คู่มือ Grab Restaurant

Frontend Restaurant js

searchRestaurant.js

// delete

const deleteRestaurant = async (id) =>{

    //ฟังชั่น deleteRestaurant ที่เป็น async ด้านในจะมีการใช้ await

    if(id){ // ถ้ามี ข้อมูลใน id

        try {

            const restaurant =  await fetch(  // ตัวแปล restaurant ที่มีการ fetch

                "http://localhost:5000/apis/restaurants/" + id, //ทำการส่ง http://localhost:5000/apis/restaurants/ พร้อม id

                {

                    method: "DELETE", //ชนิดการยิง API DELETE

                    mode: "cors", //ตั่ง mode เป็นcors

                    cache: "no-cache", //สั่งไม่ให้ เก็บเเคช

                    credentials:"same-origin",

                    headers:{

                        "Content-Type":"application/json",//กำหนดชนิดเป็น json

                    },

                }

            )

                .then((response) => {//เมื่อมีการส่ง สำเร็จ

                    return response.json();//ส่ง การตอบกลับเป็น json

                })

                .then(()=>{

                    alert(`รหัสร้านอาหาร ${id} ถูกลบแล้ว`);// เเสดงข้อความ เเนบตัวแปร id Restaurant id ${id} is delete

                    location.reload();

                });

        } catch (error) { //หาก ทำงานไม่สำเร็จ

            alert(`รหัสร้านอาหาร ${id} not found`) // ให้เเสดง Restaurant id ${id} not found

        }

    }else{

        alert("ไม่มีรหัสร้านอาหาร")

        // ถ้า ไม่ใช่กรณีที่หาไม่เจอ ให้ขึ้นเเจ้งเตือน ว่า Restaurant id is missing ที่เเปลว่าไม่มี

    }

};

//card

const addRestaurant = (element) => {

    const item = document.createElement("div");//ตัวเเปร addRestaurant ที่เก็บฟังชั่น  ที่รับเอเลเมน

    item.className = "swiper"; //กำหนดชื่อ class

    item.style ="width: 34rem; margin: 1rem;" //กำหนด style

     //กำหนดรูปเเบบ ลงไปด้วย

     const card = `

     <div class="swiper-slide box">

         <img src="${element.imageurl}" alt="${element.name}">

         <h3>${element.name}</h3>

         <div class="price">${element.type}</div>

         <a href="editRestaurant.html?id=${element.id}" class="btn-warning"><i class="far fa-edit"></i> Edit</a>

         <a href="#" class="btn-danger" onclick="deleteRestaurant(${element.id})"><i class="far fa-trash-alt"></i> Delete</a>

     </div>`;

        item.innerHTML = card;//ค่าด้านในของ เอเลเมน

        const restaurantsElement = document.querySelector(".swiper"); //เข้าถึงคลาส restaurants

        restaurantsElement.appendChild(item);//ทำการเอา  item ที่มีข้อมูลเเล้วไปเพิ่มใน html

};

//ล้างให้ว่าง

const removeAllResult = () => {//ตัวแปล removeAllResult ที่เก็บฟังชั่น

    const restaurantsElement = document.querySelector(".swiper");//เข้าถึงคลาส restaurants

    restaurantsElement.innerHTML = "";//ล้างค่า ให้ว่าง

};

//ค้นหาร้านอาหาร ใช้Enter

const searchRestaurant = async (event)=>{

    const keyword = event.target.value;//ฟังชั่น searchRestaurant ที่เป็น async ด้านในจะมีการใช้ await

    if(event.key === "Enter" && keyword){//เปรียบเทียบ key ถ้ามีค่าเท่ากับ "Enter" หรือ keyword

        const allRestaurants = await fetch("http://localhost:5000/apis/restaurants", {

            method: "GET",//ชนิดการยิง API GET

            mode: "cors",//ตั่ง mode เป็นcors

            cache: "no-cache",//สั่งไม่ให้ เก็บเเคช

            credentials:"same-origin",

            headers:{

                "Content-Type":"application/json",//กำหนดชนิดเป็น json

            },

        }).then((response) => {

            return response.json();//ส่ง การตอบกลับเป็น json

        });

        //console.log(allRestaurants);

        const result = allRestaurants.filter(//ทำการฟิลเตอร์  allRestaurants เก็บไว้ใน  result

            (item) => item.name.includes(keyword) || item.type.includes(keyword)// เอาเฉพาะ ที่ตรงกับที่ ค้านหาหรือเหมือนที่ค้นหา

            );

            removeAllResult();//เรีกใช้ ฟังชั่น removeAllResult();

            result.forEach((element) => addRestaurant(element));

             //ลูป ข้อมูลใน result โดยลูปตาม element เเล้ว เรียกฟังชั่น addRestaurant(element)

    }

};

