Mahidol University
Faculty of Information
and Communication Technology

Project Assignment; Phase II

TicketBoo

Business domain: music and entertainment

Members:

Miss Suphavadee	Cheng	6488120
Miss Ponnapassorn	Iamborisut	6488179
Miss Thadeeya	Duangkaew	6488181
Miss Ravikarn	Jarungjitvittawas	6488210

Section 1, Group 10

A Report Submitted in Partial Fulfillment of
the Requirements for

ITCS212 Web Programming

Faculty of Information and Communication Technology
Mahidol University

2022

Table of contents

Project Overview

Introduction:

The purpose of this report is to provide an overview of the web programming project that involves designing and developing a ticketing website. The website will allow the administrator to manage their tickets and admin users.

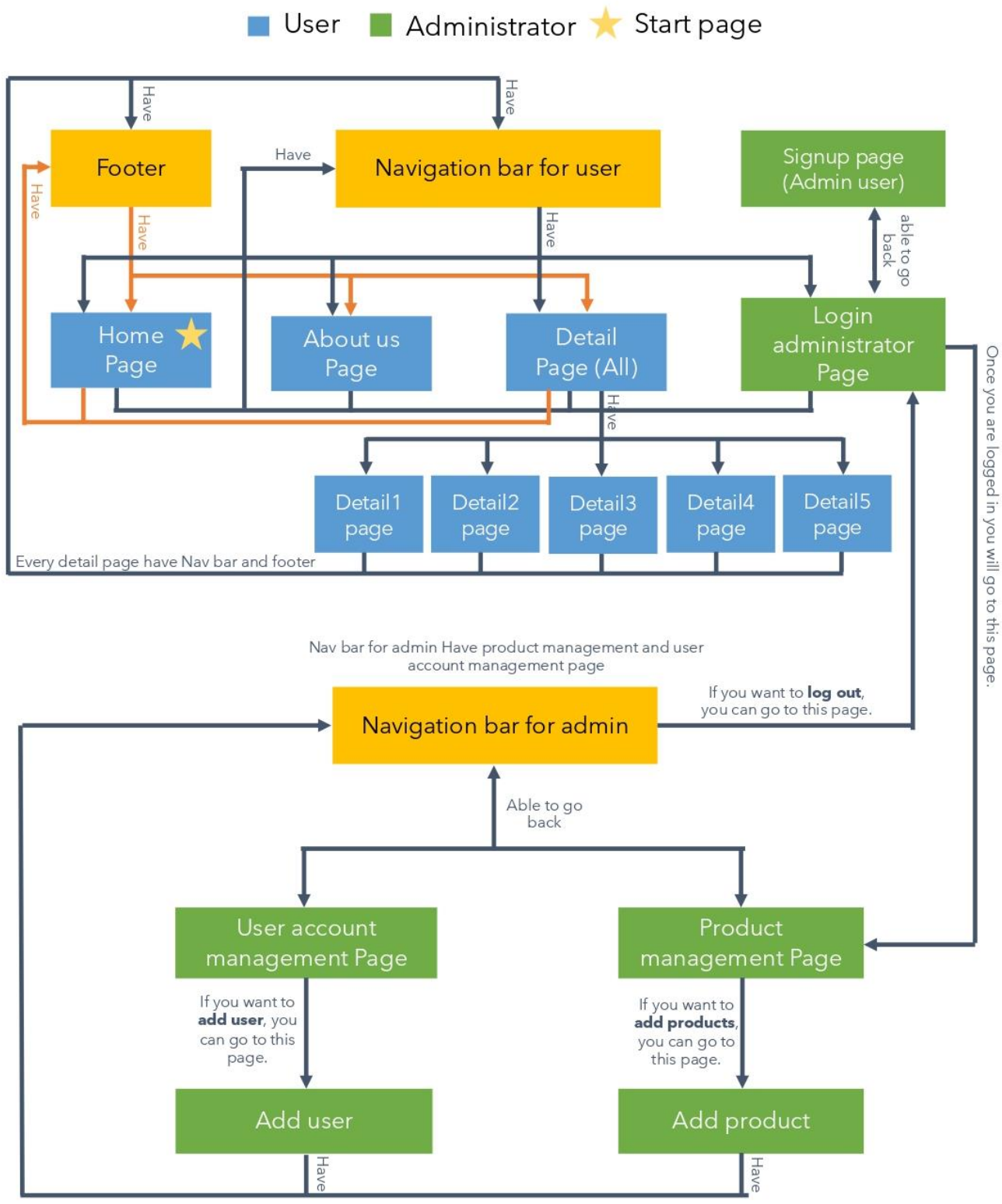
Project Overview:

TicketBoo is a web-based project developed by a sophomore ICT student in the course of web programming. The project aims to provide a user-friendly platform for booking and managing tickets for various events, including movies, concerts, and sports matches.

