โครงงาน การ์ดจอ (ระยะที่ 2 กลุ่ม 4)

รายวิชา ITDS241 Web Technologies and Applications

รายชื่อสมาชิก

นายณัฐพงศ์ หลวงพิบูลย์	รหัสนักศึกษา 6487027
นางสาวสุพิชญา อเนกศุภพล	รหัสนักศึกษา 6587041
นายฤทธิชล พลราช	รหัสนักศึกษา 6587062
นางสาวจิรัชญา ราชพลแสน	รหัสนักศึกษา 6587074

ชื่ออาจารย์ผู้สอน

อาจารย์ ดร จิดาภา ไกรสังข์

อาจารย์ ดร วุฒิชาติ แสวงผล

มหาวิทยาลัยมหิดล คณะเทคโนโลยีสารสนเทศและการสื่อสาร

ภาคการศึกษาที่ 1 / 2566

คำนำ

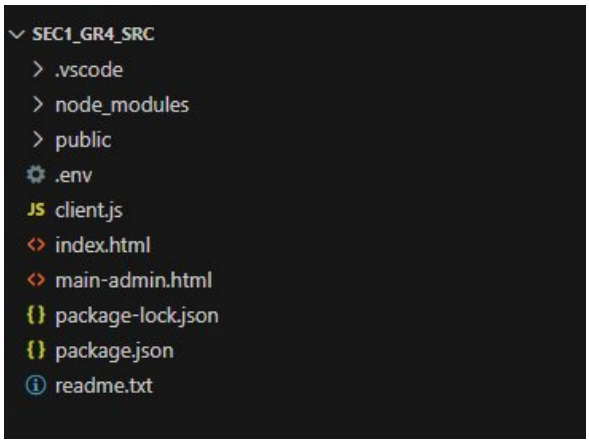
รายงานนี้เป็นส่วนหนึ่งของรายวิชา ITDS241 Web Technologies and Applications จัดทำขึ้นเพื่อนำเสนอเกี่ยวกับเพื่อให้การศึกษาและการเรียนรู้เกี่ยวกับเทคโนโลยีเว็บและการประยุกต์ใช้งานต่าง ๆ ที่เกี่ยวข้อง

รายงานนี้ถูกสร้างขึ้นเพื่อแสดงถึงการพัฒนาและสร้างเว็บไซต์ที่สามารถนำมาประยุกต์ใช้ในการพัฒนาเว็บไซต์ที่ทันสมัยและตอบสนองความต้องการของผู้ใช้หวังว่าโครงงานนี้จะเป็นความรู้ที่มีประโยชน์เกี่ยวกับการพัฒนาและสร้างเว็บไซต์ในอนาคต

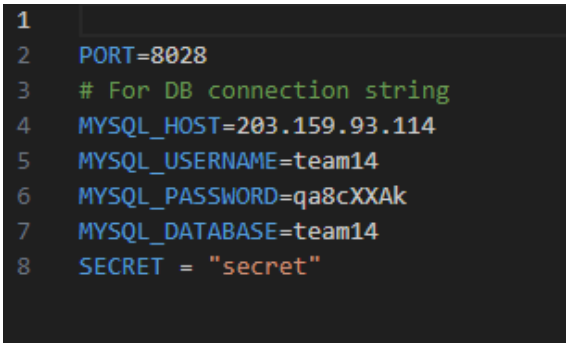
สารบัญ

ข้อมูลองค์รวมของโครงงาน

เว็บไซต์ของพวกเราเป็นเว็บไซต์ที่ไว้สำหรับการซื้อ Graphic card รุ่นต่างๆหลากหลายรุ่นเช่น NVIDIA หรือ AMD เป็นต้น ถ้าต้องการเลือกซื้อสินค้าสามารถค้นหารุ่นหรือแบรนด์ที่ต้องการค้นหาได้และเลือกสินค้าที่ต้องการได้ในส่วนของ search bar ในส่วนของหน้า Home นั้นจะสามารถกด Login ซึ่งสามารถจะ Login ผ่านระบบ Gmail ได้ ในส่วนระบบของ Admin จะมี Insert , Edite , Delete และ Information



ในส่วนของ web service จะแยกเป็น folder ที่ชื่อว่า server ซึ่งมีไฟล์ connectdb.js เป็นตัวเชื่อมกับฐานข้อมูลและมีไฟล์ .env ที่เชื่อมต่อกับฐานข้อมูล โดยที่ฐานข้อมูลอยู่ใน connection และมีด้านในจะมีเก็บข้อมูลตามภาพด้านล่าง



(.env)

Module ทั้งหมดที่ได้ทำการติดตั้ง

```
"dependencies": {
    "axios": "^1.6.1",
    "cors": "^2.8.5",
    "dotenv": "^16.3.1",
    "express": "^4.18.2",
    "jsonwebtoken": "^9.0.2",
    "mysql2": "^3.6.3",
    "node-fetch": "^3.3.2",
    "nodemon": "^3.0.1"
  }
}
```

(Package.json)

นอกเหนือจาก folder ที่ชื่อว่า server จะเป็นในส่วนของ client ในไฟล์ index.html จะมีการใช้แท็ก script อยู่ภายในไฟล์ เพื่อทำงาน log in และเรียกดูสินค้าทั้งหมด

(index.html)

```
<!-- product section Nvdia-->

<section class="product_section">
  <div class="container">
    <h2>
      <div class="product_heading">
    </h2><input style=" padding: 10px 15px;width: 100%;
        background-color: #383838;
        color: #ffffff;
        border-radius: 5px;
        font-size: 24px;
        margin-top: 10px;" type="button" value="All Product" id="product_heading">
    </h2>
  </div>

  <div class="product_container" id="nvdia_all_prod"></div>
  <!-- --- -->


  <!-- end product section -->
</section>
<script>
  const rootURL = "http://localhost:8028/";

  let login_submit = document.querySelector("#loginbutton");
  let user_login_input = document.querySelector("#txtemail");
  let pass_login_input = document.querySelector("#txtpasswd");
  let user_login,
    password_login,
    token = "";

  async function callLoginpage(url, method, atoken = "", sentData = {}) {
    let data;
    if (method === "selectall") {
      let response = await fetch(url, {
        method: "GET",
        headers: {
          Authorization: "Bearer " + atoken,
        },
      });
      data = await response.json();
    } else if (method === "login") {
      let aMethod;
```

ไฟล์ callProduct.js จะทำงานเมื่อมีการกดปุ่ม เพิ่ม ลบ อัพเดท สินค้า ซึ่งจะมี add-product.html delete-product.html และ edit-product.html ที่คอยเรียกใช้งาน

```
let apiUrl = "http://localhost:8028/";

console.log(apiUrl)

// ดึงtag ที่อยู่ในตาราง
let serach_product = document.querySelector(".search_edit_prod");
/* page add-product*/
let code_pro = document.querySelector("#code-product");
let name_pro = document.querySelector("#name-product");
let picture_pro = document.querySelector("#picture-product");
let brand_pro = document.querySelector("#brand-product");
let series_pro = document.querySelector("#series-product");
let suppier_pro = document.querySelector("#suppier-product");
let detail_pro = document.querySelector("#detail-product");
let price_pro = document.querySelector("#price-product");
// ปุ่ม
let add_product = document.querySelector("#add-product-save");

/* page delete-product*/
let delete_search = document.querySelector("#delete-search");
// ปุ่ม
let delete_search_product = document.querySelector("#find_del-product-button");
// let delete_product_button = document.querySelector("#delete_product_button");


/* edit delete-product*/
let edit_search = document.querySelector("#edit-search");
// ปุ่ม
let edit_product = document.querySelector("#edit-serch-submit");
let edit_save_product = document.querySelector("#edit-product-save");




async function callProduct(url, method, sentData = {}) {
  console.log(url)
  let data;
  if (method == "selectall") {
    let response = await fetch(url, {
      method: "GET",
    });
    data = await response.json();
  } else if (method == "select") {
    let response = await fetch(url, {
      method: "GET",
    });
    data = await response.json();
  } else if (method == "insert" || method == "update" || method == "delete") {
```

(callProduct.js)

ไฟล์ callAdmin.js จะทำงานเมื่อมีการกดปุ่ม เพิ่ม ลบ อัพเดท เกี่ยวกับ admin ซึ่งจะมี admin.html add admin.html และ update admin.html ที่คอยเรียกใช้งาน

```
let apiUrl = "http://localhost:8028/";

console.log(apiUrl)
// ดึงtag ที่อยู่ในตาราง

let search_admin = document.querySelector("#search");
/* page admin_info*/
let search_admin_submit = document.querySelector("#searchButton");
let select_all_admin = document.querySelector("#select_admin");


/* page add-admin*/
let add_admin_submit = document.querySelector("#add_admin_submit");
let fname_admin_input = document.querySelector("#txtfirstname");
let lname_admin_input = document.querySelector("#txtlastname");
let identity_admin_input = document.querySelector("#txtidentificationNo")
let address_admin_input = document.querySelector("#txtaddress");
let bday_admin_input = document.querySelector("#cldBD");
let id_admin_input = document.querySelector("#txtAdminID");
let tel_admin_input = document.querySelector("#txtPhonenumber");
let email_admin_input = document.querySelector("#txtemail");
let other_admin_input = document.querySelector("#txtothers");

/* page update-admin*/
let update_admin_submit = document.querySelector("#update_admin_submit");

/* page update-admin*/
let delete_admin_submit = document.querySelector("#delete_admin");



/* fetch สำหรับ select and select all*/
async function callAdmin(url, method) {
  console.log(url)
  let data;
  if (method == "selectall") {
    let response = await fetch(url, {
      method: "GET",
    });
    data = await response.json();
  } else if (method == "select") {
    let response = await fetch(url, {
      method: "GET",
    });
    data = await response.json();
    console.log(data)
  }
```

(callAdmin.js)

ไฟล์ product.html จะมี tag script ด้านในที่ทำการค้นหาข้อมูล

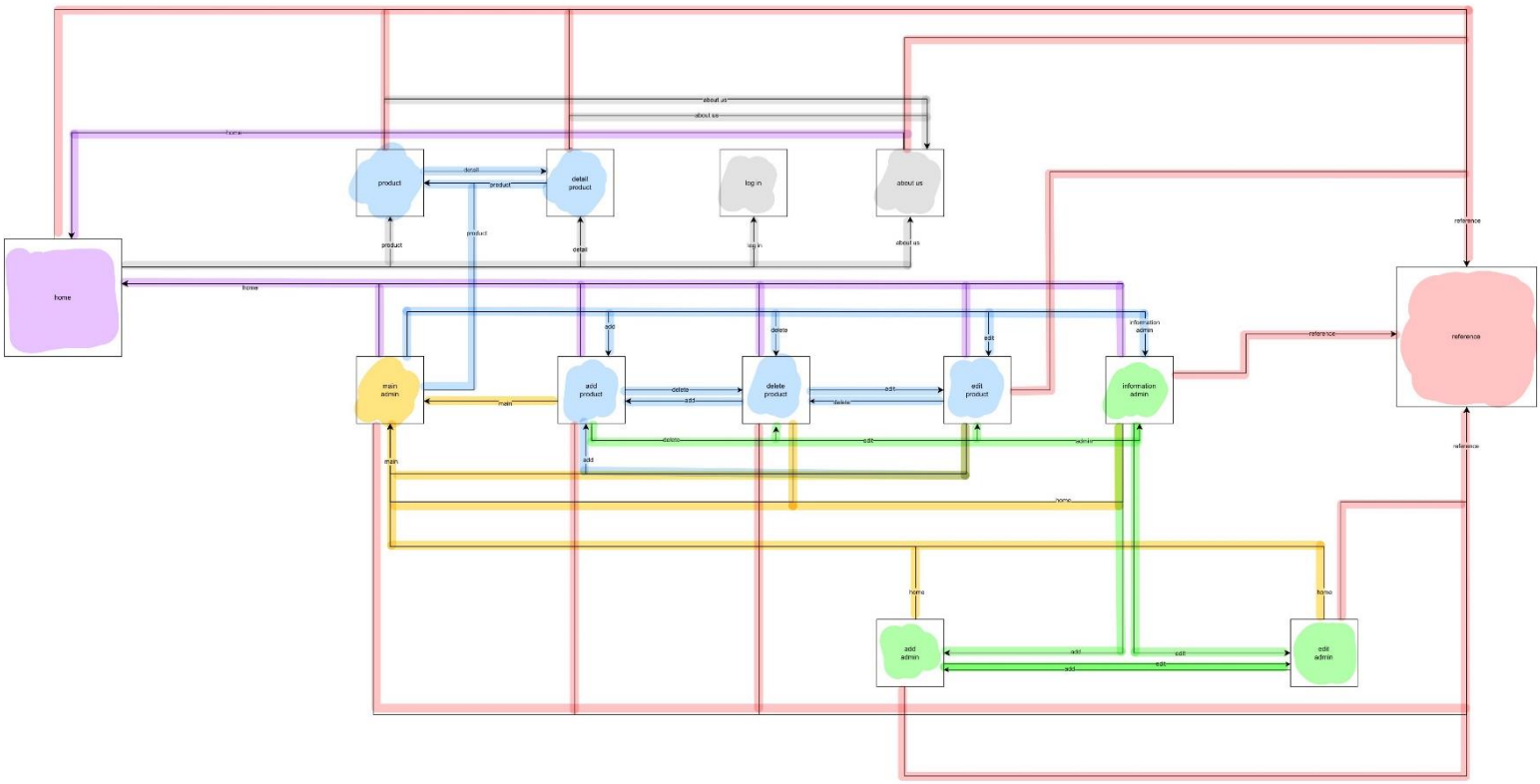
```
        <div class="input-group">
          <input id="search_product_series"type="search" class="form-control rounded" placeholder="Search by Series ex. RTX4090 RTX4070" />
        </div>
        <button type="button" class="form-control rounded" id="search_product_all">All Product</button>
        <button type="button" class="form-control rounded" id="search_product_by_type">Search</button>


        <div id="output"><p></p></div>
      </div>


      <div class="product_container" id="product_container">

    </div>
  </section>
  <!-- end product section -->

  <script>
    let rootUrl = "http://localhost:8028/product_info/"
    let serach_product_by_type = document.querySelector("#search_product_by_type");
    let serach_product_all = document.querySelector("#search_product_all");


    let serach_product_manufac = document.querySelector("#search_product_by_manufacturer");
    let serach_product_brand = document.querySelector("#search_product_by_brand");
    let serach_product_series = document.querySelector("#search_product_by_series");

    let input_product_manufac = document.querySelector("#search_product_manufac");
    let input_product_brand = document.querySelector("#search_product_brand");
    let input_product_series = document.querySelector("#search_product_series");

    async function callProduct(url, method){
      let response = await fetch(url, {
      method: method,
      headers: {
        Accept: "application/json",
        "Content-Type": "application/json",
      },
      body: JSON.stringify(),
      });
      data = await response.json();
      console.log(data)

      return data
    }
```