//เข้าถึงค้นหา โดยkeydown

const main = () =>{

    const inputElement = document.querySelector(".search");//ทำการ เข้าถึง search

    inputElement.addEventListener("keydown",searchRestaurant); //ทำการ รอการ keydown ถ้ามีการ พิมจะเรียกใช้ searchRestaurant

};

main();

allRestaurant.js

//all

const init = async()=>{

    const allRestuarants = await fetch(

        "http://localhost:5000/apis/restaurants",{

            method: "GET",

            mode: "cors",

            cache: "no-cache",

            credentials: "same-origin",

            headers: {

              "Content-Type": "application/json",

            },

    }).then((response)=> response.json());

    allRestuarants.forEach((element) => addRestaurants(element));

}

//card

const addRestaurants = (element) => {

    const item = document.createElement("div"); //สร้าง div

    item.className = "swiper"; //กำหนดชื่อ class

    item.style ="width: 35rem; margin: 1rem;" //กำหนด style

     //กำหนดรูปเเบบ ลงไปด้วย

     const card = `

     <div class="swiper-slide box">

         <img src="${element.imageurl}" alt="${element.name}">

         <h3>${element.name}</h3>

         <div class="price">${element.type}</div>

         <a href="editRestaurant.html?id=${element.id}" class="btn-warning"><i class="far fa-edit"></i> Edit</a>

         <a href="#" class="btn-danger" onclick="deleteRestaurant(${element.id})"><i class="far fa-trash-alt"></i> Delete</a>

     </div>`;

        item.innerHTML = card;  //เอาไปแทรกที่card ลงใน div

    const restaurantsElement = document.querySelector(".swiper"); //เข้าถึง class หน้า HTML

    restaurantsElement.appendChild(item); //เพิ่มลงไป

}

//DELETE

const deleteRestaurant = async (id) =>{

    //ฟังชั่น deleteRestaurant ที่เป็น async ด้านในจะมีการใช้ await

    if(id){ // ถ้ามี ข้อมูลใน id

        try {

            const restaurant =  await fetch(  // ตัวแปล restaurant ที่มีการ fetch

                "http://localhost:5000/apis/restaurants/" + id, //ทำการส่ง http://localhost:5000/apis/restaurants/ พร้อม id

                {

                    method: "DELETE", //ชนิดการยิง API DELETE

                    mode: "cors", //ตั่ง mode เป็นcors

                    cache: "no-cache", //สั่งไม่ให้ เก็บเเคช

                    credentials:"same-origin",

                    headers:{

                        "Content-Type":"application/json",//กำหนดชนิดเป็น json

                    },

                }

            )

                .then((response) => {//เมื่อมีการส่ง สำเร็จ

                    return response.json();//ส่ง การตอบกลับเป็น json

                })

                .then(()=>{

                    alert(`รหัสร้านอาหาร ${id} ถูกลบแล้ว`);// เเสดงข้อความ เเนบตัวแปร id Restaurant id ${id} is delete

                    location.reload();

                });

        } catch (error) { //หาก ทำงานไม่สำเร็จ

            alert(`รหัสร้านอาหาร ${id} not found`) // ให้เเสดง Restaurant id ${id} not found

        }

    }else{

        alert("ไม่มีรหัสร้านอาหาร")

        // ถ้า ไม่ใช่กรณีที่หาไม่เจอ ให้ขึ้นเเจ้งเตือน ว่า Restaurant id is missing ที่เเปลว่าไม่มี

    }

};

addRestaurant.js

//add

const add = async() => {

    // const id = Number.parseInt(document.getElementById("id").value);

     const name = document.getElementById("name").value;

     const type = document.getElementById("type").value;

     const imageurl = document.getElementById("imageurl").value;

     if( name && type && imageurl){

         const perams = {

            //id: id,

             name: name,

             type: type,

             imageurl: imageurl,

         }

         try{

             const restaurant =  await fetch(

                 "http://localhost:5000/apis/restaurants",

                 {

                 method: "POST",

                 mode: "cors",

                 cache: "no-cache",

                 credentials:"same-origin",

                 headers:{

                     "Content-Type":"application/json"

                 },

                 body: JSON.stringify(perams),

             }).then((response) => {

                 return response.json(); //ส่งค่าในรูปแบบ json

             }).then(() => {

                 alert(`ร้านอาหาร ${name} กำลังอัพเดท`);

                 location.replace("allRestaurant.html");

             })

     }catch(error){

         alert(`ร้านอาหาร ${name} not found`);

     }

     }else{

     alert("ไม่มีข้อมูลร้านอาหาร");

     }

  }

editRestaurant.js

const init = async () =>{

    let params = new URL(document.location).searchParams;

    let id = params.get("id");

    if(id){

        try{

            const restaurant = await fetch(

                "http://localhost:5000/apis/restaurants/" + id,{ //ต่อไอดีที่ส่งมาจากการกำปุ่ม Edit

                method: "GET",

                mode:"cors",

                cache:"no-cache",               //6-8 บอกว่า server อยู่ที่เดียวกัน

                credentals:"same-origin",

                headers:{

                    "Content-type":"application/json"  //ข้อมุลอยู่ในรูปแบบ json

                },

            }).then((response)=>{

                return response.json();  //ส่งค่าในรูปแบบ json

            });

