

WEB PROGRAMMING

FINAL REPORT

Members

Pimsuang Kanjanatawewat 6288021

Nithinee Trironnarith 6288055

Phuwanat Meeyutem 6288095

Nawadol Kuchaipoom 6288098

ITCS212 Web Programming

Faculty of Information and Communication

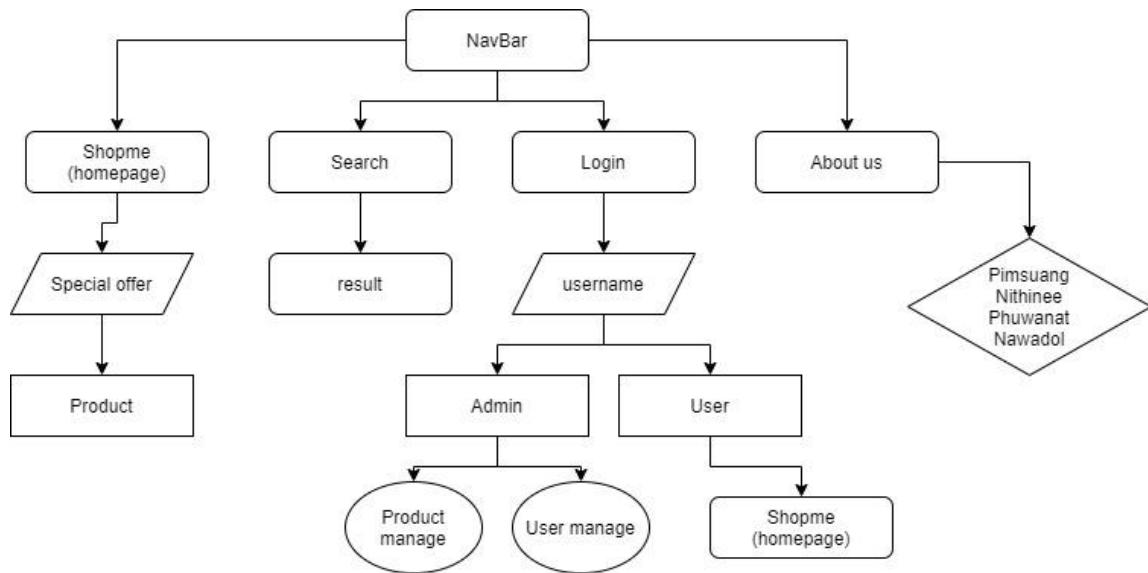
Technology

Mahidol University

2020

Overview

Navigation Diagram

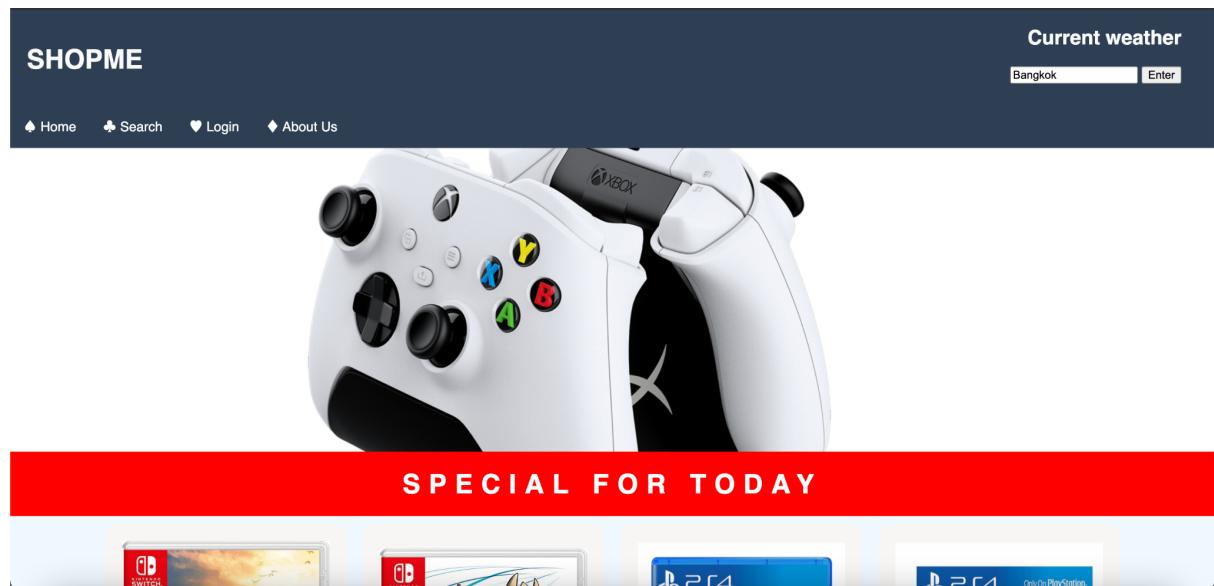


Phase1

We created 5 pages using html and CSS.

Homepage

This page shows the special offer product.



search page

This page is for searching.

SHOPME

♦ Home ♦ Search ♦ Login ♦ About Us

SHOPEME

Search...

login page

This page is for users to login.

SHOPME

♦ Home ♦ Search ♦ Login ♦ About Us

Log in

username

Password

Log in

Result page

This page shows the results.

A screenshot of a web browser displaying the 'SHOPME' website. The page title is 'SHOPME'. The navigation bar includes links for Home, search more, Login, and About us. A search bar with the placeholder 'search' is located at the top right. Below the navigation, a link labeled '← Back' is visible. The main content area is titled '"RESULT"' and shows four game covers: 'OVERWATCH', 'SLENDER: THE ARRIVAL', 'FALL GUYS: ULTIMATE KNOCKOUT', and a cartoon character wearing a crown.

About us page

This page is about the creator's information.

The 'About us' page displays four student profiles, each represented by a black suit jacket icon with a large white question mark on the head. The profiles are arranged in a row:

- 6288021 PIMSUANG KANJANATAWEWAT**
Email: pimsuang.kan@student.mahidol.ac.th
[more](#)
- 6288055 NITHINEE TRIRONARITH**
Email: nithinnee.tri@student.mahidol.ac.th
[more](#)
- 6288095 PHUWANAT MEEYUTEM**
Email: phuwat.mee@student.mahidol.ac.th
[more](#)
- 6288098 NAWADOL KUCHAIPOOM**
Email: nawadol.kuc@student.mahidol.ac.th
[more](#)

Phase2

We use node.js to make a server side of the our website from phase1

Homepage

This page shows our products that have a special offer.