(product.html)

```
if (search_product_by_type) {
search_product_by_type.addEventListener("click", () => {
  input_product_manufac = input_product_manufac.value;
  input_product_brand = input_product_brand.value;
  input_product_series = input_product_series.value;
  console.log(input_product_manufac, input_product_brand ,input_product_series);

    if(input_product_manufac && !input_product_brand && !input_product_series){
      callProduct(rootUrl + input_product_manufac, "get").then((data) => {
      console.log(data);
      if (data.data) {
        alert("Select by Manufacturer");

        if (Array.isArray(data.data)) {
          document.getElementById("product_container").innerHTML = generateHTML(data);
          input_product_manufac.value = "";
        }
      } else {
        alert("Not Found");
      }
    });
  } else if(!input_product_manufac && input_product_brand && !input_product_series){
    callProduct(rootUrl + input_product_brand, "get").then((data) => {
      console.log(data);
      if (data.data) {
        alert("Select by Brand");

        if (Array.isArray(data.data)) {
          document.getElementById("product_container").innerHTML = generateHTML(data);
          input_product_manufac.value = "";
        }
      } else {
        alert("Not Found");
      }
    });
  }else if(!input_product_manufac && !input_product_brand && input_product_series){
    callProduct(rootUrl + input_product_series, "get").then((data) => {
      console.log(data);
      if (data.data) {
        alert("Select by Series");

        if (Array.isArray(data.data)) {
          document.getElementById("product_container").innerHTML = generateHTML(data);
          input_product_manufac.value = "";
        }
      } else {
        alert("Not Found");
```

(product.html)

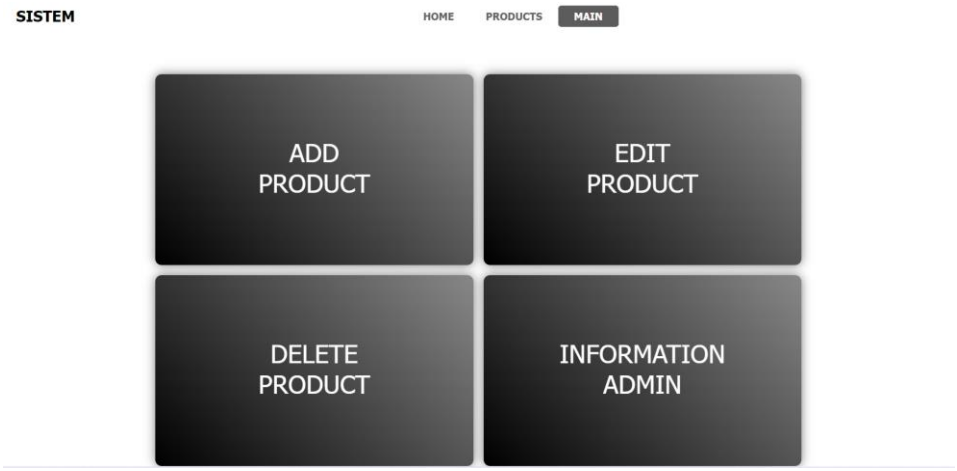
แผนผังหน้าต่างๆ ของเว็บแอพพลิเคชั่น



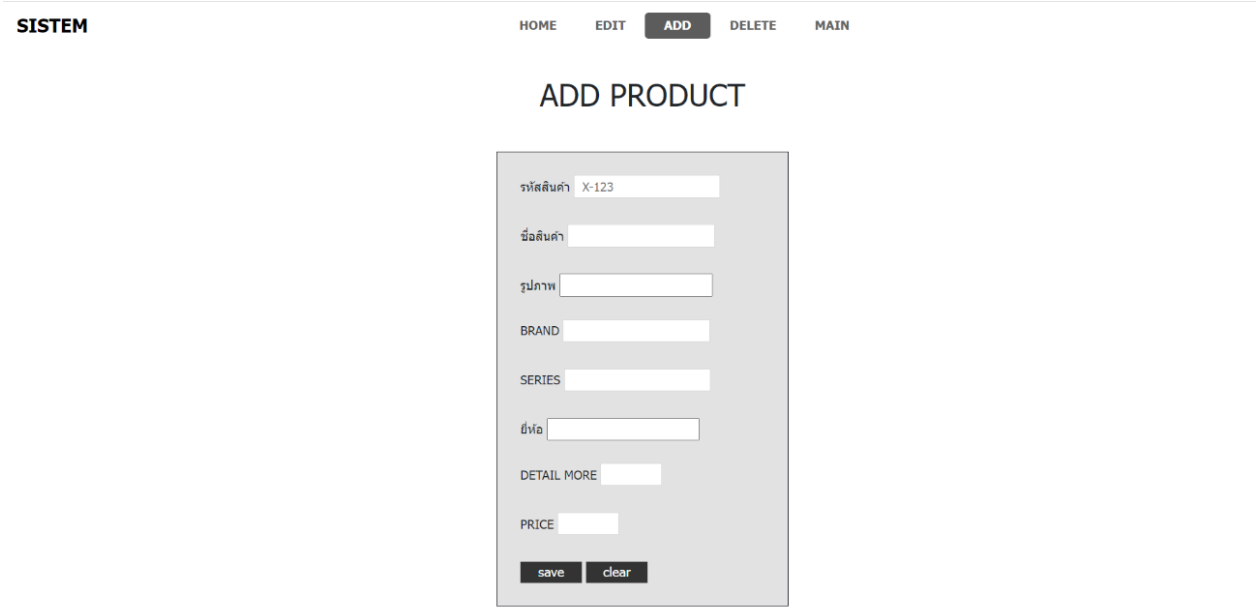
รายละเอียดหน้าเว็บต่างๆ ของผู้ดูแล

เมื่อเข้า http://localhost:8027/admin จะเข้าสู่หน้าที่สามารถจัดการรายละเอียดได้

หน้า Main เป็นหน้าหลักของระบบผู้ดูแลที่ใช้สำหรับให้ผู้ดูแลระบบเลือกเข้าสู่หน้าการเพิ่มสินค้า การแก้ไขข้อมูลสินค้า การลบข้อมูลสินค้า หรือเข้าไปจัดการเกี่ยวกับข้อมูลของแอดมินใน หน้า Information admin



หน้า Add Product เป็นหน้าที่ให้ผู้ดูแลระบบมาเพิ่มข้อมูลสินค้าที่มีรายละเอียด เกี่ยวกับ รหัสสินค้า ชื่อสินค้า รูปภาพ แบรนด์ ซีรีย์ ยี่ห้อ รายละเอียดเพิ่มเติม ราคา เมื่อเพิ่มสำเร็จจะมีข้อความแจ้งเตือนว่า New product has been created successfully เป็นการเพิ่มสินค้าเสร็จสมบูรณ์

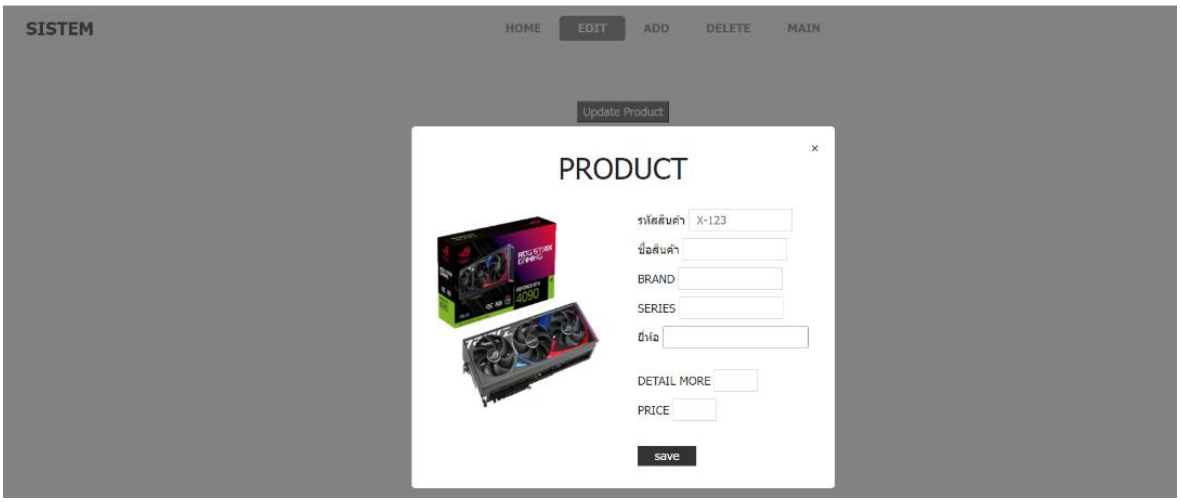


เมื่อทำการกดปุ่ม save จะเริ่มทำการดึงข้อมูลจาก input แต่ละตัว และมีการเปลี่ยนค่าที่รับ url รูปภาพ จากนั้นเก็บไว้ในตัวแปร product_data เพื่อนำข้อมูลเก็บในตาราง database

```
if (add_product) {
  add_product.addEventListener("click", () => {
    console.log("Pass")
    code_product = code_pro.value;
    name_product = name_pro.value;
    picture_product = picture_pro.value;
    brand_product = brand_pro.value;
    series_product = series_pro.value;
    console.log(series_product)
    suppier_product = suppier_pro.value;
    detail_product = detail_pro.value;
    price_product = price_pro.value;
    console.log("Add");

    // แปลง url ที่คัดลอกจาก google drive
    let urlGoogle_drive_Picture = picture_product;
    let idProduct = urlGoogle_drive_Picture.match(/\/d\/(.+?)\//)[1];
    let urlPicture = `https://drive.google.com/uc?export=view&id=${idProduct}`;

    let product_data = {
      id_product: code_product,
      name_product: name_product,
      pic_product: urlPicture,
      brand_product: brand_product,
      series_product: series_product,
      manufacturer_product: suppier_product,
      detail_product: detail_product,
      price_product: price_product,
    }
    callProduct(apiUrl + "product_info", "insert", product_data).then((data) => {
      console.log(data)
      if (data.data > 0) {
        alert(data.message);
        clearInput();
      }else{
        alert("Please provide information")
      }
    });
  });
}
```

(callProduct.js)

หน้า Update Product เป็นหน้าสำหรับการแก้ไขข้อมูลเมื่อมีการเปลี่ยนแปลงข้อมูลเกี่ยวกับ รหัสสินค้า ชื่อสินค้า รูปภาพ แบรนด์ ซีรีย์ ยี่ห้อ รายละเอียดเพิ่มเติม ราคา เมื่อแก้ไขสำเร็จจะมีข้อความแจ้งเตือนว่า Product has been updated successfully