            //set input value 19-22

            document.getElementById("id").value = restaurant.id;

            document.getElementById("name").value = restaurant.name;

            document.getElementById("type").value = restaurant.type;

            document.getElementById("imageurl").value = restaurant.imageurl;

        }catch (error){

            alert(`รหัสร้านอาหาร:${id} ไม่พบ`)

        }

    }else{

        alert("ไม่มีรหัสร้านอาหาร");

    }

  }

//แก้ไข

const edit = async () => {

    const id = document.getElementById("id").value;

    if (id) {

        const params = {

            id: document.getElementById("id").value,

            name: document.getElementById("name").value,

            type: document.getElementById("type").value,

            imageurl: document.getElementById("imageurl").value,

        };

      try {

        const restaurant = await fetch(

          "http://localhost:5000/apis/restaurants/" + id,

          {

            method: "PUT",

            mode: "cors",

            cache: "no-cache",

            credentials: "same-origin",

            headers: {

              "Content-Type": "application/json",

            },

            body: JSON.stringify(params), // เพิ่ม data

          }

        ).then((response) => {

          return response.json();

        }).then(()=>{

          alert(`รหัสร้านอาหาร:${id} กำลังอัปเดต`);

          location.replace("allRestaurant.html");

        });

      } catch (error) {

        alert(`รหัสร้านอาหาร:${id} ไม่พบ`);

      }

    } else {

      alertalert("ไม่มีรหัสร้านอาหาร");

    }

  };

Frontend User js

searchUser.js

// delete

const deleteUser = async (id) =>{

    //ฟังชั่น deleteUser ที่เป็น async ด้านในจะมีการใช้ await

    if(id){ // ถ้ามี ข้อมูลใน id

        try {

            const user =  await fetch(  // ตัวแปล user ที่มีการ fetch

                "http://localhost:5000/apis/user/" + id, //ทำการส่ง http://localhost:5000/apis/user/ พร้อม id

                {

                    method: "DELETE", //ชนิดการยิง API DELETE

                    mode: "cors", //ตั่ง mode เป็นcors

                    cache: "no-cache", //สั่งไม่ให้ เก็บเเคช

                    credentials:"same-origin",

                    headers:{

                        "Content-Type":"application/json",//กำหนดชนิดเป็น json

                    },

                }

            )

                .then((response) => {//เมื่อมีการส่ง สำเร็จ

                    return response.json();//ส่ง การตอบกลับเป็น json

                })

                .then(()=>{

                    alert(`รหัสร้านอาหาร ${id} ถูกลบแล้ว`);// เเสดงข้อความ เเนบตัวแปร id user id ${id} is delete

                    location.replace("allUser.html");

                });

        } catch (error) { //หาก ทำงานไม่สำเร็จ

            alert(`รหัสร้านอาหาร ${id} not found`) // ให้เเสดง user id ${id} not found

        }

    }else{

        alert("ไม่มีรหัสร้านอาหาร")

        // ถ้า ไม่ใช่กรณีที่หาไม่เจอ ให้ขึ้นเเจ้งเตือน ว่า user id is missing ที่เเปลว่าไม่มี

    }

};

//card

const addUser = (element) => {

    const item = document.createElement("div");//ตัวเเปร addUser ที่เก็บฟังชั่น  ที่รับเอเลเมน

    item.className = "container"; //กำหนดชื่อ class

    item.style ="width: 32rem;" //กำหนด style

     //กำหนดรูปเเบบ ลงไปด้วย

     const card = `

            <div class="card">

                <div class="card-header">

                    <img src="${element.profile}" alt="${element.firstName}" />

                </div>

                    <div class="card-body">

                        <span class="tag tag-teal">${element.pocition}</span>

                            <h4>Name : ${element.firstName} ${element.lastName}</h4>

                            <p>Gender : ${element.firstName} </p>

                            <p>Tel : ${element.gender}</p>

                            <p>Address : ${element.address}</p>

                    <div class="user">

                        <a href="editUser.html?id=${element.id}" class="btn-warning"><i class="far fa-edit"></i></a>

                        <a href="#" class="btn-danger" onclick="deleteUser(${element.id})"><i class="far fa-trash-alt"></i></a>

                    </div>

                </div>

            </div>`;

        item.innerHTML = card;//ค่าด้านในของ เอเลเมน

        const userElement = document.querySelector(".container"); //เข้าถึงคลาส restaurants

        userElement.appendChild(item);//ทำการเอา  item ที่มีข้อมูลเเล้วไปเพิ่มใน html

};