The image displays two screenshots of the SHOPME website's homepage. Both screenshots feature a dark header bar with the 'SHOPME' logo and navigation links for Home, Search, Login, and About Us. In the top screenshot, the main visual is a white Microsoft Xbox One controller. In the bottom screenshot, the main visual is a character from the video game 'The Legend of Zelda: Breath of the Wild', Link, standing in a field under a cloudy sky. Below the main visual is a red horizontal bar with the text 'SPECIAL FOR TODAY'. At the bottom of each screenshot, there is a row of four small icons representing different gaming platforms: Nintendo Switch, Xbox, PlayStation 4 (PS4), and PlayStation 5 (PS5). To the right of the main visual in the bottom screenshot is a weather widget showing current weather information for Bangkok: temperature (30.71°C), visibility (10000m), wind speed (2.06 m/s), and weather conditions (few clouds).



Legend of Zelda: Breath of the Wild

1680฿

Step into a world of discovery, exploration, and adventure in The Legend of Zelda.

[Add to Cart](#)



Pokemon Shield

1690฿

Pokemon Sword and Pokemon Shield introduce the Galar region and more Pokemon to discover!

[Add to Cart](#)



Overwatch

1200฿

The World Needs Heroes Slow down time, rain destruction from above in a jet-powered armor suit, or pilot superpower hamster in ball.

[Add to Cart](#)



Detroit: Become Human

1200฿

Puts the destiny of both mankind and androids in your hands, taking you to a future where machines have become more intelligent than human.

[Add to Cart](#)

[Back to Top](#)

[about us](#)

[email](#)

[address](#)

Login

This page lets the user login.

SHOPME

Please Login first

[login](#)



On our server side, It will check that the user role is admin or not. if the user is admin, it will redirect the user to the admin page. if the role is a user, it will redirect the user to homepage.

```
app.post('/auth', cors(), function (request, response) {
  var username = request.body.username;
  var password = request.body.password;

  //res.json({success : true})
  //if("SELECT* FROM user WHERE urole ='user'")

  if (username && password) {
    connection.query('SELECT * FROM user WHERE username = ? AND pword = ?', [username, password], function (error, results, fields) {
      if (results.length > 0) {
        if (results[0].urole === 'admin') {
          request.session.loggedin = true;
          request.session.username = username;

          //alert("Welcome");

          //
          console.log(results);
          console.log(`Welcome User: ${username}`);
          authUser(request, response, results[0].urole)
          response.redirect('/admin');

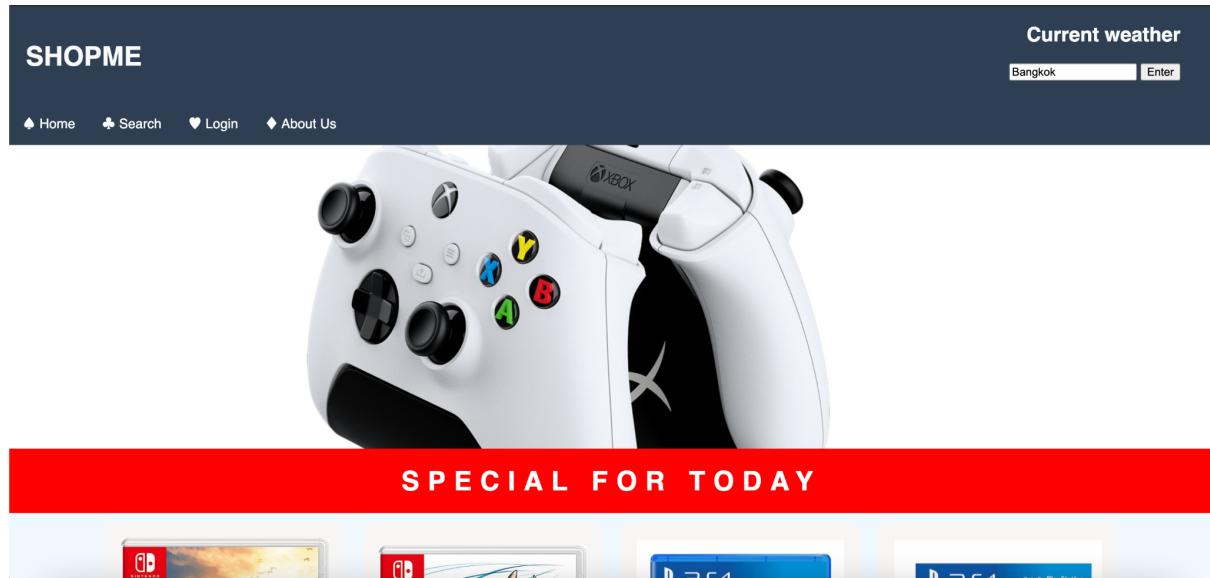
        } else {
          request.session.loggedin = true;
          request.session.username = username;

          //alert("welcome");

          //
          console.log(results);
          console.log(`Welcome User: ${username}`);
        }
      }
    })
  } else {
    console.log('Incorrect Username and/or Password!');
    response.redirect('/auth-alert');
  }
  response.end();
});
```

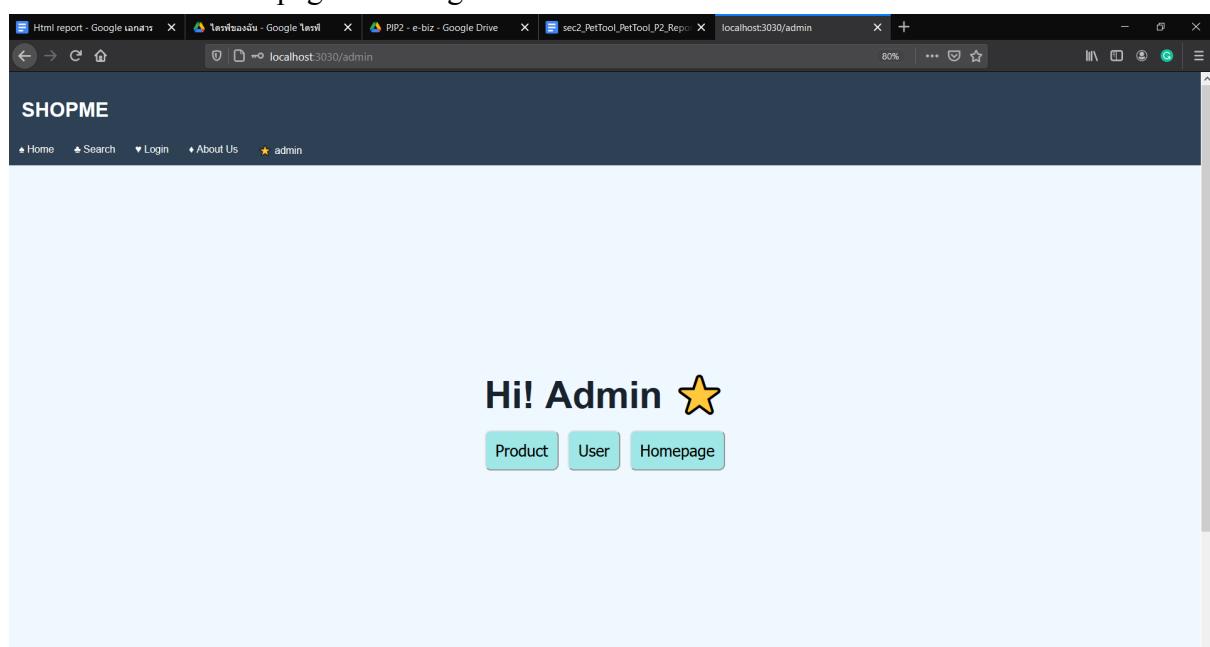
Normal User Pages with explanation for each page