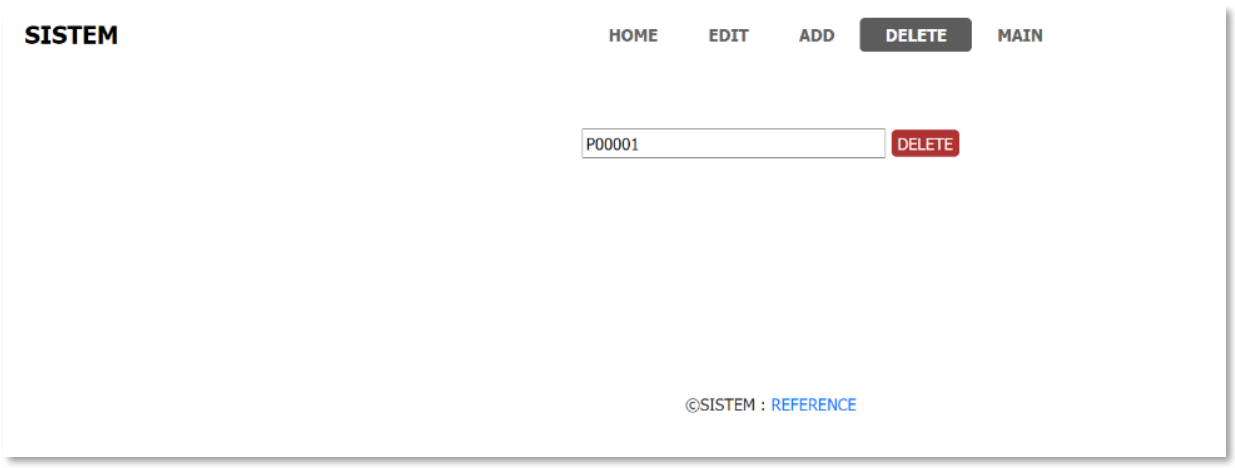


เมื่อกดปุ่ม save จะทำการดึงค่าจากข้อมูล input

```
if (edit_save_product) {
  edit_save_product.addEventListener("click", () => {
    console.log("Pass")
    code_product = code_pro.value;
    name_product = name_pro.value;
    picture_product = picture_pro.value;
    brand_product = brand_pro.value;
    series_product = series_pro.value;
    suppier_product = suppier_pro.value;
    detail_product = detail_pro.value;
    price_product = price_pro.value;
    console.log("Add product");
    let urlGoogle_drive_Picture = picture_product;
    let idProduct = urlGoogle_drive_Picture.match(/\/d\/(.+?)\//)[1];
    let urlPicture = `https://drive.google.com/uc?export=view&id=${idProduct}`;

    let product_data = {
      id_product: code_product,
      name_product: name_product,
      pic_product: urlPicture,
      brand_product: brand_product,
      series_product: series_product,
      manufacturer_product: suppier_product,
      detail_product: detail_product,
      price_product: price_product,
    }
    callProduct(apiUrl + "product_info", "update", product_data).then((data) => {
      console.log(data)
      if (data.data > 0) {
        alert(data.message);
        console.log(code_pro)
        code_pro.value = "";
        name_pro.value = "";
        brand_pro.value = "";
        series_pro.value = "";
        suppier_pro.value = "";
        detail_pro.value = "";
        price_pro.value = "";
        window.location.href  = '/adminpage/add-del-edit-product/edit-product.html#';
      }
    });
  });
}
```

(callProduct.js)

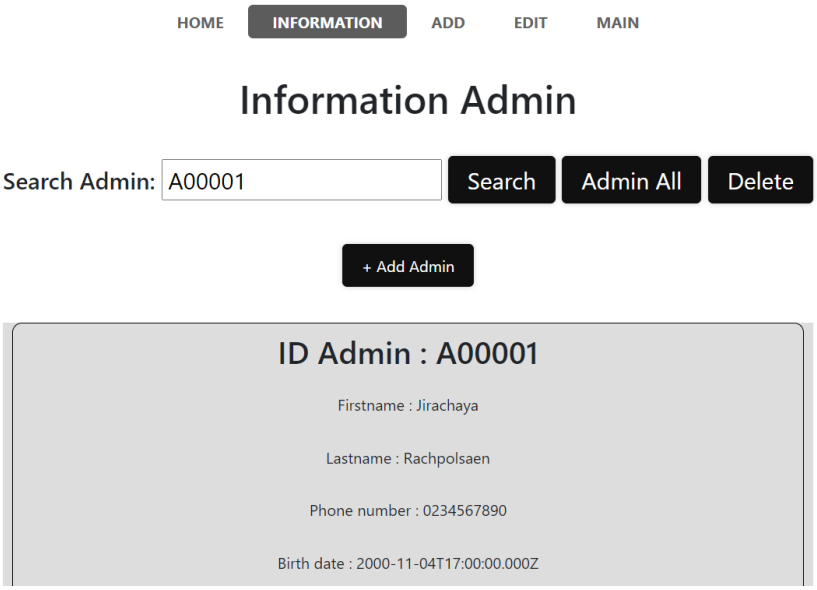
หน้า Delete Product เป็นการลบข้อมูลของสินค้าได้ตาม Id ที่กำหนด ของสินค้าที่เพิ่มไปในแต่ละครั้ง



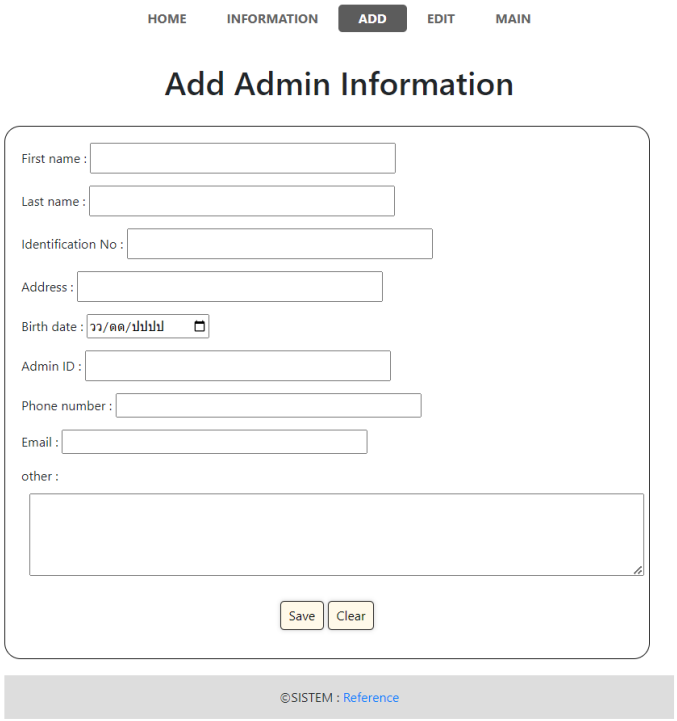
เมื่อมีการใส่ ID สินค้าที่ต้องการลบ จะให้ทำการนำ ID ที่ได้ไปดึงข้อมูล

```
if(delete_search_product) {
  delete_search_product.addEventListener("click", () =>{
  id_pro = delete_search.value;
  let product_data = {
    id_product: id_pro,};
  console.log(product_data)
  callProduct(apiUrl+ "product_info/"+ id_pro, "select").then((data)=>{

    console.log(data)
    console.log(data.id_product)
    if(data.data){
      alert(data.data.id_product);
      code_pro = data.data.id_product;
      console.log(code_pro)
      name_pro = data.data.name_product;
      brand_pro = data.data.brand_product;
      series_pro = data.data.series_product;
      manufacturer_pro = data.data.manufacturer_product;
      console.log(name_pro+"name")
      let output;
      output = "<h1>รายการ</h1> <br>";
      output += "<p> ";
      output += "</p> ";
      output += " ";
      output += "<table class='spaced-table'>";
      output += "<table '>";
      output += "<thead>";
      output += "<tr>";
      output +="<th scope='col'>รหัสสินค้า</th><th scope='col'>ชื่อสินค้า</th><th scope='col'>แบรนด์</th><th scope='col'>ยี่ห้อ</th><th scope='col'>ซีรีย์</th>";
      output += "</tr> ";
      output += "</thead>";
      output += "<tbody>"

        output += "<tr>";
        output += "<td>" + code_pro + "</td>";
        output += "<td>" + name_pro + "</td>";
        output += "<td>" + brand_pro + "</td>";
        output += "<td>" + series_pro  + "</td>";
        output += "<td>" + manufacturer_pro + "</td>";

        output += "</tr>";
      output += "</tbody>";
      output += "</table>";
      $("#edit-prod-table").html(output);
          console.log("deleteee")
          callProduct(apiUrl+ "product_info", "delete",product_data).then((data)=> {
```

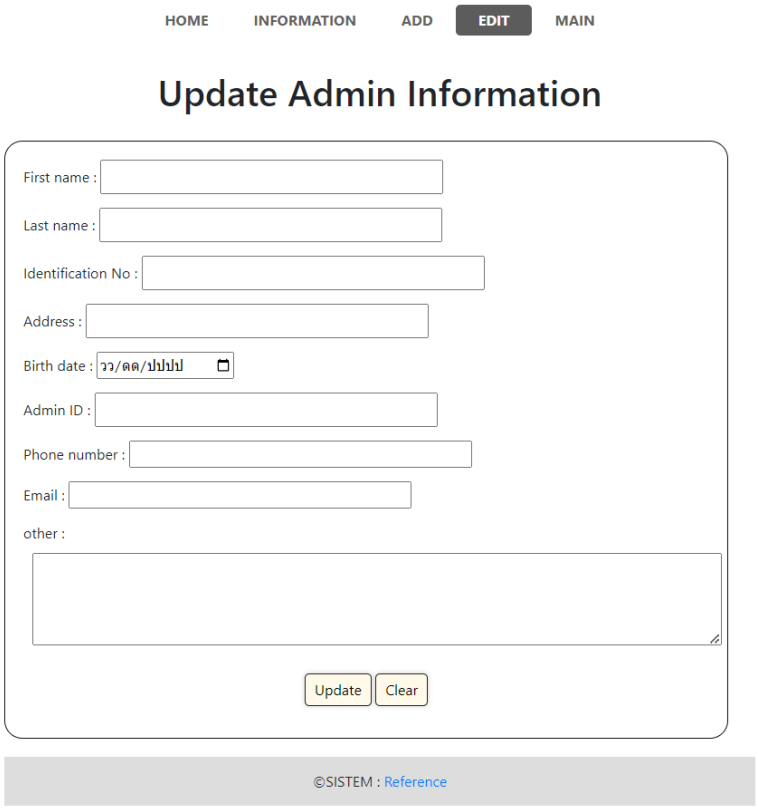
หน้า Information Admin เป็นหน้าที่ให้ผู้ดูแลระบบสามารถค้นหาข้อมูล admin ได้แบบค้นหาตาม id admin หรือ กด admin all เพิ่มแสดงข้อมูลแอดมินทั้งหมด และมีปุ่ม add admin สำหรับเชื่อมไปยังหน้า add admin เพื่อรับข้อมูล admin ที่เพิ่มเข้ามาใหม่



หน้า Add Admin Information เป็นหน้าที่ให้ผู้ดูแลระบบสามารถเพิ่มข้อมูลของ admin คนใหม่เข้าไปยังฐานข้อมูลและแสดงข้อมูล admin ในหน้า Information Admin ได้



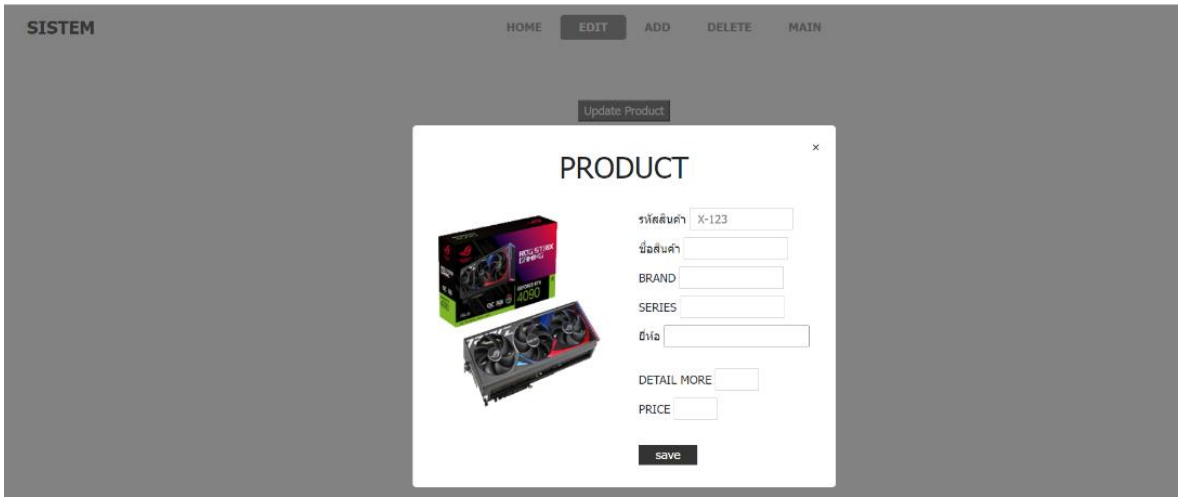
หน้า Update Admin เป็นการที่ผู้ดูแลระบบจะทำการแก้ไขข้อมูลของ Admin หรืออัพเดทเปลี่ยนแปลงตามความต้องการได้