//ล้างให้ว่าง

const removeAllResult = () => {//ตัวแปล removeAllResult ที่เก็บฟังชั่น

    const userElement = document.querySelector(".container");//เข้าถึงคลาส restaurants

    userElement.innerHTML = "";//ล้างค่า ให้ว่าง

};

//ค้นหาร้านอาหาร ใช้Enter

const searchUser = async (event)=>{

    const keyword = event.target.value;//ฟังชั่น searchUser ที่เป็น async ด้านในจะมีการใช้ await

    if(event.key === "Enter" && keyword){//เปรียบเทียบ key ถ้ามีค่าเท่ากับ "Enter" หรือ keyword

        const allUser = await fetch("http://localhost:5000/apis/user", {

            method: "GET",//ชนิดการยิง API GET

            mode: "cors",//ตั่ง mode เป็นcors

            cache: "no-cache",//สั่งไม่ให้ เก็บเเคช

            credentials:"same-origin",

            headers:{

                "Content-Type":"application/json",//กำหนดชนิดเป็น json

            },

        }).then((response) => {

            return response.json();//ส่ง การตอบกลับเป็น json

        });

        //console.log(allUser);

        const result = allUser.filter(//ทำการฟิลเตอร์  allUser เก็บไว้ใน  result

            (item) => item.firstName.includes(keyword) || item.pocition.includes(keyword)// เอาเฉพาะ ที่ตรงกับที่ ค้านหาหรือเหมือนที่ค้นหา

            );

            removeAllResult();//เรีกใช้ ฟังชั่น removeAllResult();

            result.forEach((element) => addUser(element));

             //ลูป ข้อมูลใน result โดยลูปตาม element เเล้ว เรียกฟังชั่น addUser(element)

    }

};

//เข้าถึงค้นหา โดยkeydown

const main = () =>{

    const inputElement = document.querySelector(".search");//ทำการ เข้าถึง search

    inputElement.addEventListener("keydown",searchUser); //ทำการ รอการ keydown ถ้ามีการ พิมจะเรียกใช้ searchUser

};

main();

editUser.js

const init = async () =>{

    let params = new URL(document.location).searchParams;

    let id = params.get("id");

    if(id){

        try{

            const User = await fetch(

                "http://localhost:5000/apis/user/" + id,{ //ต่อไอดีที่ส่งมาจากการกำปุ่ม Edit

                method: "GET",

                mode:"cors",

                cache:"no-cache",               //6-8 บอกว่า server อยู่ที่เดียวกัน

                credentals:"same-origin",

                headers:{

                    "Content-type":"application/json"  //ข้อมุลอยู่ในรูปแบบ json

                },

            }).then((response)=>{

                return response.json();  //ส่งค่าในรูปแบบ json

            });

            //set input value 19-22

            document.getElementById("id").value = User.id;

            document.getElementById("profile").value = User.profile;

            document.getElementById("firstName").value = User.firstName;

            document.getElementById("lastName").value = User.lastName;

            document.getElementById("pocition").value = User.pocition;

            document.getElementById("gender").value = User.gender;

            document.getElementById("tel").value = User.tel;

            document.getElementById("address").value = User.address;

        }catch (error){

            alert(`รหัสผู้ใช้งาน:${id} ไม่พบ`)

        }

    }else{

        alert("ไม่มีรหัสผู้ใช้งาน");

    }

  }

//แก้ไข

const edit = async () => {

    const id = document.getElementById("id").value;

    if (id) {

        const params = {

            id: document.getElementById("id").value,

            firstName: document.getElementById("firstName").value,

            lastName: document.getElementById("lastName").value,

            tel: document.getElementById("tel").value,

            address: document.getElementById("address").value,

        };

      try {

        const user = await fetch(

          "http://localhost:5000/apis/user/" + id,

          {

            method: "PUT",

            mode: "cors",

            cache: "no-cache",

            credentials: "same-origin",

            headers: {

              "Content-Type": "application/json",

            },

            body: JSON.stringify(params), // เพิ่ม data

          }

        ).then((response) => {

          return response.json();

        }).then(()=>{

          alert(`รหัสผู้ใช้งาน:${id} กำลังอัปเดต`);

          location.replace("allUser.html");

        });

      } catch (error) {

        alert(`รหัสผู้ใช้งาน:${id} ไม่พบ`);

      }

    } else {

      alertalert("ไม่มีรหัสผู้ใช้งาน");

    }

  };

allUser.js

//all

const init = async()=>{

    const allUser = await fetch(

        "http://localhost:5000/apis/user",{

            method: "GET",

            mode: "cors",

            cache: "no-cache",

            credentials: "same-origin",

            headers: {

              "Content-Type": "application/json",

            },

    }).then((response)=> response.json());

    allUser.forEach((element) => addUser(element));