Homepage for showing special products and have other buttons to go to the other page also the upper right is telling current weather as the user wants to see.



Administration Pages with explanation for each page

Admin home page when login admin role.



When clicking the User button it will go to the Userform page allowing the user to create, update, delete and etc. the user or admin then put and edit in database.

SHOPME

User

UserID: 123456

Firstname: XXXXX

Lastname: XXXXX

Email: xxx@xxx.xx

address: address

Phonenumber: 0xxxxxxxxx

Date of birth: YYYY-MM-DD

Username: Username

Password: Password

Role: admin/user

Insert **Update** **Delete** **Select** **Select all**

In each button, we will fetch the service from our server.

Add user

```

    };
    callUSERWS("http://localhost:3030/user", "insert", user_data).then(
      (data) => {
        console.log(data);
        if (data.data > 0) {
          alert(data.message);
          clearInput();
        }
      );
    });
  updateBtnRef.addEventListener("click", () => {

```

Update user

```

    callUSERWS("http://localhost:3030/user", "update", user_data).then(
      (data) => {
        console.log(data);
        if (data.data > 0) {
          alert(data.message);
          clearInput();
        }
      );
    );
  });

```

Delete user

```
        callUSERWS("http://localhost:3030/user", "delete", user_data).then(  
            (data) => {  
                console.log(data);  
                if (data.data > 0) {  
                    alert(data.message);  
                    clearInput();  
                }  
            }  
        );  
    })
```

Select user by ID

```
});  
selectBtnRef.addEventListener("click", () => {  
    UID = U_IDTxtRef.value;  
    callUSERWS("http://localhost:3030/user/" + UID, "select").then(  
        (data) => {  
            console.log(data);  
            if (data) {  
                alert(data.message);  
                U_IDTxtRef.value = data.data.UID;  
                U_FNAMETxtRef.value = data.data.fname;  
                U_LNAMETxtRef.value = data.data.lname;  
                U_EMAILTxtRef.value = data.data.Email,  
                U_ADDRESSTxtRef.value = data.data.address,  
                U_PHONETxtRef.value = data.data.pnumber,  
                U_DOBTxtRef.value = data.data.DOB;  
                U_USERTxtRef.value = data.data.username,  
                U_PASSTxtRef.value = data.data.pword,  
                U_ROLETxtRef.value = data.data.urole  
            }  
        }  
    );  
});
```

Select all users

```
});  
selectAllBtnRef.addEventListener("click", () => {  
    callUSERWS("http://localhost:3030/users", "select").then(  
        (data) => {  
            console.log(data);  
            if (data.data.length > 0) {  
                alert(data.message);  
                let output;  
                output = "<h1> user List </h1>";  
                output += "<table class='table'>";  
                output += "<thead>";  
                output += "<tr>";  
                output += "  |                output += "</tr>";  
                output += "</thead>";  
                output += "<tbody>";  
                data.data.forEach((element) => {  
                    output += "<tr>";  
                    output += "<td>" + element.UID + "</td>";  
                    output += "<td>" + element.fname + "</td>";  
                    output += "<td>" + element.lname + "</td>";  
                    output += "<td>" + element.Email + "</td>";  
                    output += "<td>" + element.role + "</td>";  
                    output += "</tr>";  
                });  
                document.getElementById("userList").innerHTML = output;  
            }  
        }  
    );  
});
```

When click to Product button it will go to productform page allow to create, update, delete and etc. the products then put and edit in database.

The screenshot shows a web application interface titled "SHOPME". At the top, there is a navigation bar with links for Home, Search, Login, About Us, and admin. The main content area is titled "Product". It contains several input fields: "PID" (with value "productID"), "ProductPic url" (with value "productpic url"), "Product name" (with value "Product name"), and "Product information" (with value "Product information"). Below these fields is a "Price" field containing the value "Price". At the bottom of the form are five buttons: "Insert", "Update", "Delete", "Select", and "Select all".

Add product

```
};  
callProductWS("http://localhost:3030/product", "insert", product_data).then(  
  (data) => {  
    console.log(data);  
    if (data.data > 0) {  
      alert(data.message);  
      clearInput();  
    }  
  }  
);
```

Update product

```
};  
callProductWS("http://localhost:3030/product", "update", product_data).then(  
  (data) => {  
    console.log(data);  
    if (data.data > 0) {  
      alert(data.message);  
      clearInput();  
    }  
  }  
);
```

Delete product

```
        };
        callProductWS("http://localhost:3030/product", "delete", product_data).then(
            (data) => {
                console.log(data);
                if (data.data > 0) {
                    alert(data.message);
                    clearInput();
                }
            }
        );
    })
});
```

Select product by ID

```
selectBtnRef.addEventListener("click", () => {
    PID = P_IDTxtRef.value;
    callProductWS("http://localhost:3030/product/" + PID, "select").then(
        (data) => {
            console.log(data);
            if (data) {
                alert(data.message);
                P_IDTxtRef.value = data.data.PID;
                P_PICTxtref.value = data.data.pic;
                P_NAMETxtRef.value = data.data.Pname;
                P_INFOTxtRef.value = data.data.pinfo;
                P_PRICETxtRef.value = data.data.price;
            }
        }
    );
});
selectAllBtnRef.addEventListener("click", () => {
```

Select all product

```
    },
    selectallBtnRef.addEventListener("click", () => {
        callProductWS("http://localhost:3030/products", "select").then(
            (data) => {
                console.log(data);
                if (data.data.length > 0) {
                    alert(data.message);
                    let output;
                    output = "<h1> Product List </h1>";
                }
            }
        );
    });
});
```

Search

This page lets the user input the name of the product to search for the desired product.



