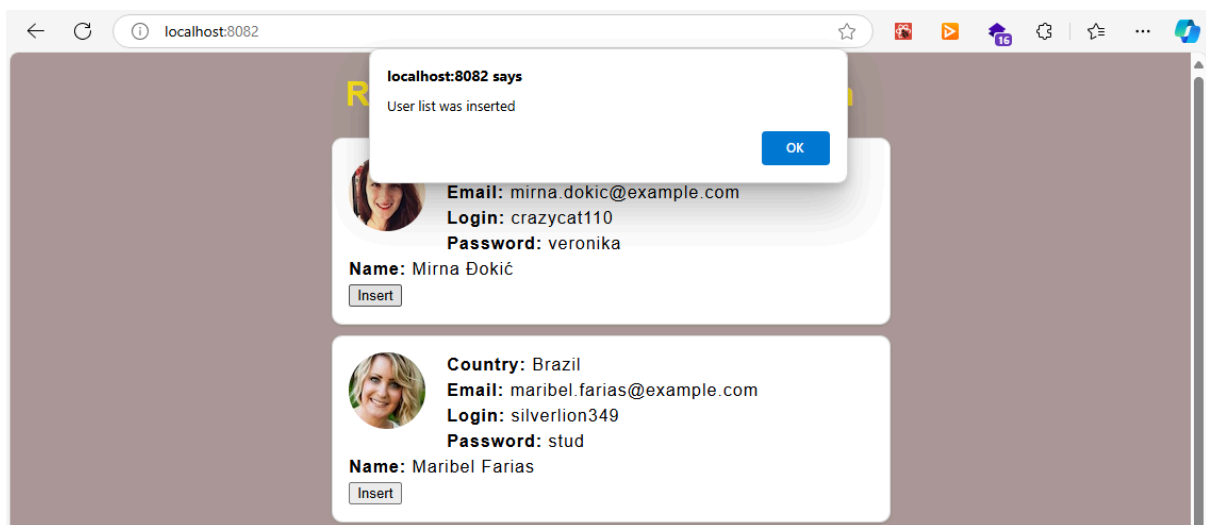
> react_starter@1.0.0 build
> webpack

Hash: fd2d8feb3b802dfe25e3
Version: webpack 4.47.0
Time: 1140ms
Built at: 10/25/2024 4:24:05 PM
    Asset      Size  Chunks             Chunk Names
bundle.js  1.22 MiB       0  [emitted]  main
Entrypoint main = bundle.js
./node_modules/css-loader/dist/cjs.js!./src/css/styles.css 1.6 KiB {main} [built]
./src/actions/users.js 107 bytes {main} [built]
./src/app.js 4.42 KiB {main} [built]
./src/css/styles.css 526 bytes {main} [built]
./src/reducers/users.js 1.52 KiB {main} [built]
./src/store/store.js 225 bytes {main} [built]
+ 84 hidden modules
PS C:\laragon\www\FI\final-team04-lab-engce301\Step2> npm run start-server

> react_starter@1.0.0 start-server
> nodemon server/index.js

[nodemon] 2.0.22
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node server/index.js`
server started on port 8082
```

ทดสอบเพิ่มผู้ใช้ 1 คน กดที่ปุ่ม Insert



ใช้คำสั่ง SELECT * FROM users; จากฐานข้อมูลจะมี user เพิ่มขึ้นมา

id	gender	name_title	name_first	name_last	country	email	dob	uuid	username	password	md5	sha1	sha256	picture_large	picture_medium	picture_thumbnail
1	female	Mrs	Pia	Fosdøl	Norway	pia.fosdøl@example.com	1968	0000	goldenwolf280	preston	0000	0000	0000	https://randomuser.me/api/portraits/women/51...	https://randomuser.me/api/portraits/med/wom...	https://randomuser.me/api/portraits/thumb/...
2	female	Madame	Kathi	Brun	Switzerland	kathi.brun@example.com	1968	0000	organicebra167	1966	0000	0000	0000	https://randomuser.me/api/portraits/women/52...	https://randomuser.me/api/portraits/med/wom...	https://randomuser.me/api/portraits/thumb/...
11	female	Ms	Mirna	Đokić	Serbia	mirna.dokic@example.com	1968	0000	crazycat110	veronika	0000	0000	0000	https://randomuser.me/api/portraits/women/75...	https://randomuser.me/api/portraits/med/wom...	https://randomuser.me/api/portraits/thumb/...

ในส่วนของการ Login ด้วย API ใช้ปรับปรุงดังนี้

ปรับปรุงไฟล์/index.js ที่ Step1/client/src/api ของ function login ดังนี้

```

export async function login({ email, password }) {
  const token = "1234567890";

  return await fetch("/api/auth/login", {
    method: "POST",
    body: JSON.stringify({ email, password }),
    headers: {
      "Content-Type": "application/json",
      Authorization: `Bearer ${token}`,
    },
  })
  .then((response) => {
    if (!response.ok) {
      throw new Error("HTTP status " + response.status);
    }

    return response.json();
  })
  .catch((err) => {
    console.log(err);
  });
}

```

ຢືນຢູ່ໄວ້/index.js ຕໍ່ Step1/client/src/components/login ບ່ອນ function login ດັ່ງນີ້

```

function Login({ onLoginSuccessful }) {
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [hasError, setHasError] = useState(false);

  const onEmailChange = (event) => setEmail(event.target.value);
  const onPasswordChange = (event) => setPassword(event.target.value);

  const onSubmit = async (event) => {
    event.preventDefault();
    setHasError(false);
    const loginResult = await login({ email, password });
    if (!loginResult) setHasError(true);
    else {
      const { name, token } = loginResult;
      // Save user IDs on local storage
      localStorage.setItem("name", name);
      localStorage.setItem("token", token);
      onLoginSuccessful();
    }
  };
}

```

ຢືນ proxy ບ່ອນ Client ຕໍ່ Step1/client/package.json ໃຫ້ມີ port API