The website was developed using VSCode as the primary integrated development environment and Node.js as the server-side framework. The project also includes various web technologies such as HTML, CSS, JavaScript, .env, and database management with SQL. Throughout the course, the student has received extensive training from experienced instructors and has acquired the necessary knowledge and skills to develop a full-fledged web application. TicketBoo is an outcome of the student`s learning and showcases their proficiency in web development. The website`s features include a responsive and intuitive user interface that allows users to browse available events, view ticket prices, and make bookings seamlessly. The website also incorporates various security measures, such as user authentication and data encryption, to protect sensitive user information.

In conclusion, TicketBoo is a web-based project that showcases the student's learning and proficiency in web development. The project provides a user-friendly platform for booking and managing tickets for various events, and it incorporates various web technologies and security measures to ensure a smooth and secure user experience.

Navigation Diagram of our Website



Database

1. Administrator information and Administrator login information are stored in the `**admin_info**` table.
2. The tickets are store in `**ticket**` table.

Relational Schema

Admin_info

admin_code	fname	lname	dob	gender	password	login_log	role

Ticket

ticket_code	ticket_title	ticket_img	ticket_desc	ticket_loca	ticket_date	ticket_time	ticket_price

Data Dictionary

Table	Attribute	Contents	Type	Format	Range	Key	FK
Admin_info	admin_code	Admin code	VARCHAR(3)	xxx		PK	
	fname	First name	VARCHAR(100)	xxxxxxxxxxxxx			
	lname	Last name	VARCHAR(100)	xxxxxxxxxxxxx			
	dob	Date of birth	DATE	yyyy-mm-dd			
	gender	Gender	CHAR(1)	x	M,F		
	password	Password	VARCHAR(100)	xxxxxxxxxxxxx			
	login_log	Login time/date	DATETIME	yyyy-mm-dd hh:mm:ss			
	role	Role	VARCHAR(100)	xxxxxxxxxxxxx			
Ticket	ticket_code	Ticket's code	VARCHAR(10)	xxxxxxxxxx		PK	
	ticket_title	Title	VARCHAR(200)	xxxxxxxxxxxxx			
	ticket_img	URL of image	TEXT(50000)	xxxxxxxxxxxxx			
	ticket_desc	Description	VARCHAR(500)	xxxxxxxxxxxxx			
	ticket_loca	Location	VARCHAR(200)	xxxxxxxxxxxxx			
	ticket_date	Date	DATE	yyyy-mm-dd			
	ticket_time	Time	TIME	hh:mm:ss			
	ticket_price	Price	INT	xxxxx			

Table Description

Table	Schema View	Information
Admin_info Admin information who has a role in this website	admin_info Columns admin_code fname lname dob gender password login_log role	**Table:** admin_info **Columns:** admin_code varchar(3) PK fname varchar(100) lname varchar(100) dob date gender char(1) password varchar(100) login_log datetime role varchar(100)
Ticket Tickets of the event that Ticketboo website sells	ticket Columns ticket_code ticket_title ticket_img ticket_desc ticket_loca ticket_date ticket_time ticket_price	**Table:** ticket **Columns:** ticket_code varchar(10) PK ticket_title varchar(200) ticket_img mediumtext ticket_desc varchar(500) ticket_loca varchar(200) ticket_date date ticket_time time ticket_price int