รายละเอียดหน้าเว็บเซอร์วิส

เมื่อเข้า http://localhost:8027/home จะเข้าสู่หน้า home

```
})
router.get('/index.html', (req, res) => {
    console.log('index page')
    res.redirect("/home");
})
router.get('/home', (req, res) => {
    console.log('Request at' + req.url)
    res.sendFile(path.join(`${__dirname}/index.html`))
})
```



(client.js)

```
login_submit.addEventListener("click", () => {
  console.log("login submit")
  let user_login_data = {
    login_sistem: {
      username_sistem: user_login_input.value,
      password_sistem: pass_login_input.value,
    },
  };

  console.log(user_login_data)
  callLoginpage(rootURL + "login_info", "login", token, user_login_data).then((data) => {
```

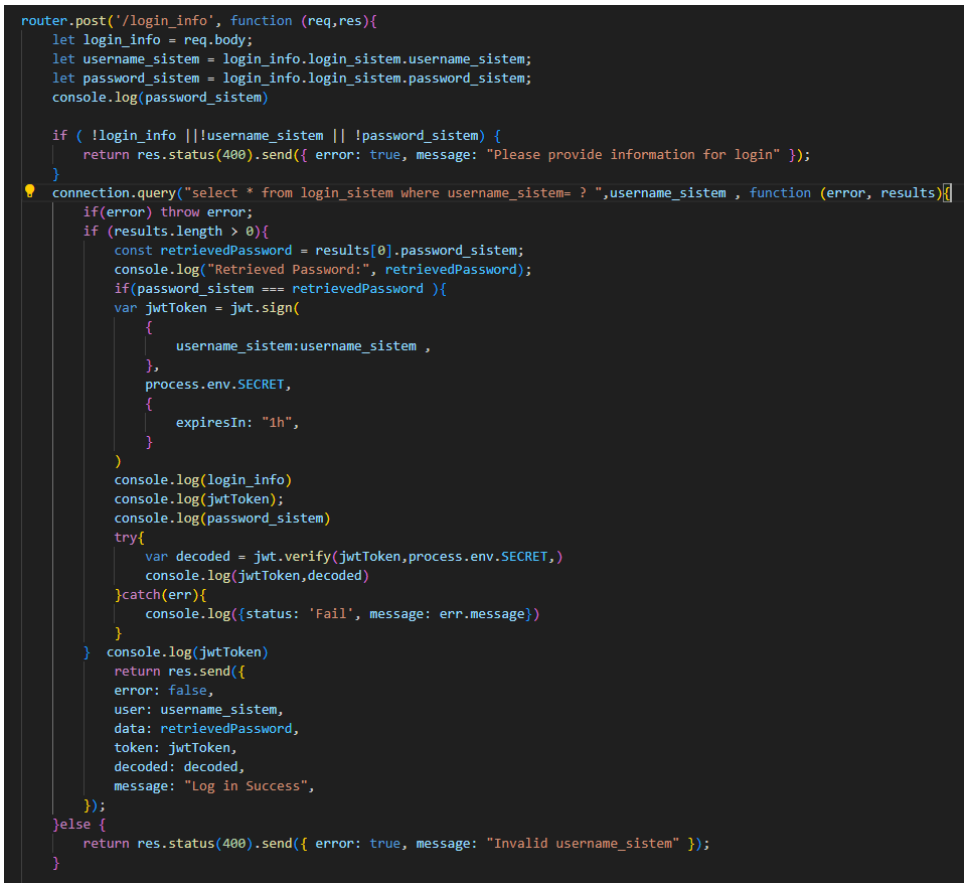
(index.html)

เมื่อมีการเรียก function callLoninpage จะมีการนำค่า method ไปเช็คว่าตรงกับ method ใด เพื่อนำไป fetch ดึงข้อมูลจากฐานข้อมูล

```
let user_login,
  password_login,
  token = "";

async function callLoginpage(url, method, atoken = "", sentData = {}) {
  let data;
  if (method === "selectall") {
    let response = await fetch(url, {
      method: "GET",
      headers: {
        Authorization: "Bearer " + atoken,
      },
    });
    data = await response.json();
  } else if (method === "login") {
    let aMethod;
    if (method === "insert" || method === "login") {
      aMethod = "POST";
    } else if (method === "update") {
      aMethod = "PUT";
    } else if (method === "delete") {
      aMethod = "DELETE";
    }
    let response = await fetch(url, {
      method: aMethod,
      headers: {
        Accept: "application/json",
        "Content-Type": "application/json",
        Authorization: "Bearer " + atoken,
      },
      body: JSON.stringify(sentData),
    });
    data = await response.json();
  }

  return data;
}
```

(index.html)

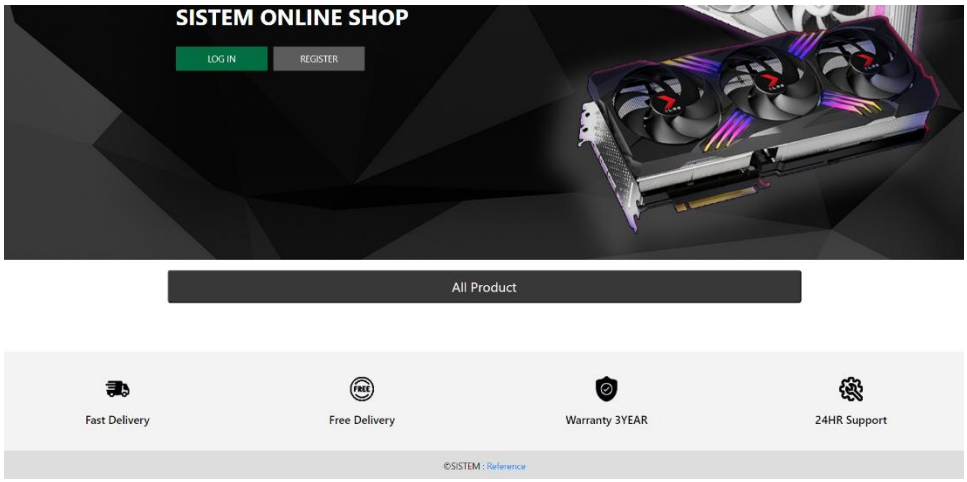
ฐานข้อมูล โดยถ้าค่าที่ได้มีความยาวมากกว่า 0 ให้เช็คว่าค่าที่รับมามีค่าเท่ากันในฐานข้อมูลหรือไม่ ถ้ามีจะให้ทำการสร้างตัวแปร jwt เพื่อรวมข้อมูล username และรหัสที่เป็นคีย์ลับ (การสร้าง token)



```
router.post('/login_info', function (req,res){
    let login_info = req.body;
    let username_sistem = login_info.login_sistem.username_sistem;
    let password_sistem = login_info.login_sistem.password_sistem;
    console.log(password_sistem)

    if ( !login_info ||!username_sistem || !password_sistem) {
        return res.status(400).send({ error: true, message: "Please provide information for login" });
    }
    connection.query("select * from login_sistem where username_sistem= ? ",username_sistem , function (error, results){
        if(error) throw error;
        if (results.length > 0){
            const retrievedPassword = results[0].password_sistem;
            console.log("Retrieved Password:", retrievedPassword);
            if(password_sistem === retrievedPassword ){
            var jwtToken = jwt.sign(
                {
                    username_sistem:username_sistem ,
                },
                process.env.SECRET,
                {
                    expiresIn: "1h",
                }
            )
            console.log(login_info)
            console.log(jwtToken);
            console.log(password_sistem)
            try{
                var decoded = jwt.verify(jwtToken,process.env.SECRET,)
                console.log(jwtToken,decoded)
            }catch(err){
                console.log({status: 'Fail', message: err.message})
            }
        }  console.log(jwtToken)
            return res.send({
            error: false,
            user: username_sistem,
            data: retrievedPassword,
            token: jwtToken,
            decoded: decoded,
            message: "Log in Success",
        });
    }else {
        return res.status(400).send({ error: true, message: "Invalid username_sistem" });
    }
```

(connectdb.js)

เมื่อเลื่อนหน้า home มาที่ด้านล่าง จะมีแถบที่ชื่อว่า All Product เมื่อทำการ click ที่แถบนี้ จะสามารถดู product ทั้งหมดได้

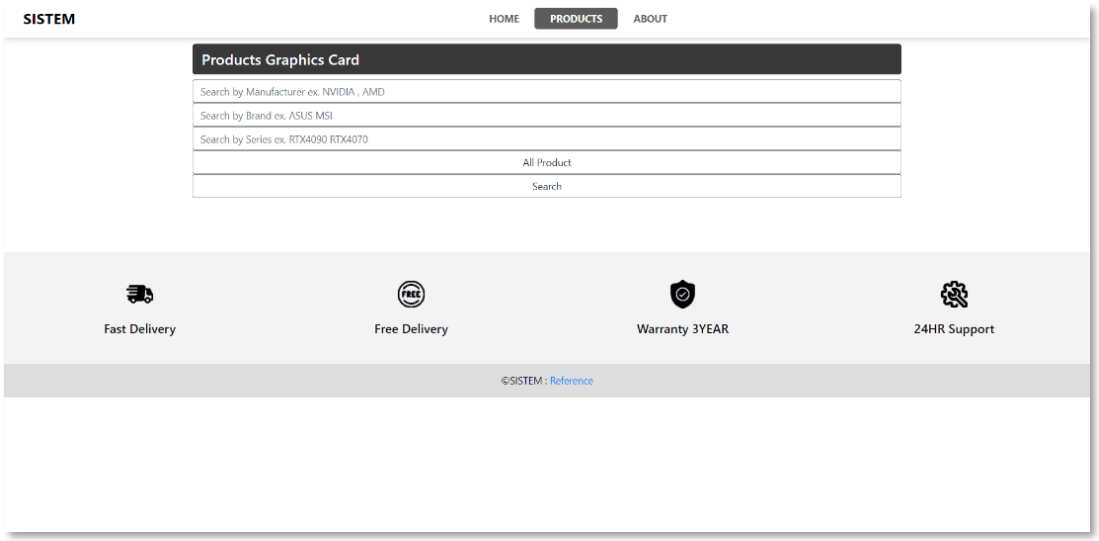


เมื่อมีการ click ที่ All Product จะทำการ fetch ซึ่งเป็น function เดียวกับ fetch ใน log in เมื่อได้ข้อมูลจาก database มาแล้ว จะทำการนำมาแสดงผลบนหน้า web

```
let nvdia_all_product = document.querySelector("#product_heading");
if (nvdia_all_product) {
  nvdia_all_product.addEventListener("click", () => {
    callLoginpage(rootURL + "product_info", "selectall").then((data) => {
      console.log(data)
      if (data.data) {
        alert(data.message)

        // let array_data = {
        document.getElementById("nvdia_all_prod").innerHTML = generateHTML(data);

        function generateHTML(data) {
          // สร้าง HTML code โดยใช้ลูป
          const product_nvdia = data.data.map(item => `
            <style>
              /* CSS */
              body {
              align-items: center;
              }
              .product_all_index {
                  align-items: center;
                  background-color: #ececec;
              }
            </style>
            <body>
            <div class="product_all_index" style="border-radius: 10px;  padding: 10px; margin: 10px; float: left; width: 30%;">
              <img src="${item.pic_product}">
              <h3>Name: ${item.name_product}</h3>
              <h4>Price : ${item.price_product}</h4>
            </div>
            </body>
          `).join('');


          return product_nvdia;
        }

      } else {
        alert("no product")
      }
    })
  })
}
```

(index.html)

เมื่อ click ที่ปุ่ม product ที่แถบ navigation bar จะเจอกับช่องค้นหา ที่ลองรับการค้นหาทั้งหมด 3 ประเภท ได้แก่ ค้นหาจากผู้ผลิต แบรนด์ และซีรีย์ ซึ่งในหน้านี้จะสามารถดูสินค้าทั้งหมดได้ผ่านปุ่ม All Product



เมื่อทำการใส่ข้อมูลและกดปุ่ม search จะเข้าเงื่อนไขต่างๆเผื่อตรวจว่ามีช่องที่รับข้อมูลเข้ามากี่ช่อง และเป็นช่องใดบ้าง จากนั้นจะทำการ fetch เพื่อดึงข้อมูล โดยจะแยกไปดึงข้อมูลจากคนละ path