}

//card

const addUser = (element) => {

    const item = document.createElement("div"); //สร้าง div

    item.className = "container"; //กำหนดชื่อ class

    item.style ="width: 32rem;" //กำหนด style

     //กำหนดรูปเเบบ ลงไปด้วย

     const card = `

            <div class="card">

                <div class="card-header">

                    <img src="${element.profile}" alt="${element.firstName}" />

                </div>

                    <div class="card-body">

                        <span class="tag tag-teal">${element.pocition}</span>

                            <h4>Name : ${element.firstName} ${element.lastName}</h4>

                            <p>Gender : ${element.firstName} </p>

                            <p>Tel : ${element.gender}</p>

                            <p>Address : ${element.address}</p>

                    <div class="user">

                        <a href="editUser.html?id=${element.id}" class="btn-warning"><i class="far fa-edit"></i></a>

                        <a href="#" class="btn-danger" onclick="deleteUser(${element.id})"><i class="far fa-trash-alt"></i></a>

                    </div>

                </div>

            </div>`;

        item.innerHTML = card;  //เอาไปแทรกที่card ลงใน div

    const UserElement = document.querySelector(".container"); //เข้าถึง class หน้า HTML

    UserElement.appendChild(item); //เพิ่มลงไป

}

//DELETE

const deleteUser = async (id) =>{

    //ฟังชั่น deleteUser ที่เป็น async ด้านในจะมีการใช้ await

    if(id){ // ถ้ามี ข้อมูลใน id

        try {

            const user =  await fetch(  // ตัวแปล restaurant ที่มีการ fetch

                "http://localhost:5000/apis/user/" + id, //ทำการส่ง http://localhost:5000/apis/user/ พร้อม id

                {

                    method: "DELETE", //ชนิดการยิง API DELETE

                    mode: "cors", //ตั่ง mode เป็นcors

                    cache: "no-cache", //สั่งไม่ให้ เก็บเเคช

                    credentials:"same-origin",

                    headers:{

                        "Content-Type":"application/json",//กำหนดชนิดเป็น json

                    },

                }

            )

                .then((response) => {//เมื่อมีการส่ง สำเร็จ

                    return response.json();//ส่ง การตอบกลับเป็น json

                })

                .then(()=>{

                    alert(`รหัสผู้ใช้งาน ${id} ถูกลบแล้ว`);// เเสดงข้อความ เเนบตัวแปร id user id ${id} is delete

                    location.replace("allUser.html");

                });

        } catch (error) { //หาก ทำงานไม่สำเร็จ

            alert(`รหัสผู้ใช้งาน ${id} not found`) // ให้เเสดง user id ${id} not found

        }

    }else{

        alert("ไม่มีรหัสผู้ใช้งาน")

        // ถ้า ไม่ใช่กรณีที่หาไม่เจอ ให้ขึ้นเเจ้งเตือน ว่า user id is missing ที่เเปลว่าไม่มี

    }

};

addUser.js

//add

const addUser = async() => {

    // const id = Number.parseInt(document.getElementById("id").value);

     const profile = document.getElementById("profile").value;

     const firstName = document.getElementById("firstName").value;

     const lastName = document.getElementById("lastName").value;

     const pocition = document.getElementById("pocition").value;

     const gender = document.getElementById("gender").value;

     const tel = document.getElementById("tel").value;

     const address = document.getElementById("address").value;

     if( profile && firstName && lastName && tel && address){

         const perams = {

            //id: id,

            profile: profile,

            firstName: firstName,

            lastName: lastName,

            pocition: pocition,

            gender: gender,

            tel: tel,

            address: address,

         }

         try{

             const user =  await fetch(

                 "http://localhost:5000/apis/user",

                 {

                 method: "POST",

                 mode: "cors",

                 cache: "no-cache",

                 credentials:"same-origin",

                 headers:{

                     "Content-Type":"application/json"

                 },

                 body: JSON.stringify(perams),

             }).then((response) => {

                 return response.json(); //ส่งค่าในรูปแบบ json

             }).then(() => {

                 alert(`ผู้ใช้งานชื่อ ${firstName} กำลังอัพเดท`);

                 location.replace("allUser.html");

             })

     }catch(error){

         alert(`ผู้ใช้งานชื่อ ${firstName} not found`);

     }

     }else{

     alert("ไม่มีข้อมูลผู้ใช้งาน");

     }

  }

Resful\_SQL\_API

Configs -> db.configs.js

module.exports = {

    HOST: "localhost",

    USER: "root",

    PASSWORD: "",

    DB: "restaurants",

};

Models -> db.js

const mysql = require("mysql");

const dbConfig = require("../configs/db.configs");

//Create Connection Database

const connection = mysql.createConnection({

    host: dbConfig.HOST,

    user: dbConfig.USER,

    password: dbConfig.PASSWORD,

    database: dbConfig.DB,

});

//Open the my SQL connection

connection.connect((error)=>{

    if(error) throw error;

    console.log("เชื่อมต่อกับฐานข้อมูลเรียบร้อยแล้ว ...");

})

module.exports = connection;

Models -> restaurant.model.js

const sql = require("./db");