Admin_Info

```
CREATE TABLE `admin_info` (
    `admin_code` VARCHAR(3) PRIMARY KEY,
    `fname` VARCHAR(100) NOT NULL,
    `lname` VARCHAR(100) NOT NULL,
    `dob` DATE NOT NULL,
    `gender` CHAR(1) NOT NULL,
    `password` VARCHAR(100) NOT NULL,
    `login_log` DATETIME NOT NULL,
    `role` VARCHAR(100) NOT NULL
);
```

admin_code	fname	lname	dob	gender	password	login_log	role
001	Wudhichart	Sawangphol	1980-12-11	M	Wudhichart	2023-03-12 12:35:20	Admistrator
002	Jidapa	Kraisangka	1979-09-01	F	Jidapa	2023-03-24 09:46:17	Admistrator
003	Pisit	Praiwattana	1979-04-24	M	Pisit	2023-03-25 23:08:56	Admistrator
120	Suphavadee	Cheng	2002-11-15	F	Suphavadee	2023-04-01 20:29:07	Admistrator
179	Ponnapassorn	Iamborisut	2003-02-24	F	Ponnapassorn	2023-04-02 04:46:19	Admistrator
181	Thadeeya	Duangkaew	2001-08-26	F	Thadeeya	2023-04-02 10:37:58	Admistrator
210	Ravikarn	Jarungjitvittawas	2003-10-02	F	Ravikarn	2023-04-06 16:24:31	Admistrator

Ticket

```
CREATE TABLE `ticket` (
    `ticket_code` VARCHAR(10) PRIMARY KEY,
    `ticket_title` VARCHAR(200) NOT NULL,
    `ticket_img` TEXT(50000) NOT NULL,
    `ticket_desc` VARCHAR(500) NOT NULL,
    `ticket_loca` VARCHAR(200) NOT NULL,
    `ticket_date` DATE NOT NULL,
    `ticket_time` TIME NOT NULL,
    `ticket_price` INT NOT NULL
);
```

ticket_code	ticket_title	ticket_img	ticket_desc	ticket_loca	ticket_date	ticket_time	ticket_price
TK_02	WATERBOMB BA...	/images/WaterBomb-Poster.jpg	Definitely t...	Thunderdome S...	2023-04-13	12:00:00	4600
TK_03	ABOUT DAMN TI...	/images/Ph1-Poster.jpg	IMC Live Gl...	CentralwOrld LIVE	2023-03-19	19:00:00	3800
TK_05	Pelupo Festival ...	/images/Pelupo-Poster.png	Leave your ...	Siam Country Cl...	2023-03-11	15:00:00	4000
TK_04	Arctic Monkeys L...	/homepagePic/card/arcMon.png	The wait is ...	BITEC Bangna	2023-03-09	18:00:00	6000
TK_01	The 1975 Live in...	/homepagePic/card/1975con.jpg	Mangostee...	Impact Arena	2023-04-04	19:00:00	5000

Web application

File: HomePage.html

head

```
<head>
1  <title>Ticket Boo!</title>
2  <link rel="stylesheet" href="/style/HomePage.css">
   <script src="https://kit.fontawesome.com/a076d05399.js"></script>
   <link rel='stylesheet' href='https://use.fontawesome.com/releases/v5.7.0/css/all.css'
     integrity='sha384-lZN37f5QGtY3VHgisS14W3ExzMWZxybE1SJSEsQp9S+oqd12jhcu+A56Ebc1zFSJ' crossorigin='anonymous'>
   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
</head>
```

1. Link to HomePage.css in the folder that is named style.
2. Other external links that use for symbol.

body

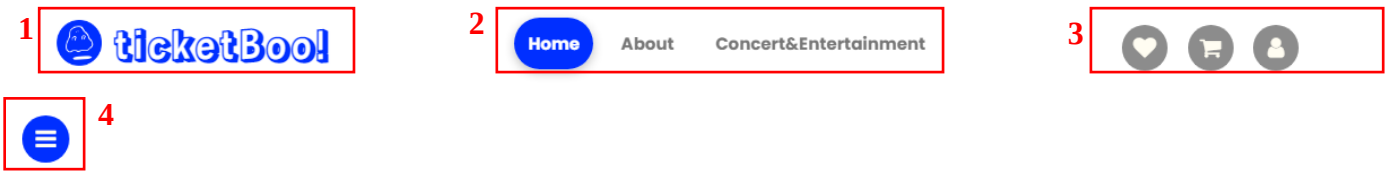
<header>

```
<header>
    1
    <a href="/html/HomePage.html" class="logo"><img src="/homepagePic/logo/TicketBoo_Logo.png"> ticketBoo!</a>
    <!--show the logo in nav bar-->
    <!--show the text button in the middle-->
    2
    <nav class="bar">
        <a class="thispage" href="/html/HomePage.html">Home</a>
        <a href="/html/aboutUs.html">About</a>
        <a href="/html/AllDetails.html">Concert&Entertainment</a>
    </nav>
    <!--show the symbol in the right of the page -->
    3
    <div class="iconNav">
        <a href="#" class="fa fa-heart" id="heartbar"></a>
        <a href="#" class="fa fa-shopping-cart" id="shopbar"></a>
        <a href="/html/Userlogin.html"><i class="fa fa-user" id="acctbar"></i></a>
    </div>
    <!--this will not show yet at fisrt until the size of screen get smaller-->
    4
```

```
    <div class="iconbar" onclick="showBar()">
        <i class="fa fa-bars" id="menubar"></i>
    </div>
</header>
```