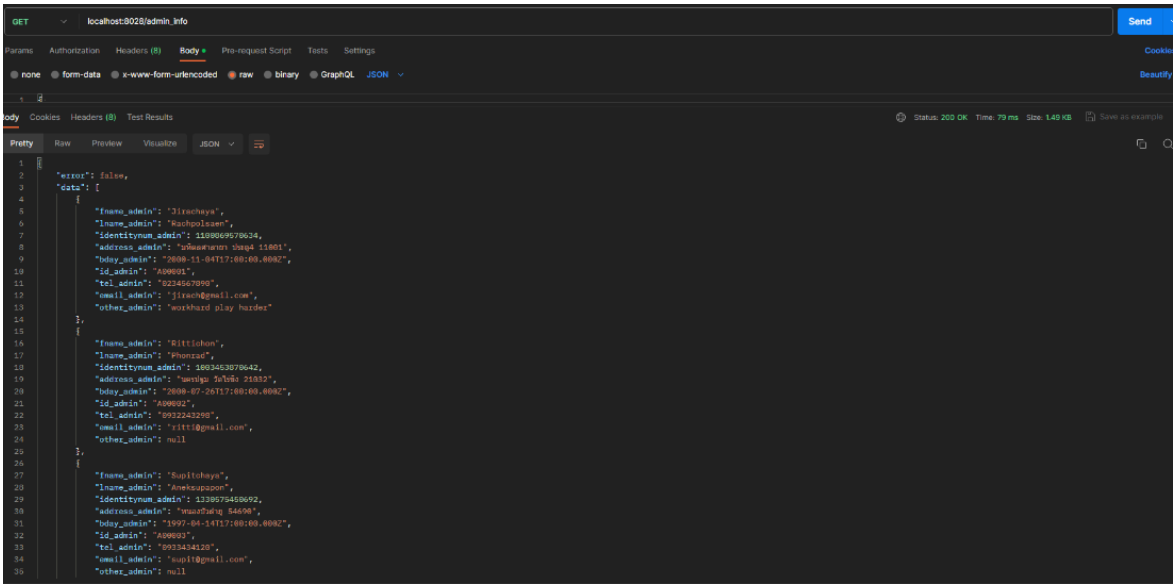
(product.html)

```
if (search_product_by_type) {
search_product_by_type.addEventListener("click", () => {
  input_product_manufac = input_product_manufac.value;
  input_product_brand = input_product_brand.value;
  input_product_series = input_product_series.value;
  console.log(input_product_manufac, input_product_brand ,input_product_series);

    if(input_product_manufac && !input_product_brand && !input_product_series){
      callProduct(rootUrl + input_product_manufac, "get").then((data) => {
      console.log(data);
      if (data.data) {
        alert("Select by Manufacturer");

        if (Array.isArray(data.data)) {
          document.getElementById("product_container").innerHTML = generateHTML(data);
          input_product_manufac.value = "";
        }
      } else {
        alert("Not Found");
      }
    });
  } else if(!input_product_manufac && input_product_brand && !input_product_series){
    callProduct(rootUrl + input_product_brand, "get").then((data) => {
      console.log(data);
      if (data.data) {
        alert("Select by Brand");

        if (Array.isArray(data.data)) {
          document.getElementById("product_container").innerHTML = generateHTML(data);
          input_product_manufac.value = "";
        }
      } else {
        alert("Not Found");
      }
    });
  }else if(!input_product_manufac && !input_product_brand && input_product_series){
    callProduct(rootUrl + input_product_series, "get").then((data) => {
      console.log(data);
      if (data.data) {
        alert("Select by Series");

        if (Array.isArray(data.data)) {
          document.getElementById("product_container").innerHTML = generateHTML(data);
          input_product_manufac.value = "";
        }
      } else {
        alert("Not Found");
```

ผลการทดลองของเว็บเซอร์วิสทั้งหมด โดยโปรแกรม Postman หรือโปรแกรมอื่นๆ

- Postman Admin

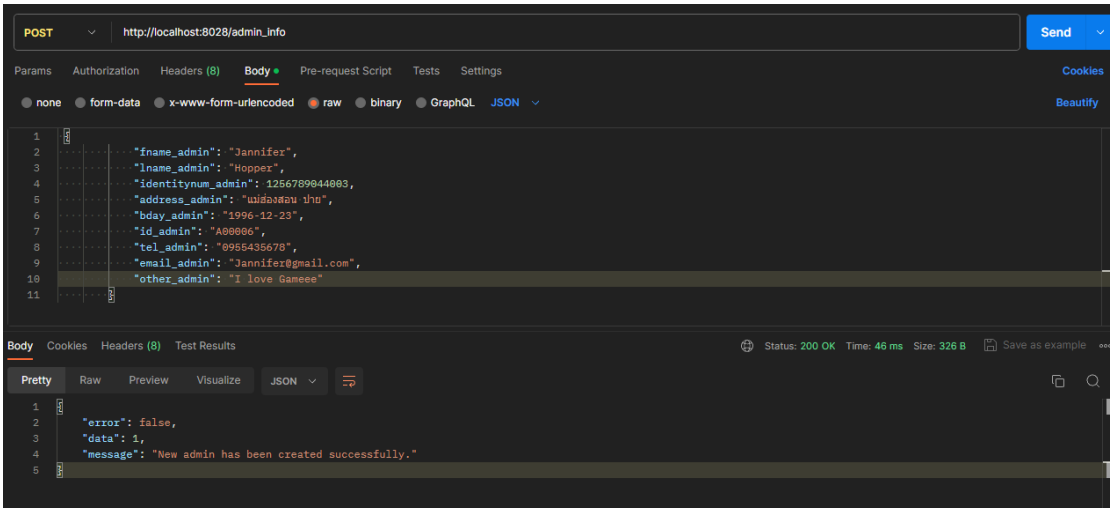
Get admin

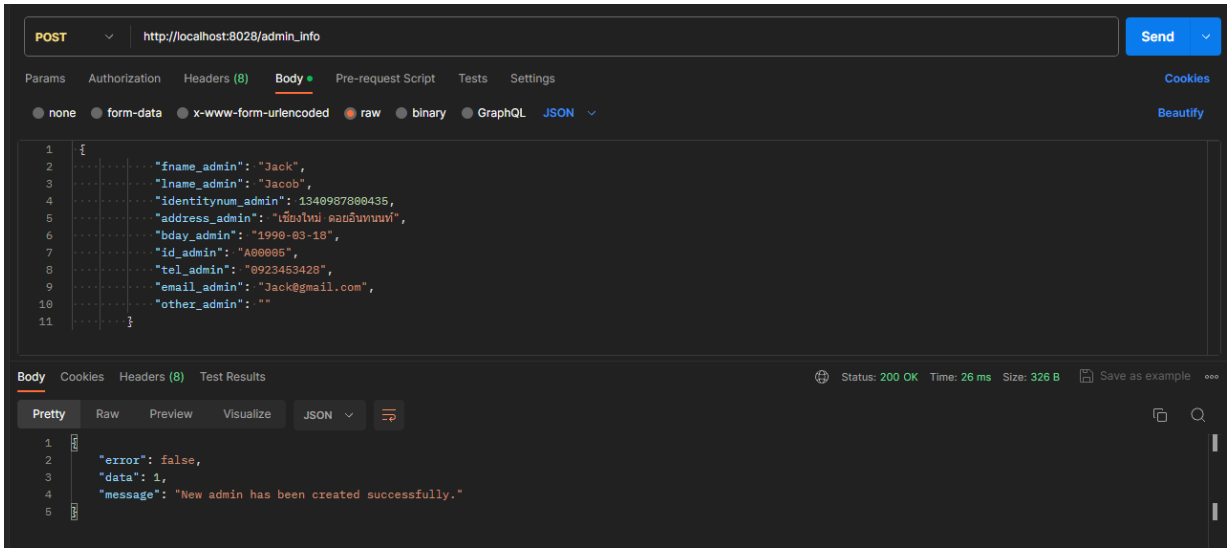


Code get admin

```
/* select all*/
router.get('/admin_info', function (req,res){
    connection.query("select * from admin_info", function (error, results){
        if(error) throw error;
        return res.send({
            error: false,
            data: results,
            message: "Admin All Information",
        });
    })
})
```

insert admin

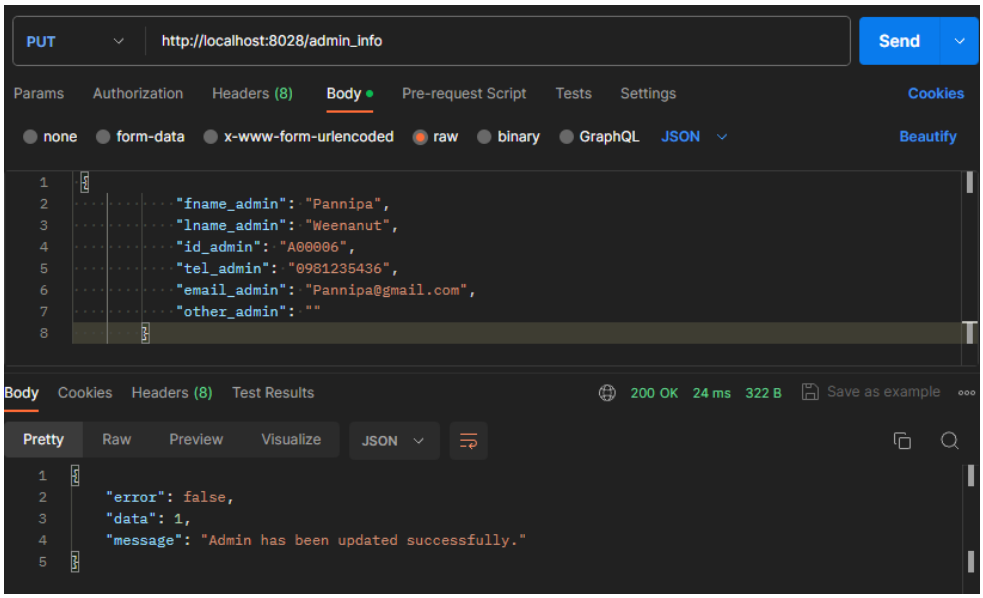


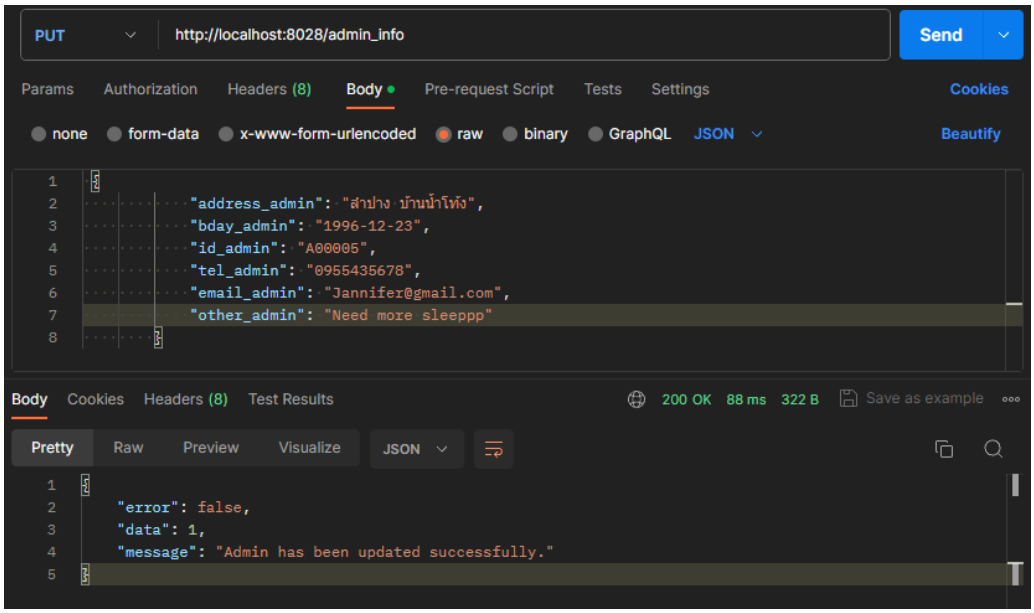


Code insert admin

```
/* INSERT */
router.post('/admin_info', function (req,res){
    let admin_info = req.body;
    console.log(admin_info);
    if (!admin_info){
        return res.status(400).send({error: true, messege: " Please provide product infomation"});

    }
    connection.query("insert into admin_info set ? ", admin_info, function (error, results){
            if (error)
              return res.send({
                error: admin_info,
                message: "The admin information incorrect.",
              });
        return res.send({
            error: false,
            data: results.affectedRows,
            message: "New admin has been created successfully.",
          });
    })
})
```