So, if we input the word, it will send a request to show the product.

```
        }
        callallproduct("http://localhost:3030/allproducts/" + out).then(
          (data) => {
            console.log(data);
            if (data.data.length > 0) {

              //alert(data.message);
              console.log(data.data)
              let output = "";
              output += `<h1><mark style="background-color:white"> Searching for "${out}"</mark></h1>`;
              output += "<div class='container'>";
              data.data.forEach((element) => {

                output += "<div class='card' style='width: 18rem;'>";
                output += "<img src='" + element.pic + "' class='card-img-top' alt='card'>";
                output += "<div class='card-body'>";
                output += "<h4><b>" + element.Pname + "</b></h4>";
                output += "<section class='price'>$" + element.price + "</section><br>";
                output += "<section class='des'>" + element.pinfo + "</section><br>";
                output += "<p><button>Add to cart</button></p>";
                output += "</div>";
                output += "</div>";
              });
              output += "</div>"

            }
          }
        )
      }

    
```

On the server side, It will check that the word that we input consists of the name of the product or not. Then it will show us the result of searching.

```

    route.get('/allproducts/:keyword', cors(),function (req, res) {
      console.log("result page request");
      console.log("search for = " + req.params.keyword);
      let word = req.params.keyword;
      if (word != "") {
        connection.query(`SELECT * FROM product WHERE Pname like '%${word}%'`, function (error, results) {
          if (error) throw error;
          else {

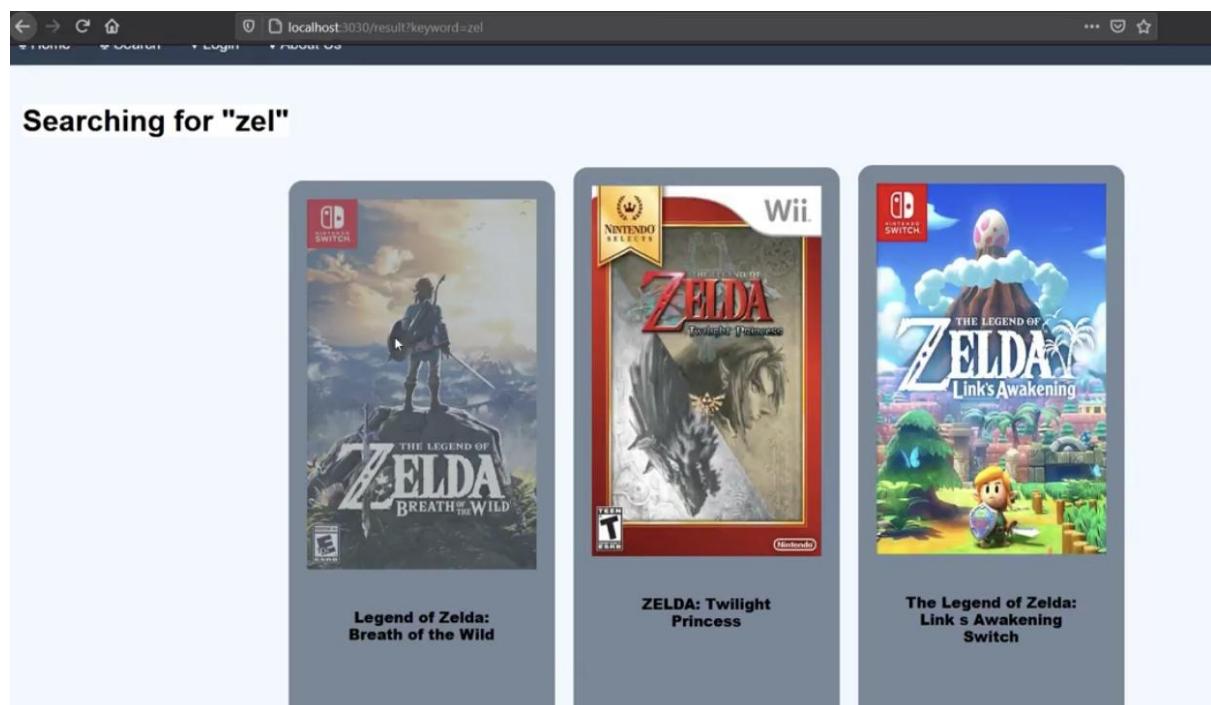
            if (results.length > 0) {
              return res.send({
                error: false,
                data: results,
                message: 'Product list'
              });
            } else {
              connection.query(`SELECT * FROM product`, function (error, results) {
                if (error) throw error;
                else {
                  return res.send({
                    error: false,
                    data: results,
                    message: 'Product list'
                  });
                }
              })
            }
          }
        })
      }
    })
  )
}

```

Example: “ZEL”

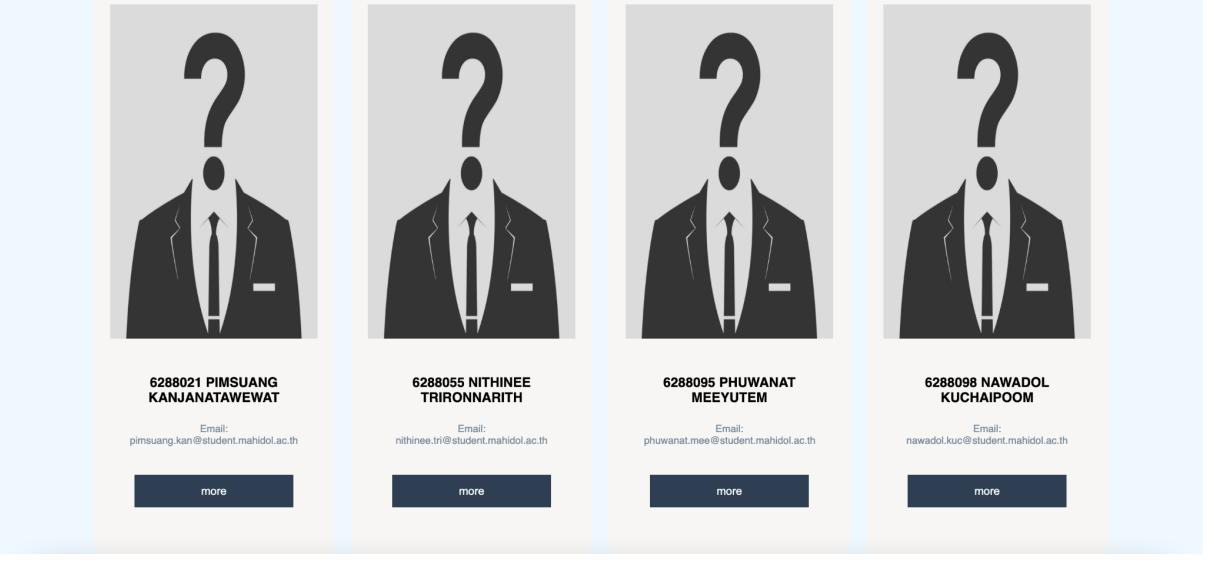
The products that contain the word ‘zel’ will be shown.