<script>

```
/* this use for open and close the bar*/
4.1 let show = 0;
function showBar() {
    /* if show is 0, it will open the bar. and 1 will close the bar*/
    /* we will change the color and change the position of the bar and icon too
    (icon will change in the case of the size of screen change to less than 700px*/
    if (show == 0) {
        show = 1;
        document.getElementsByClassName('bar')[0].style.left = "0px";
        document.getElementsByClassName('iconNav')[0].style.left = "0px";
    } else {
        show = 0;
        document.getElementsByClassName('bar')[0].style.left = "-100%";
        document.getElementsByClassName('iconNav')[0].style.left = "-100%";
    }
}
```



1. When users press the logo, it will link to HomePage.html This will show picture <img> and the text.
2. This is the navigation bar, which is Home, About, and Concert & Entertainment. When users click, it will link to that path.
3. This is also the navigation bar, which is Favorite, Shopping Cart, and Login. In this case Favorite button and Shopping Cart button don't have links.
4. This icon will show when users resize the page.

   4.1 showBar() will be displayed when the size is small. If show is 0 mean it will open the bar and 1 is for close the bar.

** **

```
<div class="slides-container">
    <div class="slide fade">
        <a href="/html/Detail05.html"><img src="/homepagePic/slide/slide1.jpg" style="width:100%"></a>
    </div>
    <div class="slide fade">
        <a href="/html/Detail01.html"><img src="/homepagePic/slide/slide2.jpg" style="width:100%"></a>
    </div>
    <div class="slide fade">
        <a href="/html/Detail03.html"><img src="/homepagePic/slide/slide3.jpg" style="width:100%"></a>
    </div>
    <!--this is button for change the slide-->
    <a class="prev" onclick="changeSlide(-1)">
        <p>&#10094;</p>
    </a>
    <a class="next" onclick="changeSlide(1)">
        <p>&#10095;</p>
    </a>
</div>
```

<script>

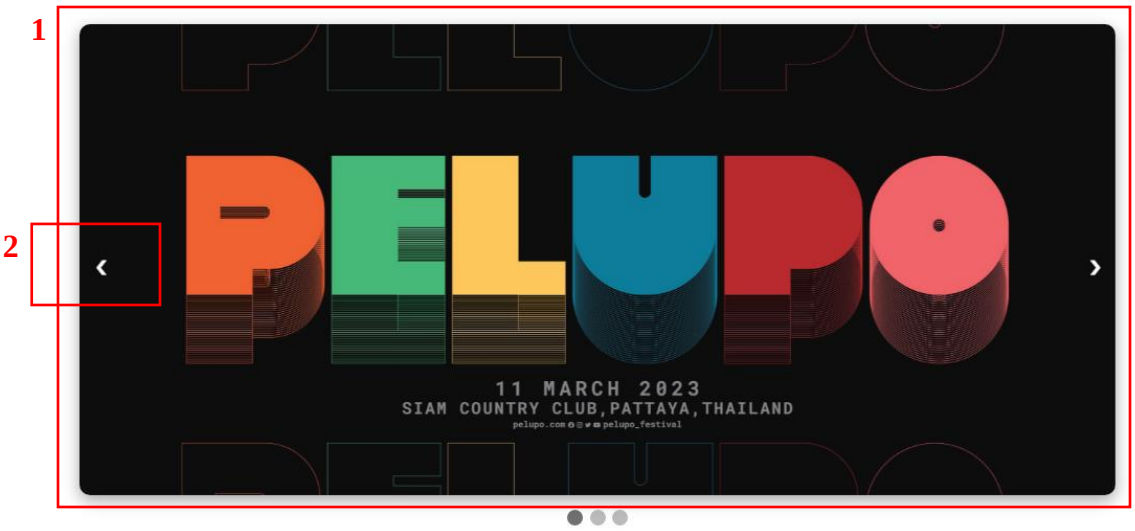
```
/*set the public attribute*/
let slideIndex = 1;

/*call the function*/
showSlidesAuto();
showSlides(slideIndex);

/*this function use to change to next slide*/
/*it use with the button of previous and next for slides*/
function changeSlide(n) {
    slideIndex += n;  /*we will add n */
    showSlides(slideIndex);
}

/*this will set current slide that you click it in dot button*/
function currSlide(n) {
    slideIndex = n;
    showSlides(n);
}
```