Update admin

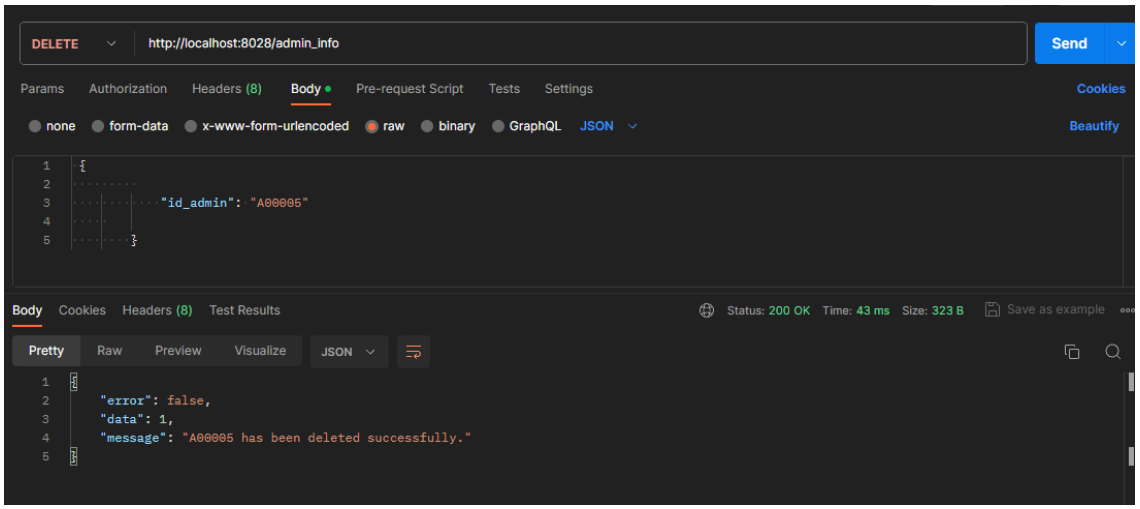


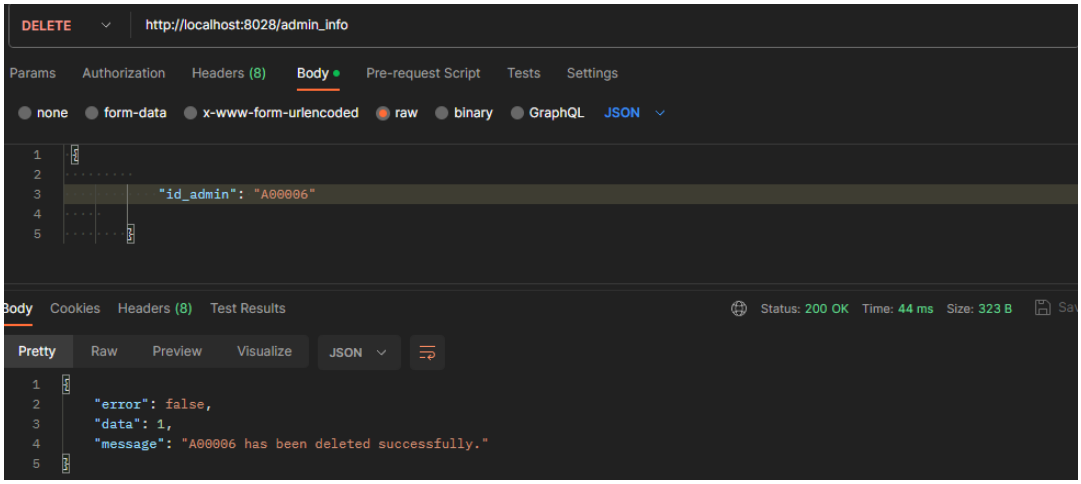


Code update admin

```
/* Update */
router.put('/admin_info', function (req,res){
    let adminID = req.body.id_admin;
    let admin_info = req.body;
    console.log(adminID);

    if (!admin_info || !adminID){
        return res.status(400).send({error: true, messege: " Please provide admin infomation"});
    }
    connection.query("update admin_info set ? where id_admin = ?", [admin_info,adminID], function (error, results){
        if(error) throw error;
        return res.send({
            error: false,
            data: results.affectedRows,
            message: "Admin has been updated successfully.",
          });
    })
})
```

Delete admin

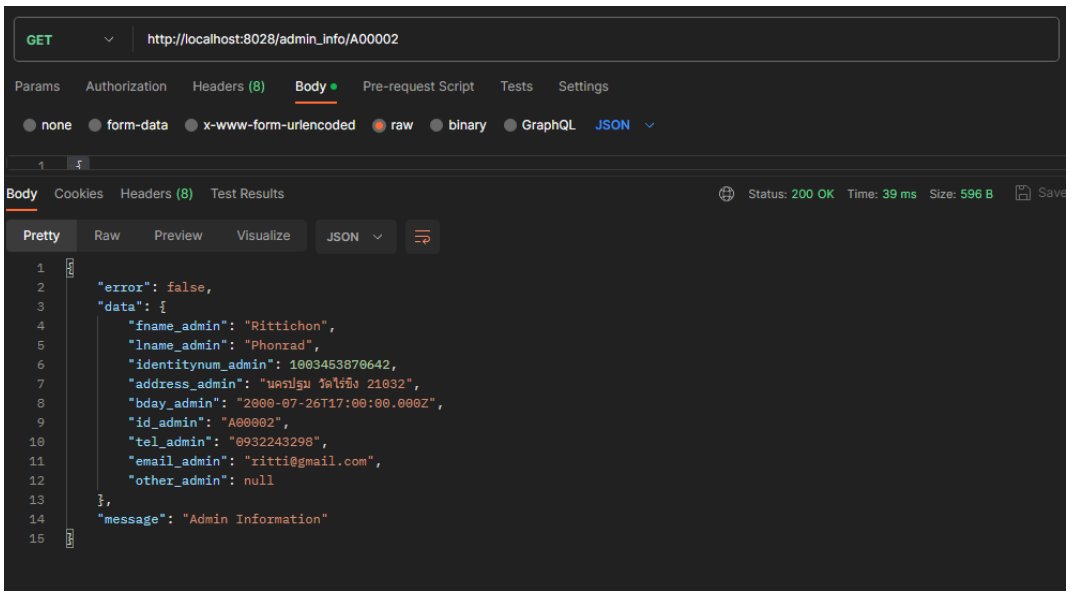




Code delete admin

```
/* DELETE */
router.delete('/admin_info', function (req,res){
    let adminID = req.body.id_admin;
    console.log(adminID);

    if (!adminID ){
        return res.status(400).send({error: true, messege: " Please provide ID admin"});
    }
    connection.query("DELETE from admin_info where id_admin = ?", adminID, function (error, results){
        if(error) throw error;
        return res.send({
            error: false,
            data: results.affectedRows,

            message: adminID+" has been deleted successfully.",
        });
    })
})
```

Select by ID admin

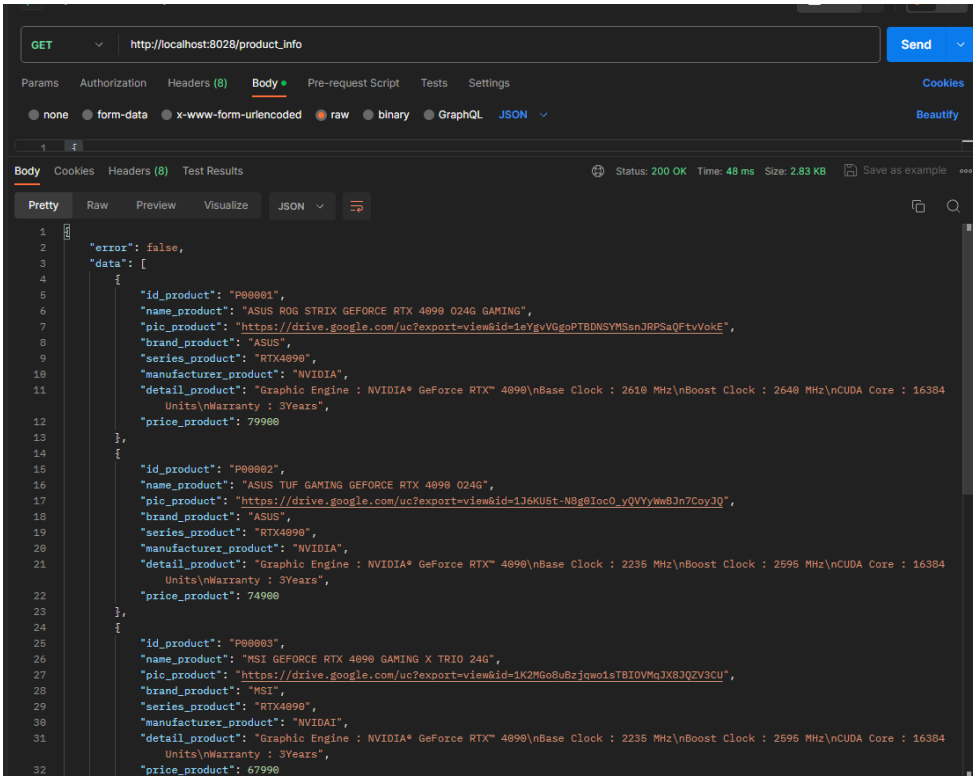


Code select by ID admin

```
// http://localhost:8028/admin_info/A00002
 select */
router.get('/admin_info/:id', function (req,res){
    let adminID = req.params.id;
    console.log(adminID);

    if (!adminID ){
        return res.status(400).send({error: true, messege: " Please provide ID admin"});
    }
    if(adminID.indexOf("A000") !== -1 || adminID.indexOf("a000") ) {
        connection.query("select * from admin_info where id_admin like  ?", `%${adminID}%`, function (error, results){
            if(error) throw error;

            return res.send({
                error: false,
                data: results[0],
                message: "Admin Information",
              });


})
    })
```

- Postman Product

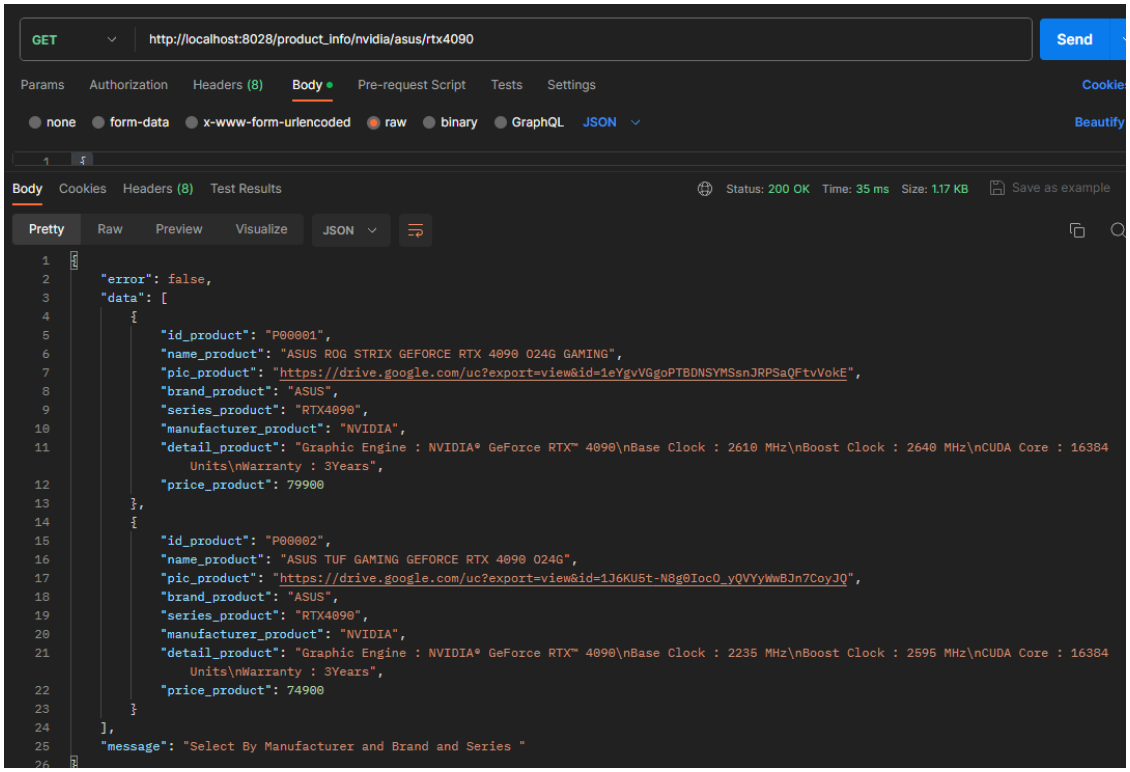
Product get all



Code product get all

```
/* select all*/
router.get('/product_info', function (req,res){
    connection.query("select * from product_sistem", function (error, results){
        if(error) throw error;
        return res.send({
            error: false,
            data: results,
            message: "select all of product",
          });
    })
})
```