- Legend of Zelda: Breath of the Wild
- Zelda: Twilight Princess
- The Legend of Zelda: Link’s Awakening Switch



About us

This page shows the information of the developer.



The page displays four developer profiles, each consisting of a placeholder image (a suit jacket with a question mark for a head), the developer's ID and name, an email address, and a 'more' button.

ID	Name	Email	Action
6288021	PIMSUANG KANJANATAWEWAT	pimsuang.kan@student.mahidol.ac.th	more
6288055	NITHINEE TRIRONNARITH	nithinee.tri@student.mahidol.ac.th	more
6288095	PHUWANAT MEEYUTEM	phuwanat.mee@student.mahidol.ac.th	more
6288098	NAWADOL KUCHAIPOOM	nawadol.kuc@student.mahidol.ac.th	more

API

The api that we choose is the weather API. This API tells the current weather around the world. The reason why we choose this api is because when the user orders a product from our website we must deliver it to the customer so checking weather can know the obstacle with delivery.

Code to show in html in the homepage ask the user input country's name then show information about current weather from the input country.

```
297      </div>
298      <!-- Api weather<-->
299      <div>
300          <div class="right"><h2>Current weather</h2>
301          <div class="input">
302              <input type="text" placeholder="Type:Bangkok, New York or Japan" value="Bangkok" class="input_text">
303              <input type="submit" value="Enter" class="submit">
304          <div class="box2">
305              <h1 class="name" id="name"></h1>
306              <p class="temp"></p>
307              <p class="visibility"></p>
308              <p class="wind"></p>
309              <p class="desc"></p>
310          </div>
311          <script src="/static/weather.js"></script>
312      </div>
313  </div>
```

Raw data from the api we use to show on the homepage.

```

{
  "lat": 13.75,
  "weather": [
    {
      "id": 801,
      "main": "Clouds",
      "description": "few clouds",
      "icon": "02n"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 302.9,
    "feels_like": 309.02,
    "temp_min": 301.48,
    "temp_max": 305.37,
    "pressure": 1010,
    "humidity": 76
  },
  "visibility": 10000,
  "wind": {
    "speed": 3.12,
    "deg": 183,
    "gust": 4.44
  },
  "clouds": {
    "all": 24
  },
  "dt": 1619881191,
  "sys": {
    "type": 3,
    "id": 2003771,
    "country": "TH",
    "sunrise": 1619823378,
    "sunset": 1619868816
  },
  "timezone": 25200,
  "id": 1609350,
  "name": "Bangkok"
}

```

Javascript code for connect with api, we create various variables as we want from the api have

We take the country's name from the homepage's input, insert it into the api link and take the current data and show it to the homepage.

```

lines (23 sloc) | 1.08 KB

1 var input = document.querySelector('.input_text');
2 var main = document.querySelector('#name');
3 var temp = document.querySelector('.temp');
4 var desc = document.querySelector('.desc');
5 var wind = document.querySelector('.wind');
6 var visibility = document.querySelector('.visibility');
7 var button = document.querySelector('.submit');

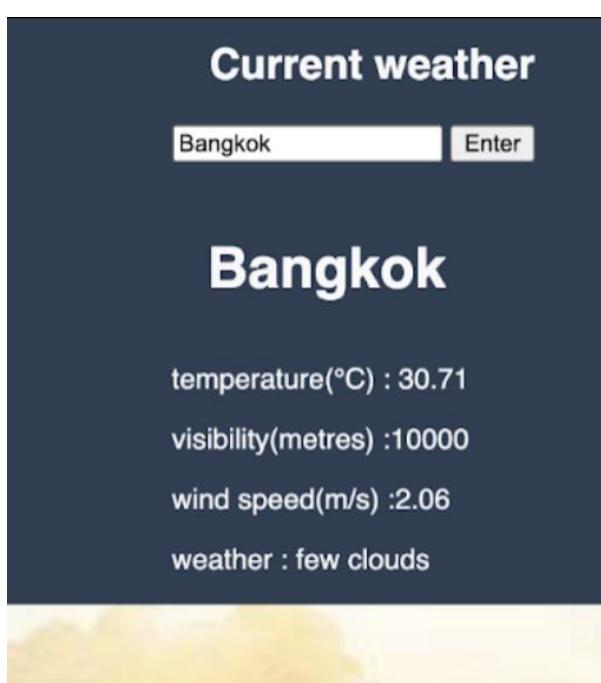
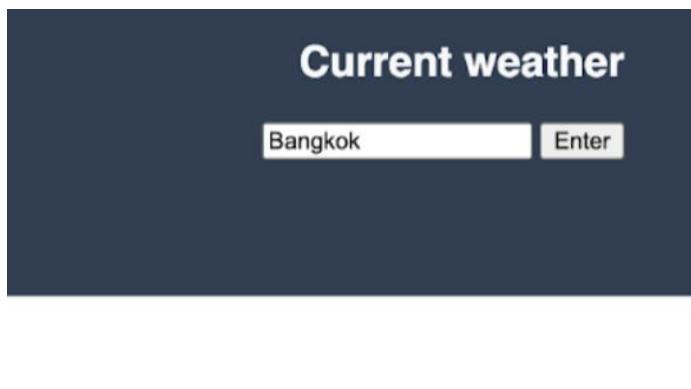
8

9
button.addEventListener('click', function () {
10   fetch('https://api.openweathermap.org/data/2.5/weather?q=' + input.value + '&appid=29d045b6f4e8f10d2c24deceec3a2785&units=metric')
11     .then(response => response.json())
12     .then(data => {
13       var tempValue = data['main']['temp'];
14       var nameValue = data['name'];
15       var descValue = data['weather'][0]['description'];
16       var visibilityValue = data['visibility'];
17       var windValue = data['wind']['speed'];
18
19       main.innerHTML = nameValue;
20       desc.innerHTML = "weather : " + descValue;
21       temp.innerHTML = "temperature(°C) : " + tempValue;
22       visibility.innerHTML = "visibility(metres) :" + visibilityValue;
23       wind.innerHTML = "wind speed(m/s) :" + windValue;
24     })
25   })