2.

```
/*this use for display the slide*/
function showSlides(n) {
    /*first keep the values of slides and dots from class of 'slide' and 'dot'*/
    let slides = document.getElementsByClassName("slide");
    let dots = document.getElementsByClassName("dot");

    /*this will see first that the number that they send
    is larger than the slide lenght (5) or not. If it greater than
    slide length, it will loop back to the first pictire*/
    if (n > slides.length) {
        slideIndex = 1;
    }
    /*But id n is less than 1 that's mean it has to go back to the last
    picture in thie slide. So we have to set slide indec to be slide length.*/
    if (n < 1) {
        slideIndex = slides.length;
    }

    /*this will set every slides to be not show*/
    for (let i = 0; i < slides.length; i++) {
        slides[i].style.display = "none";
    }
    /*this will replace the one that has class of showThis to not have
    this class*/
    for (let i = 0; i < dots.length; i++) {
        dots[i].className = dots[i].className.replace(" showThis", "");
    }

    /*set the new showing picture by calling the one element in array by
    using the index. And call the style.display to change from none to block(show in block).*/
    slides[slideIndex - 1].style.display = "block";
    /*this will set the dots too, so it will change the color of dot by using classname to do this*/
    dots[slideIndex - 1].className += " showThis";
}
```

3

```
/*this is same as showSlides(n) but this add in terms of setTimeout*/
function showSlidesAuto() {

    let i;
    let slides = document.getElementsByClassName("slide");
    let dots = document.getElementsByClassName("dot");
    for (i = 0; i < slides.length; i++) {
        slides[i].style.display = "none";
    }
    slideIndex++;

    /*this will check whether the current index is larger than the lenght
    or not. If it yes, it's mean we have to go back to the first slide*/
    if (slideIndex > slides.length) {
        slideIndex = 1;
    }
    for (i = 0; i < dots.length; i++) {
        dots[i].className = dots[i].className.replace(" showThis", "");
    }

    slides[slideIndex - 1].style.display = "block";
    dots[slideIndex - 1].className += " showThis";

    setTimeout(showSlidesAuto, 5000); // This will call this function in every 5 seconds
}
```



1. This will show the picture and link to the detail page of that concert.
2. This will show the symbol of < and > and it will be placed the most left and right on the picture. This has onclick function by sending +1 for the forward and -1 for the backward.
   2.1 function changeSlide(n): it will be called when user click < and > by using the function onclick. In this case, the value of n will be 1 and -1 for forward and backward. It will add 1 or minus 1 to the slideIndex then sending this value to showSlides(slideIndex) function.
   2.2 It will check first that the slide index which is sent is greater than the amount of slides, it will set the index again to 1. If it is less than 1, it will set to the last slide. Then in for loop, it will use css to show that slide. Lastly, we need to hide the previous slide.
3. showSlidesAuto() function is used for changing the picture every 5 seconds automatically.