```

"proxy": "http://localhost:4000",

```

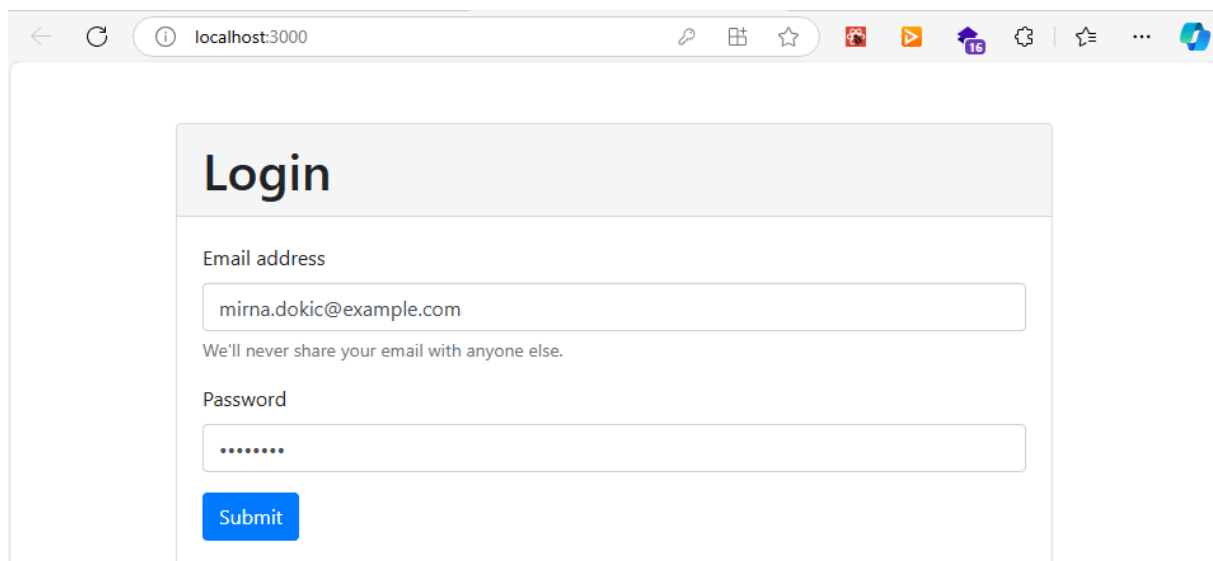
รับ Client ด้วย npm start

```
i [wds]: Project is running at http://192.168.110.1/
Starting the development server...
Compiled successfully!
localhost:3000
On Your Network: http://192.168.110.1:3000

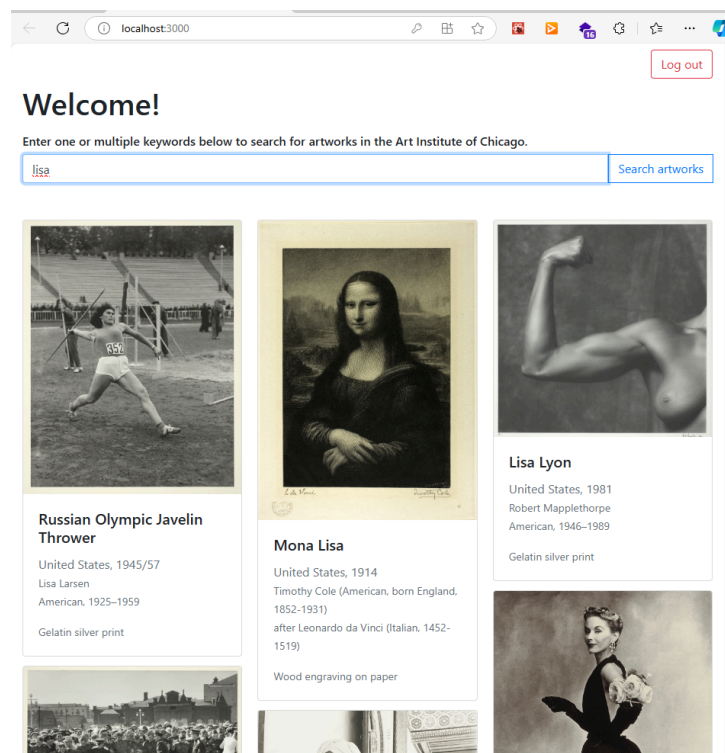
Note that the development build is not optimized.
To create a production build, use npm run build.
```

ทดสอบ login

ทดลองเข้าสู่ระบบจาก user ที่เพิ่มเข้ามาจาก step2




สามารถค้นหาได้ตามปกติ



Step6

<https://github.com/ltsPatthanan/final-team04-lab-engce301.git>


final-team04-lab-engce301
Public

[forked from ce-rmutl/1-67-engce301-final-lab](#)

main

1 Branch

Tags


Add file

<> Code

This branch is 2 commits ahead of [ce-rmutl/1-67-engce301-final-lab:main](#)

Contribute

Sync fork


ItsPatthan
๑๗๖๑๑๑

0b131a4 · 11 minutes ago

17 Commits

Step1	success	6 hours ago
Step2	success	6 hours ago
Step3	success	6 hours ago
.gitignore	Initial commit	8 months ago
README.md	Update README.md	yesterday

README

✎

1-67 Final Lab for ENGCE301 and ENGCE112

About

Final LAB for ENGCE301

Readme

Activity

0 stars

0 watching

0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Languages

JavaScript 90.6%

HTML 5.4%

CSS 4.0%