```

All web services testing results

1.API current weather



Phase 3

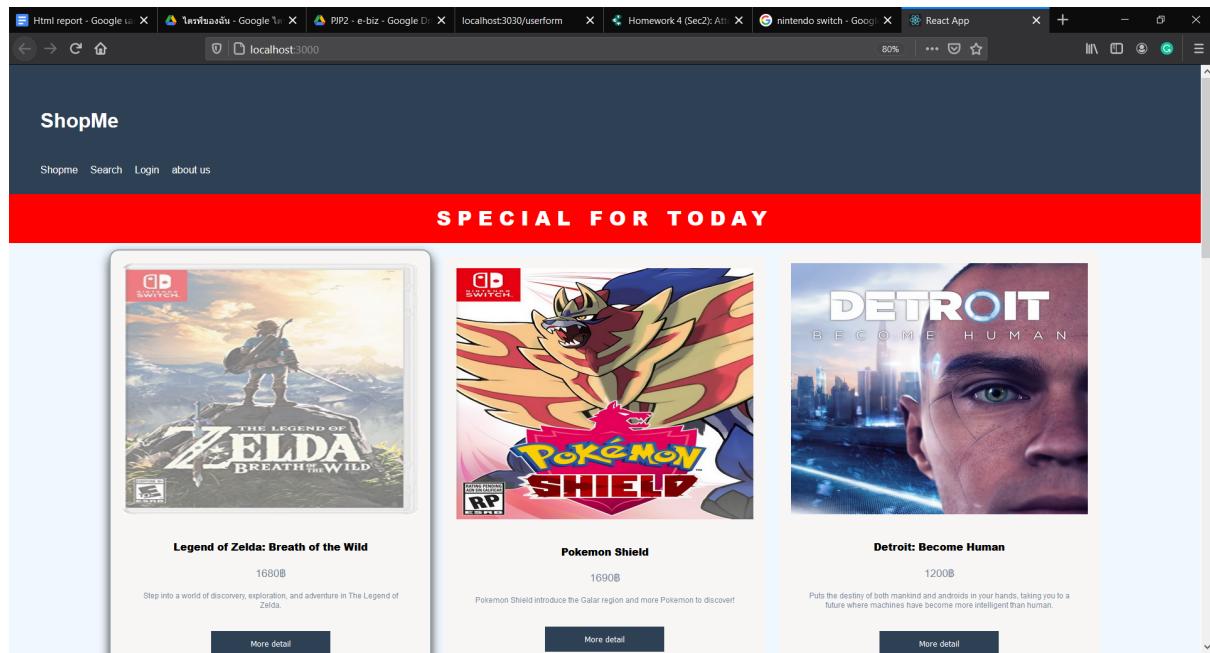
We use React to create the website and we also use projects from Phase1 and Phase 2 to implement our website.

```
MYREACT
  node_modules
  public
  src
    components
      aboutus.js
      admin.js
      home.js
      login.js
      navbar.js
      product.js
      productmana...
      search.js
      usermanage.js
    App.css
    App.js
    App.test.js
    index.css
    index.js
    logo.svg
    reportWebVitals.js
```

As you can see from our files. We have navbar.js. We use navbar.js to link between each page.

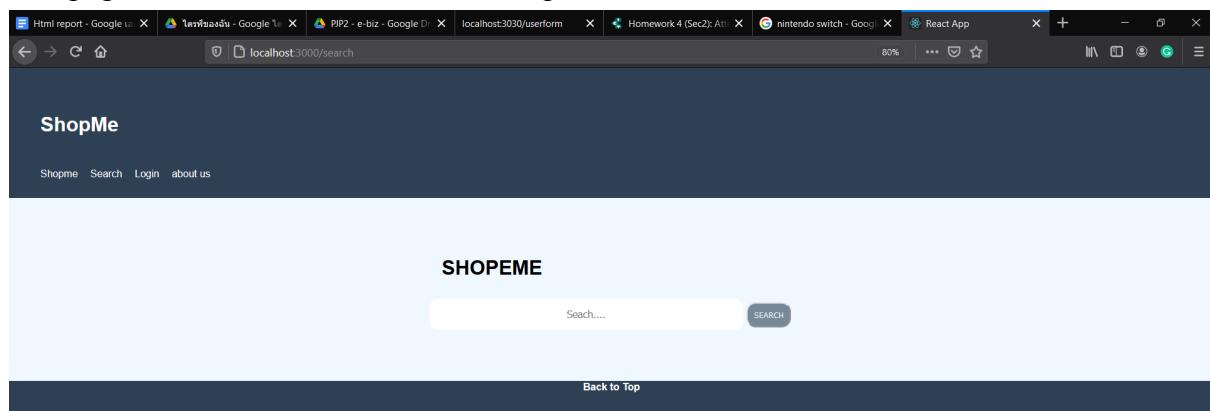
Homepage

This page shows the special offer product but we implement it from react.



Search page

This page is for user to search the desired product.



We do similar things with phase2 but this time we fetch the web service from our project2 to show the data from our DB instead of calling the DB directly.

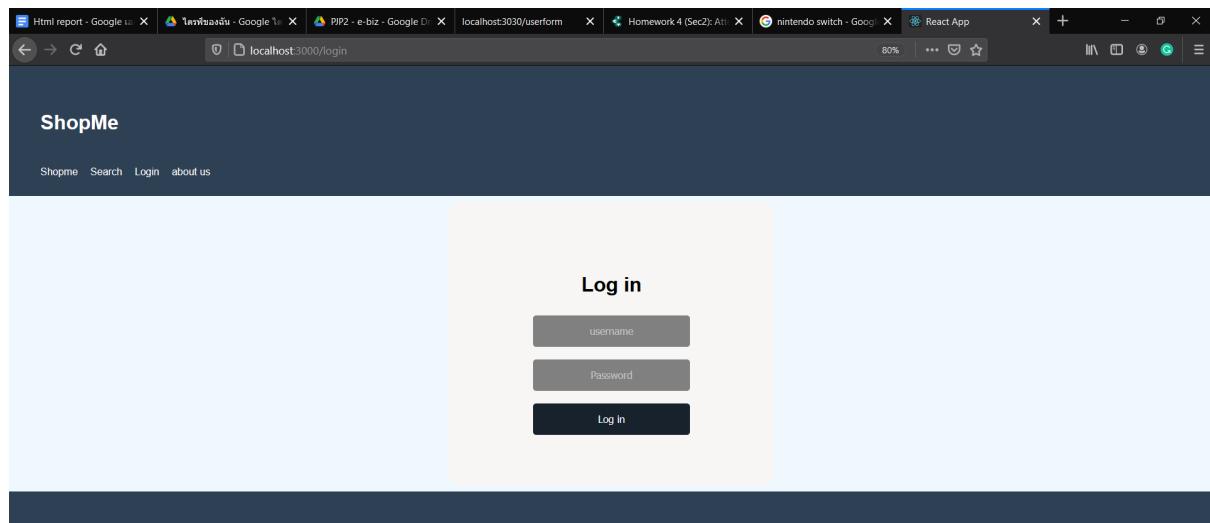
```
let rooturl = "http://localhost:3030/allproducts/" + this.state.menu;
fetch(rooturl,
{
  method: "GET",
  headers: {
    "content-type": "application/json",
    "accept": "application/json"
  },
}
  then(response) => response.json()
```