Select Product by input 3 ตัว

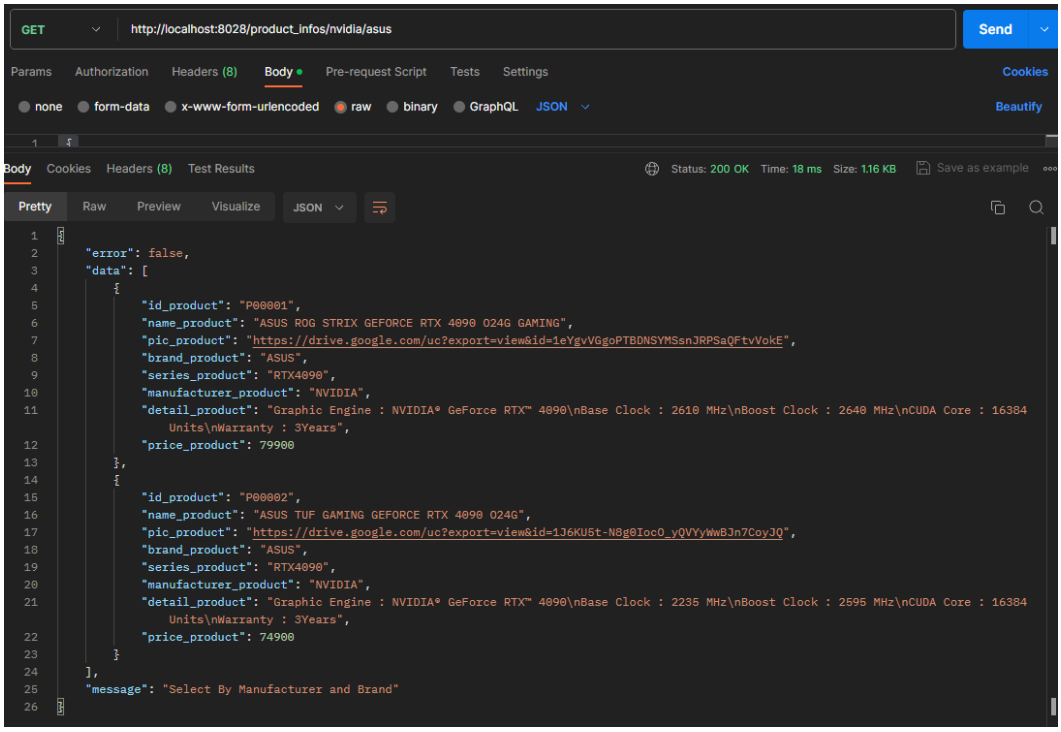


code Select Product by input 3 ตัว

```
router.get('/product_info/:input1/:input2/:input3', function (req, res) {
    let product_by_manufac = req.params.input1;
    let product_by_brand = req.params.input2;
    let product_by_series = req.params.input3;

    console.log(product_by_manufac);
    console.log(product_by_brand);
    console.log(product_by_series);

    // Check if both id and name are provided
    if (!product_by_manufac || !product_by_brand|| !product_by_series){
        return res.status(400).send({ error: true, message: "Please provide both ID and Name" });
    }

    // Query the database to check if both values exist in the same row
    connection.query("SELECT * FROM product_sistem WHERE manufacturer_product LIKE ? AND brand_product LIKE ? AND series_product LIKE ?",
     [`%${product_by_manufac}%`, `%${product_by_brand}%`,  `%${product_by_series}%`], function (error, results) {
        if (error) throw error;

        if (results.length === 0) {
            return res.status(404).send({ error: true, message: "Product not found" });
        } else {
            return res.send({
                error: false,
                data: results,
                message: "Select By Manufacturer and Brand and Series ",
            });
        }
    });
});
```

Select Product by input 2 ตัว

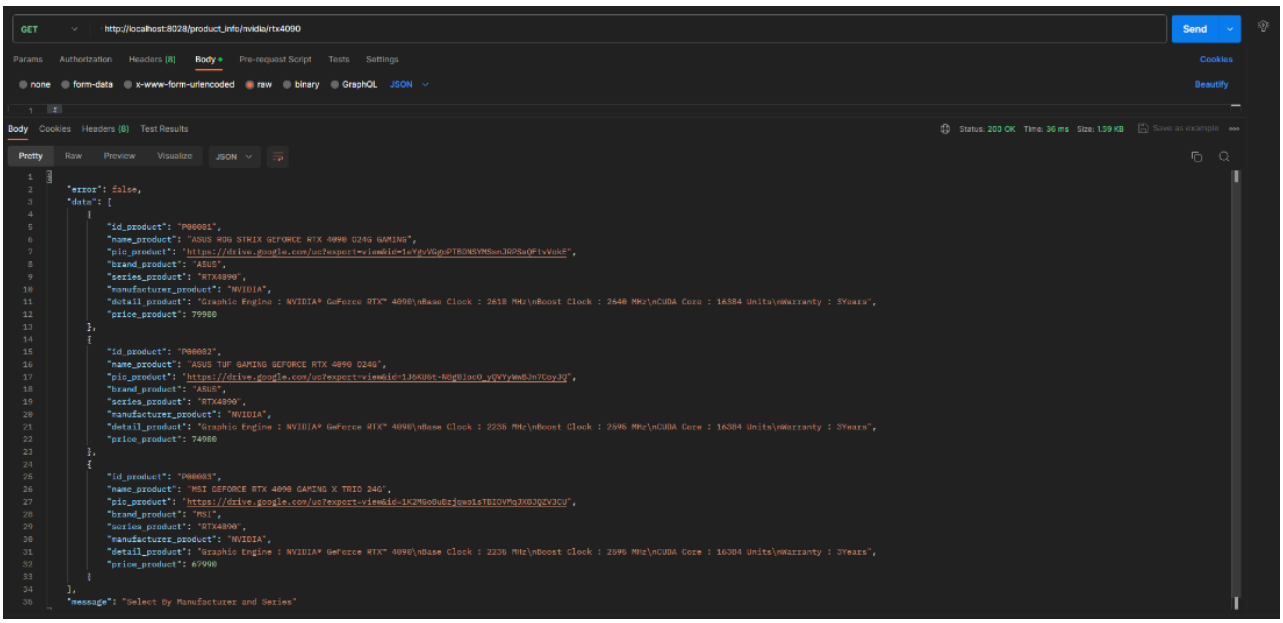
Manufa and brand



Code Manufa and brand

```
router.get('/product_infos/:input1/:input2', function (req, res) {
    let product_by_manufac = req.params.input1;
    let product_by_brand = req.params.input2;

    console.log(product_by_manufac);
    console.log(product_by_brand);

    if (!product_by_manufac || !product_by_brand) {
        return res.status(400).send({ error: true, message: "Please provide both ID and Name" });
    }

    connection.query("SELECT * FROM product_sistem WHERE manufacturer_product LIKE ? AND brand_product LIKE ?",
    [`%${product_by_manufac}%`, `%${product_by_brand}%`], function (error, results) {
        if (error) throw error;

        if (results.length === 0) {
            return res.status(404).send({ error: true, message: "Product not found" });
        } else {
            return res.send({
                error: false,
                data: results,
                message: "Select By Manufacturer and Brand",
            });
        }
    });
});
```

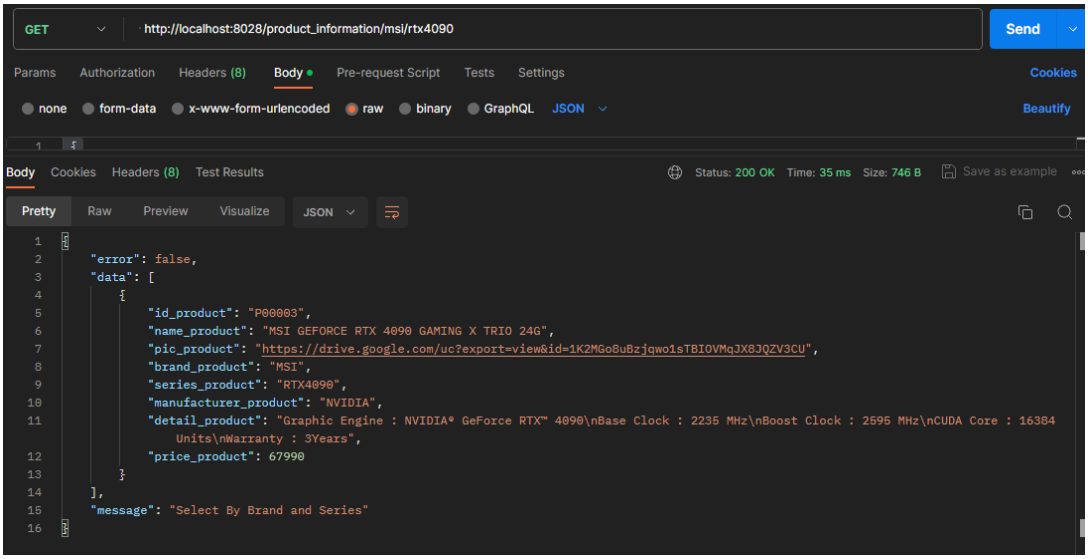
Manufac and series



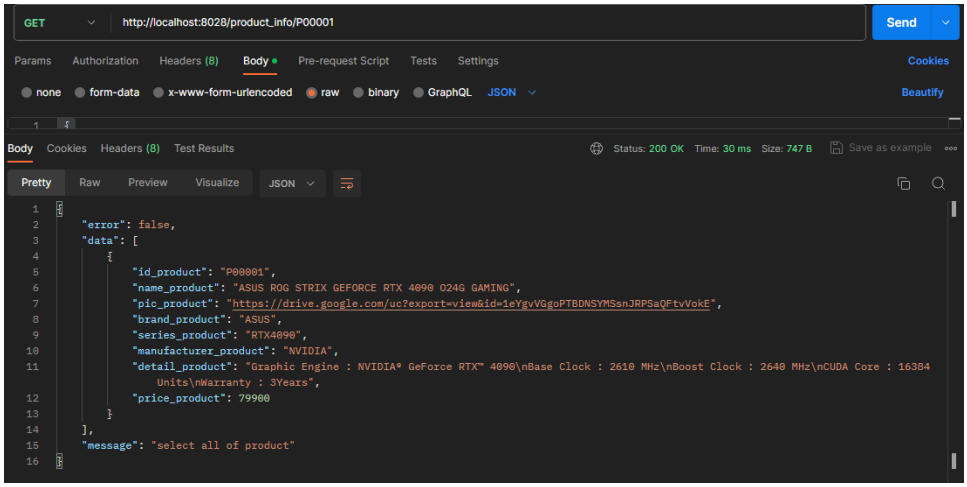
Code Manufac and series

```
router.get('/product_info/:input1/:input2', function (req, res) {
    let product_by_manufac = req.params.input1;
    let product_by_series = req.params.input2;

    console.log(product_by_manufac);
    console.log(product_by_series);

    if (!product_by_manufac || !product_by_series) {
        return res.status(400).send({ error: true, message: "Please provide both ID and Name" });
    }

    connection.query("SELECT * FROM product_sistem WHERE manufacturer_product LIKE ? AND series_product LIKE ?",
     [`%${product_by_manufac}%`, `%${product_by_series}%`], function (error, results) {
        if (error) throw error;

        if (results.length === 0) {
            return res.status(404).send({ error: true, message: "Product not found" });
        } else {
            return res.send({
                error: false,
                data: results,
                message: "Select By Manufacturer and Series",
            });
        }
    });
});
```

Brand and series



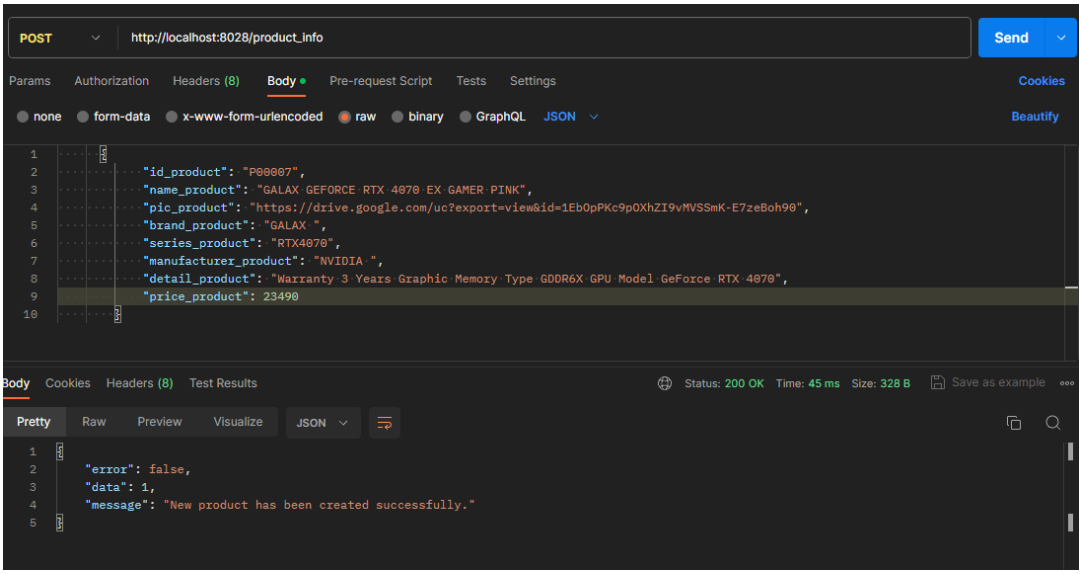
ID Product



Code ID product

```
router.get('/product_info/:id', function (req, res) {
    let productID = req.params.id;

    if (!productID) {
        return res.status(400).send({error: true, message: "Please provide ID product"});
    }

    const regex = new RegExp(productID, 'i');

    connection.query("SELECT * FROM product_sistem WHERE manufacturer_product LIKE ? OR brand_product LIKE ? OR id_product LIKE ? OR series_product
    [`%${productID}%`, `%${productID}%`, `%${productID}%`, `%${productID}%`], function (error, results){
        if (error) throw error;

        if (results.length === 0) {
            return res.status(404).send({error: true, message: "Product not found"});
        } else {
            return res.send({
                error: false,
                data: results,
                message: "Select "+ productID,
            });
        }
    });
});
```