//constructor

const Restaurant = function (restaurants){  //ใช้ arrow function ไม่สามารถกำหนด คอนสเต็คเตอร์ได้

    //Attributes

    this.id = restaurants.id;

    this.name = restaurants.name;

    this.type = restaurants.type;

    this.imageurl = restaurants.imageurl;

};

//Method insert Data

Restaurant.create = (newRestaurant, result) =>{

    //INSERT INTO restaurants SET id, name , type ,imageurl Values ("1","KFC","FastFood","url")

    sql.query("INSERT INTO restaurants SET ?",newRestaurant,(err, res)=>{

        if(err){

            console.log("error",err);

            result(err,null);

            return;

        }

        console.log("created restaurant:",{id:res.insertId, ...newRestaurant});

        result(null,{ id: res.insertId, ...newRestaurant});

    })

};

//Getdata byId

Restaurant.getById = (restaurantId, result) =>{

    //SELECT \* FROM restaurants where id = restaurantId

    sql.query(`SELECT \* FROM restaurants WHERE id = ${restaurantId}`,

    (err, res)=>{

        if(err){  //ถ้ามี error ค่าข้อมูลจะว่าง

            console.log("error ",err);

            result(err, null);

            return;

        }

        if(res.length){ //ถ้าหากเจอข้อมูล

            result(null, res[0]); //ส่งข้อมูล array ตำแหน่งที่ 1 กลับมา

            return;

        }

        //restaurant not found  with this Id

        result({kind: "not\_found"}, null);

        }

    );

};

//Get All restaurantrant

Restaurant.getAll = (result ) =>{

    //SELECT \* FROM restaurants

    sql.query("SELECT \* FROM restaurants", (err, res)=>{

        if(err){  //ถ้ามี error ค่าข้อมูลจะว่าง

            console.log("error ",err);

            result(err, null);

            return;

        }

        result(null ,res);

    });

};

//Update Restaurant Data

Restaurant.updateById = (id, restaurant,result) =>{

//UPDATE restaurant SET name = "name" ,type = "type", imageurl = "imageurl" WHERE id = "id"

sql.query(

    "UPDATE restaurants SET name = ? ,type = ?, imageurl = ? WHERE id = ?",

    [restaurant.name, restaurant.type, restaurant.imageurl, id],

    (err,res)=>{

        if(err){  //ถ้ามี error ค่าข้อมูลจะว่าง

            console.log("error ",err);

            result(err, null);

            return;

        }

        if(res.affectedRows == 0){ //เช็คแถวที่อัพเดท

            result({ kind: "not\_found"}, null)

            return;

        }

        //Restaurant data is updated

        result(null, {id:id, ...restaurant });

        }

    );

};

//Delete restaurant by Id

Restaurant.removeById = (id,result) =>{

//DELETE FROM restaurants WHERE id = ?

sql.query("DELETE FROM restaurants WHERE id = ?", id , (err, res)=>{

    if(err){

        console.log("error ",err);

            result(err, null);

            return;

    }

    if(res.affectedRows == 0){ //เช็คแถวที่อัพเดท

        result({ kind: "not\_found"}, null)

        return;

    }

    console.log("Delete restaurant with id : ", id);

    result(null, res);

});

};

Restaurant.removeAll = () =>{

};

module.exports = Restaurant;

Models -> User.model.js

const sql = require("./db");

//constructor

const User = function (user){  //ใช้ arrow function ไม่สามารถกำหนด คอนสเต็คเตอร์ได้

    //Attributes

    this.id = user.id;

    this.profile = user.profile;

    this.firstName = user.firstName;

    this.lastName = user.lastName;

    this.pocition = user.pocition;

    this.gender = user.gender;

    this.tel = user.tel;

    this.address = user.address;

};

//Method insert Data

User.create = (newUser, result) =>{

    //INSERT INTO user (id, firstName, lastName, tel, address) VALUES ([value-1],[value-2],[value-3],[value-4],[value-5])

    sql.query("INSERT INTO user SET ?",newUser,(err, res)=>{

        if(err){

            console.log("error",err);

            result(err,null);

            return;

        }

        console.log("created user:",{id:res.insertId, ...newUser});

        result(null,{ id: res.insertId, ...newUser});

    })

};

//Getdata byId

User.getById = (userId, result) =>{

    //SELECT \* FROM user where u\_id  = 2

    sql.query(`SELECT \* FROM user WHERE id = ${userId}`,

    (err, res)=>{

        if(err){  //ถ้ามี error ค่าข้อมูลจะว่าง

            console.log("error ",err);

            result(err, null);

            return;

        }

        if(res.length){ //ถ้าหากเจอข้อมูล

            result(null, res[0]); //ส่งข้อมูล array ตำแหน่งที่ 1 กลับมา

            return;

        }

        //restaurant not found  with this Id

        result({kind: "not\_found"}, null);

        }

    );

};