If we click More detail, it will bring us to the detail of the product page.
 In the product page we also fetch the data from our web service from project2 but in this case we add the specific “key” that is product ID to classify the product page.

```
fetch("http://localhost:3030/allproductsdetail/" + key, {
  method: "GET",
  headers: {
    "content-type": "application/json",
    accept: "application/json",
  },
})  
then((response) => response.json())
```

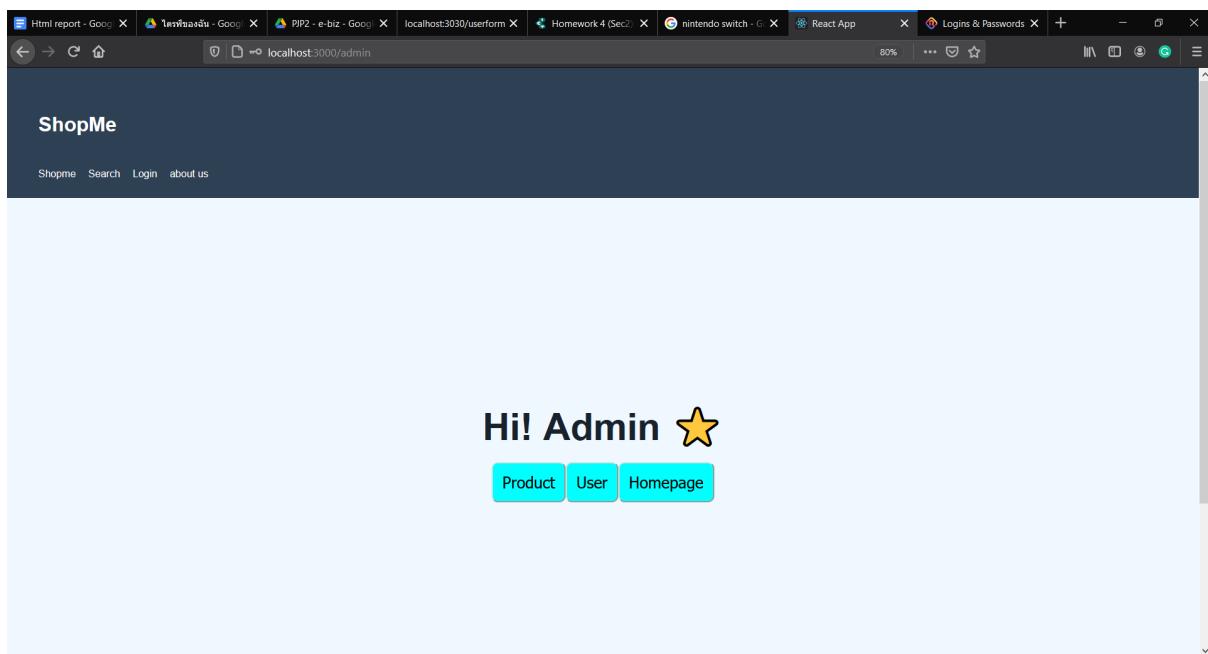
Login Page

This page asks users to login to their account.



In the log in page, we also fetch the service from our service in project 2 to check whether the user and password is correct or not. and also check the role of the user. If the role of the user is admin, it will bring the user to the admin page.

```
let rooturl = "http://localhost:3030/logincheck/" + this.state.username + "&" + this.state.password;
fetch(rooturl,
{
  method: "GET",
  /*body: JSON.stringify({
    username: this.state.username,
    password: this.state.password
  }),*/
  headers: {
    "content-type": "application/json",
    "accept": "application/json"
  },
})
.then((response) => response.json())
.then((data) => {
  console.log(data.data[0])
  if (data.data[0].urole === 'admin')
  {
    window.location.replace('http://localhost:3000/admin');
  }
  else
  {
    window.location.replace('http://localhost:3000/');
  }
})
```



Admin page consists of 2 features, product management and User management.

User management page

A screenshot of the 'User management' page from the ShopMe application. The title bar shows 'localhost:3000/usermanage'. The page has a dark header with 'ShopMe' and navigation links. Below the header is a form titled 'User management' with fields for UserID, Firstname, Lastname, Email, Address, Phone number, Date of birth, Username, Password, and Role. There are buttons for Insert, Update, Delete, Select, and search. At the bottom is a 'Search by ID' section with a UserID field and a search button.

We fetch the web service from our last project. This page allows the admin to manage all users in the database and also can search users by their ID.

Add user

```
let rooturl = "http://localhost:3030/user";
fetch(rooturl,
{
  method: "POST",
  headers: {
    "content-type": "application/json",
    "accept": "application/json"
  },
  body: JSON.stringify(user_data)
})
.then((response) => response.json())
.then((data) => {
  alert("Insert successfully")
})
```

Update user

```
let rooturl = "http://localhost:3030/user";
fetch(rooturl,
{
  method: "PUT",
  headers: {
    "content-type": "application/json",
    "accept": "application/json"
  },
  body: JSON.stringify(user_data)
})
.then((response) => response.json())
.then((data) => {
  alert("Update successfully")
})
```

Delete user

```
let rooturl = "http://localhost:3030/user";
fetch(rooturl,
  {
    method: "DELETE",
    headers: {
      "content-type": "application/json",
      "accept": "application/json"
    },
    body: JSON.stringify(user_data)
})
.then((response) => response.json())
.then((data) => {
  alert("user deleted");
})
```