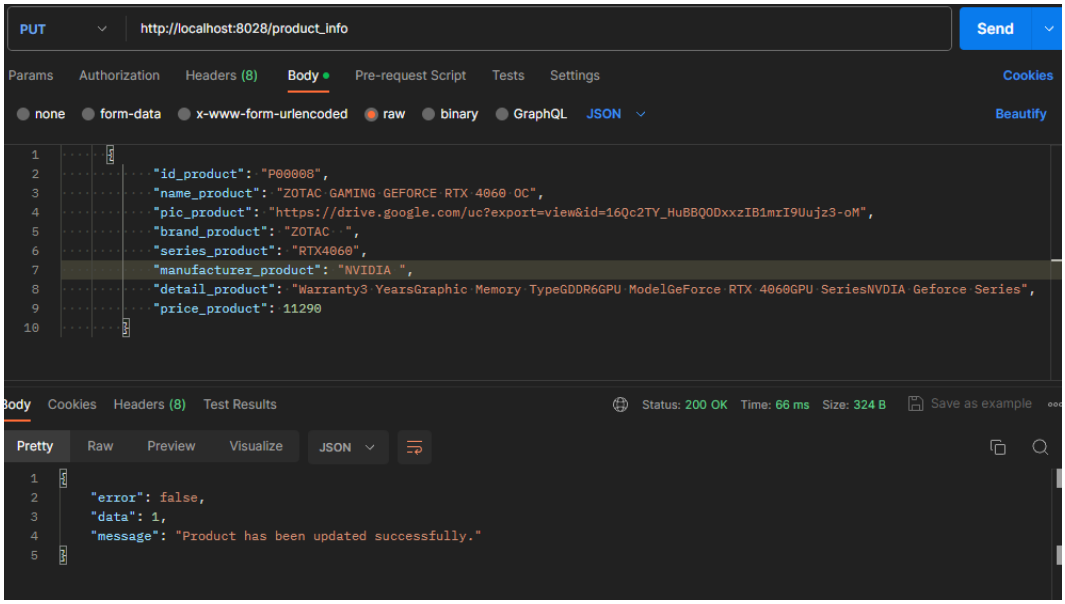
Insert product



Code Insert product

```
router.post('/product_info', function (req,res){
    let product_info = req.body;
    console.log(product_info)
    if (!product_info){
        return res.status(400).send({error: true, messege: " Please provide product infomation"});

    }
    connection.query("insert into product_sistem set ? ", product_info, function (error, results){
            if (error)
              return res.send({
                error: product_info,
                message: "The product incorrect.",
              });
        return res.send({
            error: false,
            data: results.affectedRows,
            message: "New product has been created successfully.",
          });
    })
})
```

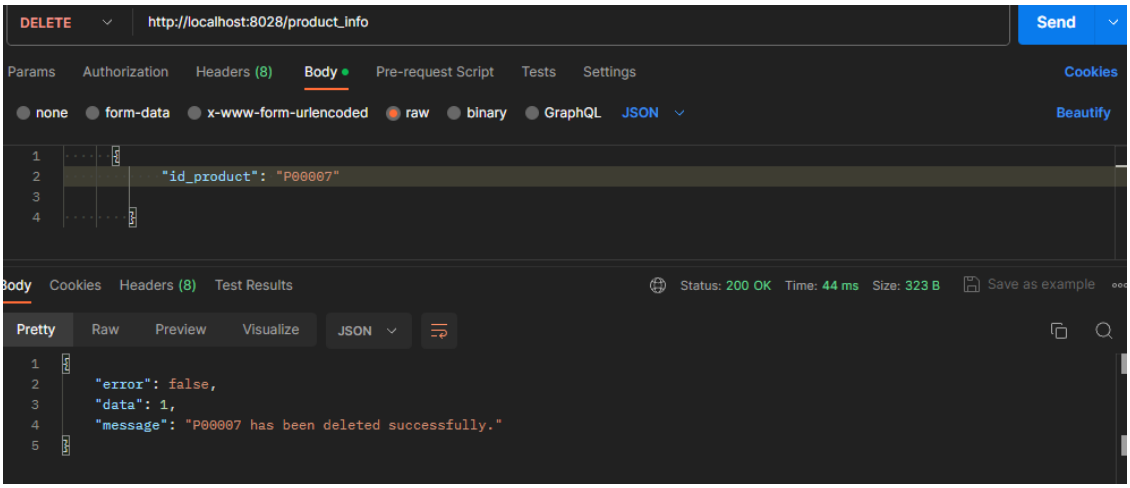
Update product



Code Update product

```
/* Update */
router.put('/product_info', function (req,res){
    let productID = req.body.id_product;
    let product_info = req.body;
    // console.log(productID);
    // console.log(product_info);

    if (!product_info || !productID){
        return res.status(400).send({error: true, messege: " Please provide product infomation"});
    }
    connection.query("update product_sistem set ? where id_product = ?", [product_info,productID], function (error, results){
        if(error) throw error;
        return res.send({
            error: false,
            data: results.affectedRows,
            message: "Product has been updated successfully.",
          });
    })
})
```

Delete product

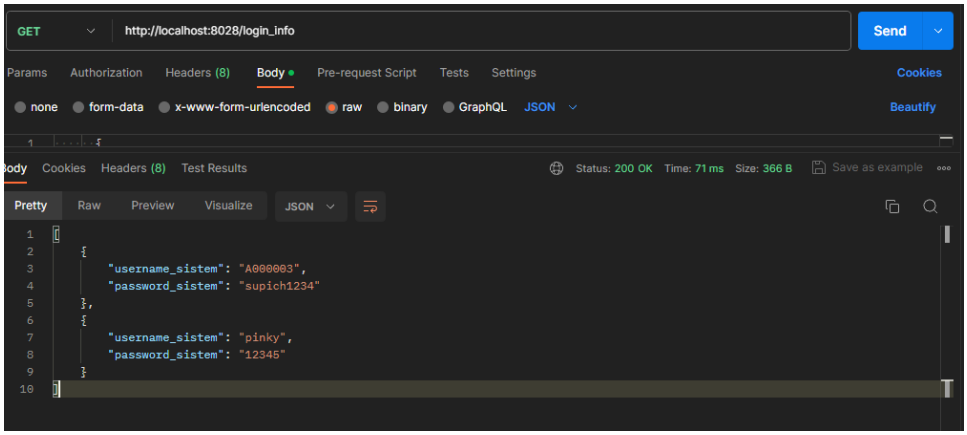


Code Delete product

```
/* DELETE */
router.delete('/product_info', function (req,res){
    console.log("deleteee")
    let productID = req.body.id_product;

    if (!productID ){
        return res.status(400).send({error: true, messege: " Please provide ID product"});
    }


    connection.query("DELETE from product_sistem where id_product = ?", productID, function (error, results){
        if(error) throw error;
        return res.send({
            error: false,
            data: results.affectedRows,

            message: productID+" has been deleted successfully.",
        });
    })
})
```

- Postman Log in

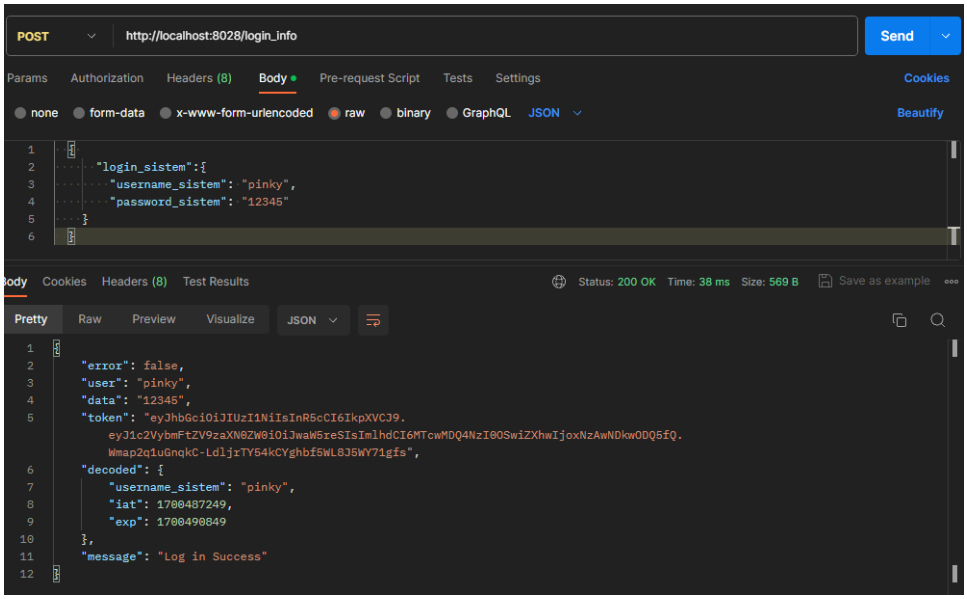
Selecte all Log in user



Code Selecte all Log in user

```
/* log in  */
router.get('/login_info', function (req,res){
    connection.query("select * from login_sistem", function (error, results){
        if(error) throw error;
        return res.json(results);
    })
})
```

Log in

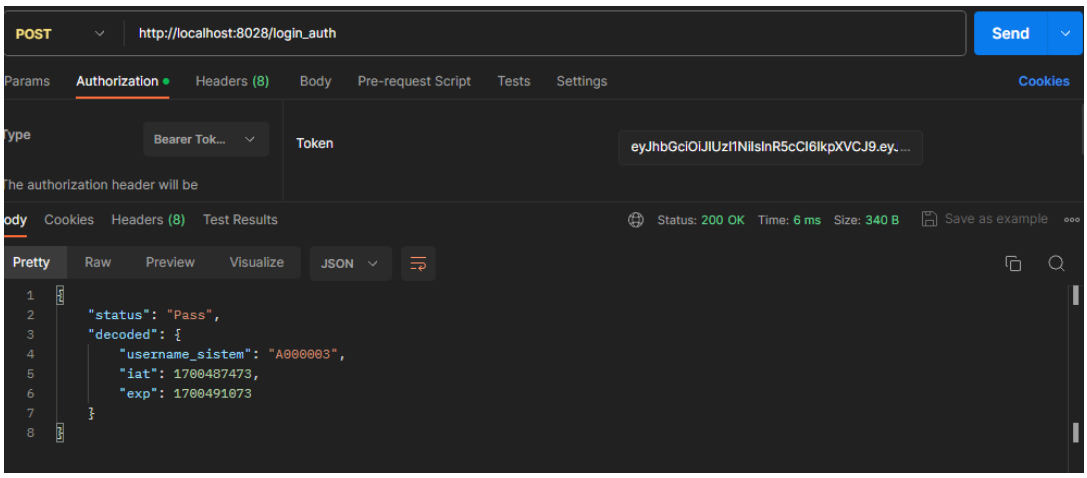


Code Log in

```
router.post('/login_info', function (req,res){
    console.log("login")
    console.log(req.body);
    let login_info = req.body;
    let username_sistem = login_info.login_sistem.username_sistem;
    let password_sistem = login_info.login_sistem.password_sistem;
    console.log(password_sistem)

    if ( !login_info ||!username_sistem || !password_sistem) {
        return res.status(400).send({ error: true, message: "Please provide information for login" });
    }

    connection.query("select * from login_sistem where username_sistem= ? ",username_sistem , function (error, results){

        if(error) throw error;


        if (results.length > 0){
            const retrievedPassword = results[0].password_sistem;
            console.log("Retrieved Password:", retrievedPassword);
            if(password_sistem === retrievedPassword ){
            var jwtToken = jwt.sign(
                {
                    username_sistem:username_sistem ,
                },
                process.env.SECRET,
                {
                    expiresIn: "1h",
                }
            )
            console.log(login_info)
            console.log(jwtToken);
```

```
            console.log(jwtToken);
            console.log(password_sistem)
            try{

                var decoded = jwt.verify(jwtToken,process.env.SECRET,)

                console.log(jwtToken,decoded)
            }catch(err){

                console.log({status: 'Fail', message: err.message})
            }


        }  console.log(jwtToken)
            return res.send({
            error: false,
            user: username_sistem,
            data: retrievedPassword,
            token: jwtToken,
            decoded: decoded,
            message: "Log in Success",
        });
    }else {
        return res.status(400).send({ error: true, message: "Invalid username_sistem" });
    }
```

Check token



Code Check token

```
router.post('/login_auth', function (req,res){
    console.log("print token")
    try{
        const token = req.headers.authorization.split(' ')[1]
        var decoded = jwt.verify(token,process.env.SECRET,)
        res.json({status: 'Pass', decoded})

    }catch(err){
        res.json({status: 'Fail', message: err.message})
    }

})
```