//Get All User

User.getAll = (result ) =>{

    //SELECT \* FROM user

    sql.query("SELECT \* FROM user", (err, res)=>{

        if(err){  //ถ้ามี error ค่าข้อมูลจะว่าง

            console.log("error ",err);

            result(err, null);

            return;

        }

        result(null ,res);

    });

};

//Update Restaurant Data

User.updateById = (id, user,result) =>{

    sql.query(

        //UPDATE `user` SET `id`='[value-1]',`firstName`='[value-2]',`lastName`='[value-3]',`tel`='[value-4]',`address`='[value-5]',`district`='[value-6]',`province`='[value-7]' WHERE 1

        "UPDATE user SET firstName = ?, lastName = ?, tel = ?, address = ?  WHERE id = ?",

        [user.firstName,user.lastName,user.tel,user.address, id],

        (err,res)=>{

            if(err){  //ถ้ามี error ค่าข้อมูลจะว่าง

                console.log("error ",err);

                result(err, null);

                return;

            }

            if(res.affectedRows == 0){ //เช็คแถวที่อัพเดท

                result({ kind: "not\_found"}, null)

                return;

            }

            //Restaurant data is updated

            result(null, {id:id, ...user });

            }

        );

    };

//Delete restaurant by Id

User.removeById = (id,result) =>{

    //DELETE FROM user WHERE u\_id = ?

    sql.query("DELETE FROM user WHERE id = ?", id , (err, res)=>{

        if(err){

            console.log("error ",err);

                result(err, null);

                return;

        }

        if(res.affectedRows == 0){ //เช็คแถวที่อัพเดท

            result({ kind: "not\_found"}, null)

            return;

        }

        console.log("ลบผู้ใช้งานรหัส : ", id);

        result(null, res);

    });

    };

module.exports = User;

routes -> restaurant.router.js

const express = require('express');

const router = express.Router();

const Restaurant = require("../models/restaurant.model");

//Method insert Data

// http://localhost:5000/apis/restaurants

router.post("/restaurants",(req,res) =>{

    //Create a restaurant

    const newRestaurant = new Restaurant({

        name:req.body.name,

        type:req.body.type,

        imageurl:req.body.imageurl,

    });

    //Save to database

    Restaurant.create(newRestaurant,(err, data)=>{

        if(err)

            res.status(500).send({

                message: err.message || "เกิดข้อผิดพลาดขณะสร้างร้านอาหาร"

            })

        else res.send(data);

    })

});

//แสดงตามid

// http://localhost:5000/apis/restaurants/1

router.get('/restaurants/:id', (req,res)=>{

    const restaurantId = Number.parseInt(req.params.id);  //แปลงให้เป็นจำนวนเต็ม

    Restaurant.getById(restaurantId, (err, data)=>{

        if(err){

            if(err.kind === 'not\_found'){

                res.status(404).send({

                    message: `ไม่พบร้านอาหารที่มีรหัสนี้ ${restaurantId}`,

                });

            }

            else{

            res.status(500).send({

                message: "เกิดข้อผิดพลาดในการเรียกข้อมูลด้วยรหัสนี้ " + restaurantId,

            })

            }

        }

        else

        {

            res.send(data);

        }

    });

});

//Get All restaurant

// http://localhost:5000/apis/restaurants

router.get('/restaurants',(req,res) => {

    Restaurant.getAll((err,data) => {

        if(err){

            res.status(500).send({

                message: err.message || "Come err occurred while retrieving restaurants",

            });

        }

        else{

            res.send(data);

        }

    });

});

//Updata restaurant Data

// http://localhost:5000/apis/restaurants/1

router.put("/restaurants/:id",(req, res)=>{

    const restaurantId = Number.parseInt(req.params.id);  //แปลงให้เป็นจำนวนเต็ม

    if(req.body.constructor === Object && Object.keys(req.body).length === 0){//เช็คค่าว่าง

        res.status(400).send({

            message : "เนื้อหาไม่สามารถเว้นว่างได้"

        });

    }

    Restaurant.updateById(restaurantId, new Restaurant(req.body), (err,data)=>{

        if(err){

            if(err.kind === "not\_found"){

                res.status(404).send({

                    message: `ไม่พบร้านอาหารที่มีรหัสนี้ ${restaurantId}`,

                });

            }

            else{

            res.status(500).send({

                message: "เกิดข้อผิดพลาดในการอัปเดตข้อมูลร้านอาหารด้วยรหัสนี้ " + restaurantId,

            });

            }

        }

        else

        {

            res.send(data);

        }

    });

});

//Delete restaurant by Id

// http://localhost:5000/apis/restaurants/1

router.delete("/restaurants/:id", (req,res)=>{

    const restaurantId = Number.parseInt(req.params.id);  //แปลงให้เป็นจำนวนเต็ม

    Restaurant.removeById(restaurantId,(err,data)=>{

        if(err){

            if(err.kind === "not\_found"){

                res.status(404).send({

                    message: `ไม่พบร้านอาหารที่มีรหัสนี้ ${restaurantId}`,

                });