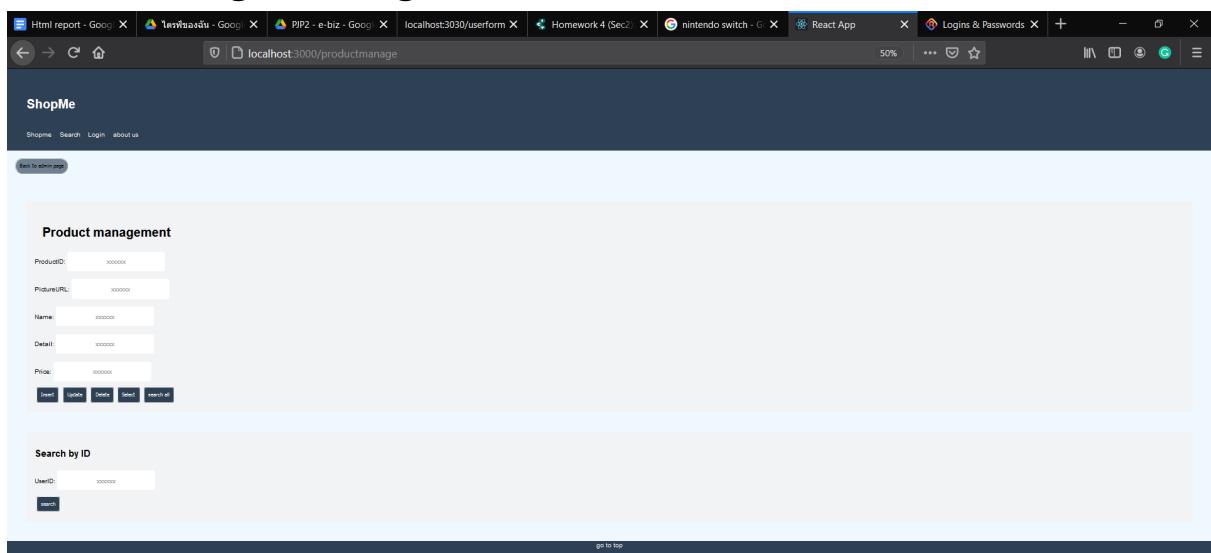
Search user by ID

```
let rooturl = "http://localhost:3030/user/" + this.state.UID;
fetch(rooturl,
  {
    method: "GET",
    headers: {
      "content-type": "application/json",
      "accept": "application/json"
    },
  })
.then((response) => response.json())
.then((data) => {
```

Select all users

```
let rooturl = "http://localhost:3030/users";
fetch(rooturl,
{
  method: "GET",
  headers: {
    "content-type": "application/json",
    "accept": "application/json"
  },
})
.then((response) => response.json())
.then((data) => {
```

Product management Page



In the product management page we also do the same things as the user management page.

Add product

```
let rooturl = "http://localhost:3030/product";
fetch(rooturl,
{
  method: "POST",
  headers: {
    "content-type": "application/json",
    "accept": "application/json"
  },
  body: JSON.stringify( product_data )
})
.then((response) => response.json())
.then((data) => {
  alert("Insert successfully")
```

Update product

```
let rooturl = "http://localhost:3030/product";
fetch(rooturl,
{
  method: "PUT",
  headers: {
    "content-type": "application/json",
    "accept": "application/json"
  },
  body: JSON.stringify( product_data )
})
.then((response) => response.json())
.then((data) => {
  alert("Update successfully")
```

Delete products

```
let rooturl = "http://localhost:3030/product";
fetch(rooturl,
{
  method: "DELETE",
  headers: {
    "content-type": "application/json",
    "accept": "application/json"
  },
  body: JSON.stringify( product_data )
})
.then((response) => response.json())
.then((data) => {
```

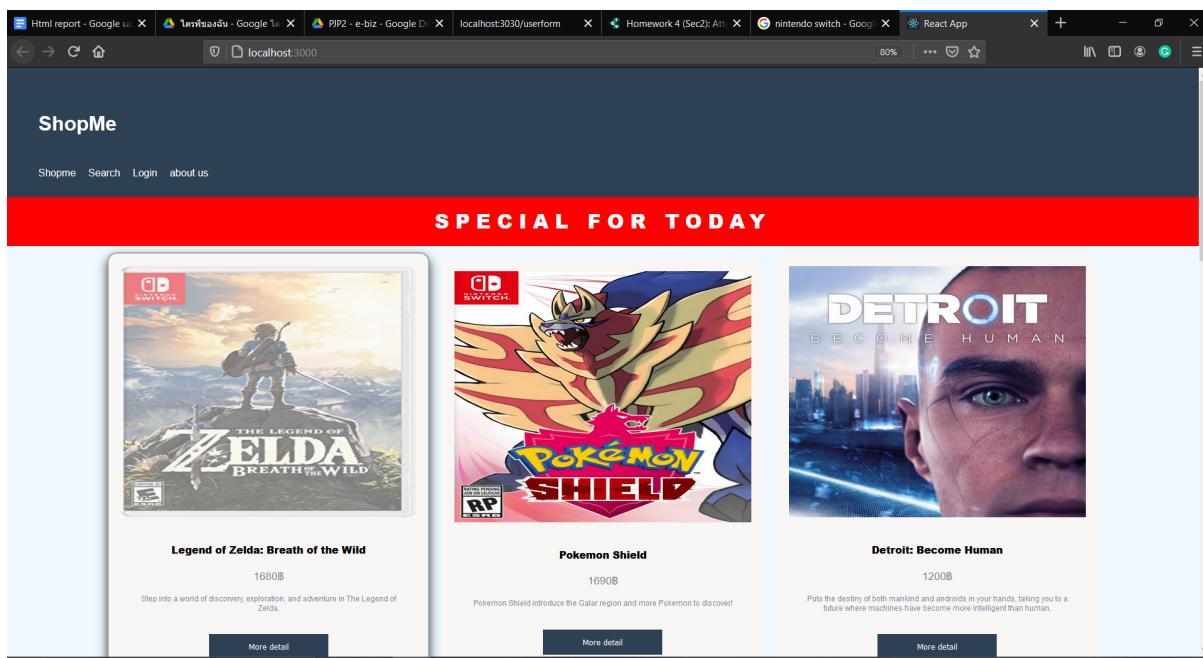
Search product by ID

```
let rooturl = "http://localhost:3030/product/"+this.state.PID;
fetch(rooturl,
{
  method: "GET",
  headers: {
    "content-type": "application/json",
    "accept": "application/json"
  },
})
.then((response) => response.json())
.then((data) => {
  if(data.data!=null){
```

Select all products

```
let rooturl = "http://localhost:3030/products";
fetch(rooturl,
{
  method: "GET",
  headers: {
    "content-type": "application/json",
    "accept": "application/json"
  },
})
.then((response) => response.json())
.then((data) => {
  console.log(data.data);
```

In case that user role is a normal user, it will bring the user to the home page.



About us page

This page shows the information of the developer.