            }

            else{

            res.status(500).send({

                message: "เกิดข้อผิดพลาดในการลบข้อมูลร้านอาหารด้วยรหัสนี้ " + restaurantId,

            });

            }

        }

        else

        {

            res.send({message: "ลบร้านอาหารเรียบร้อยแล้ว"});

        }

    })

})

module.exports = router;

routes -> user.router.js

const express = require('express');

const router = express.Router();

const user = require("../models/user.models");

//Method insert Data

//http://localhost:5000/apis/user

router.post("/user",(req,res) =>{

    //สร้างผู้ใช้

    const newUser = new user({

        profile:req.body.profile,

        firstName:req.body.firstName,

        lastName:req.body.lastName,

        pocition:req.body.pocition,

        gender:req.body.gender,

        tel:req.body.tel,

        address:req.body.address,

    });

    //บันทึกลงฐานข้อมูล

    user.create(newUser,(err, data)=>{

        if(err)

            res.status(500).send({

                message: err.message || "เกิดข้อผิดพลาดขณะสร้างผู้ใช้งาน"

            })

        else res.send(data);

    })

});

//แสดงตามid

// http://localhost:5000/apis/user/2

router.get('/user/:id', (req,res)=>{

    const userId = Number.parseInt(req.params.id);  //แปลงให้เป็นจำนวนเต็ม

    user.getById(userId, (err, data)=>{

        if(err){

            if(err.kind === 'not\_found'){

                res.status(404).send({

                    message: `ไม่พบผู้ใช้งานนี้ ${userId}`,

                });

            }

            else{

            res.status(500).send({

                message: "เกิดข้อผิดพลาดในการเรียกข้อมูลด้วยรหัสนี้ " + userId,

            })

            }

        }

        else

        {

            res.send(data);

        }

    });

});

//Get All restaurant

// http://localhost:5000/apis/user

router.get('/user',(req,res) => {

    user.getAll((err,data) => {

        if(err){

            res.status(500).send({

                message: err.message || "เกิดข้อผิดพลาดขณะดึงข้อมูลผู้ใช้งาน",

            });

        }

        else{

            res.send(data);

        }

    });

});

//Updata restaurant Data

// http://localhost:5000/apis/user/1

router.put("/user/:id",(req, res)=>{

    const userId = Number.parseInt(req.params.id);  //แปลงให้เป็นจำนวนเต็ม

    if(req.body.constructor === Object && Object.keys(req.body).length === 0){//เช็คค่าว่าง

        res.status(400).send({

            message : "เนื้อหาไม่สามารถเว้นว่างได้"

        });

    }

    user.updateById(userId, new user(req.body), (err,data)=>{

        if(err){

            if(err.kind === "not\_found"){

                res.status(404).send({

                    message: `ไม่พบผู้ใช้งานที่มีรหัสนี้ ${userId}`,

                });

            }

            else{

            res.status(500).send({

                message: "เกิดข้อผิดพลาดในการอัปเดตข้อมูลผู้ใช้งานด้วยรหัสนี้ " + userId,

            });

            }

        }

        else

        {

            res.send(data);

        }

    });

});

//Delete user by Id

// http://localhost:5000/apis/user/1

router.delete("/user/:id", (req,res)=>{

    const userId = Number.parseInt(req.params.id);  //แปลงให้เป็นจำนวนเต็ม

    user.removeById(userId,(err,data)=>{

        if(err){

            if(err.kind === "not\_found"){

                res.status(404).send({

                    message: `ไม่พบผู้ใช้งานที่มีรหัสนี้ ${userId}`,

                });

            }

            else{

            res.status(500).send({

                message: "เกิดข้อผิดพลาดในการลบข้อมูลผู้ใช้งานด้วยรหัสนี้ " + userId,

            });

            }

        }

        else

        {

            res.send({message: "ลบผู้ใช้งานเรียบร้อยแล้ว"});

        }

    })

})

module.exports = router;

server.js

const express = require('express');

const restaurantRouter = require('./routes/restaurant.router');

const userRouter = require('./routes/user.routes');

const cors = require('cors')

//Create Server

const app = express();

//Use Middleware

app.use(cors());

app.use(express.json());

app.use(express.urlencoded({extended: false}));

//Router

app.get('/',(req,res)=>{

    res.send('<h1>This is Restaurant API</h1>'); //โชว์ข้อมูลที่กำหนด

});

// เรียกใช้ router

//restaurant

app.use("/apis",restaurantRouter);

//user

app.use("/apis",userRouter);

//Run server

app.listen(5000, ()=>{

    console.log('เซิร์ฟเวอร์กำลังฟังพอร์ต 5000')

})

วิธีรัน

Frontend กด live server ได้เลย

Resful\_SQL\_API new Terminal ใช้คำสั่ง nodemon server.js