** **

1

```
<!--this is the dot bar below the slide and it will show the recent slide with the diffence color to others-->
<div style="text-align:center">
    <!--set the first dot to 'showThis' first-->
    <!--if click the dot it will call the function of currSlide(n) to set new picture-->
    <span class="dot showThis" id="dot1" onclick="currSlide(1)"></span>
    <span class="dot" id="dot2" onclick="currSlide(2)"></span>
    <span class="dot" id="dot3" onclick="currSlide(3)"></span>
    <!-- <span class="dot" id="dot4" onclick="currSlide(4)"></span>
    <span class="dot" id="dot5" onclick="currSlide(5)"></span> -->
</div>
```

1. It will set the class of the first one to be showThis to indicate that this dot will show. It also has a function of onclick which will call currSlide(n). We use currSlide(n) to change the current slide. Each of one will be 1 ,2 ,and 3.

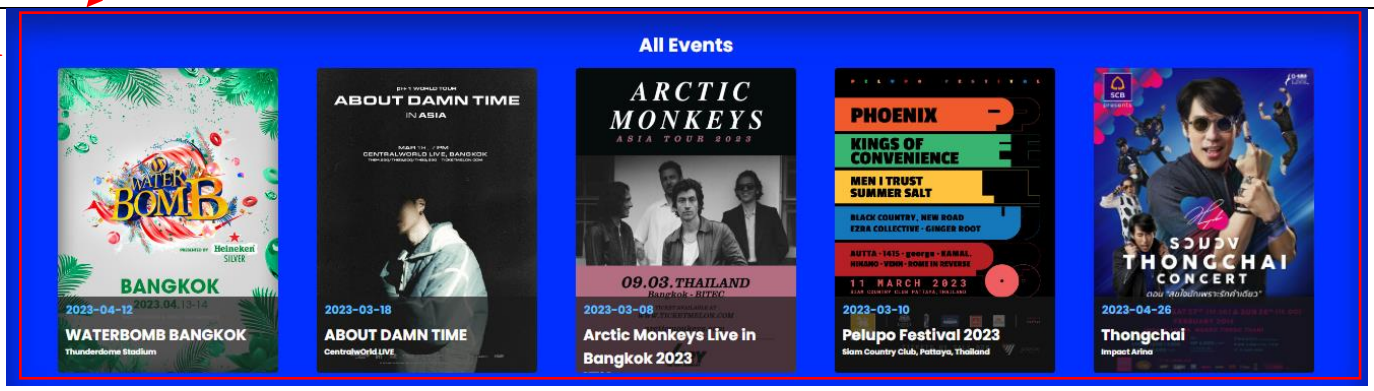
<section> and <div id= "infoPop"

```
<br><br><br><br><br><br><br>
<!--this section use to show the recommend concert-->
<section>
    <div class="cards-container">
        <!--this will show the topic of this section-->
        <h1 class="topicRec">All Events</h1>
        <div class="rowSlide" id='output'>

        </div>
    </div>
</section>

<div id="infoPop" class="wrapper" style="display:none;">
    <p id="pop"></p>
</div>
```

1

2

1



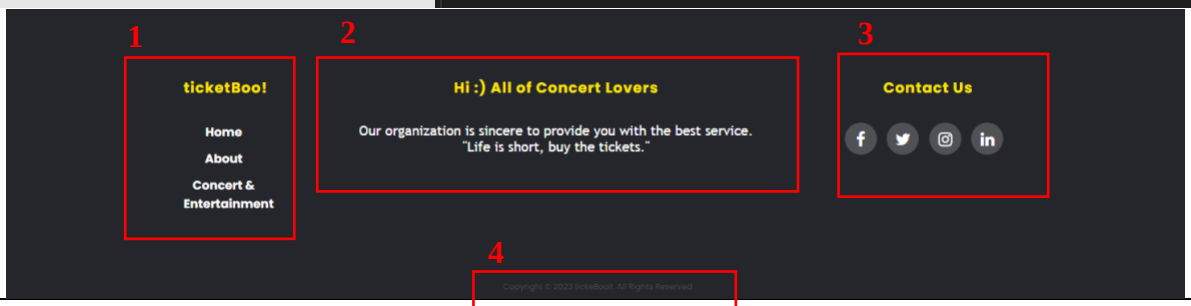
This part will be explained in Task 3 in Phase 2. But to introduce this part, the class of ‘cards-container’ will be shown all of the events that are on our ticketboo website. The part of the class wrapper that will pop up if the user clicks the ticket card First, it will be hidden by displaying none, and it will be displayed as a block after the user clicks the card.

footer

```
<footer class="footer-container">
1   <!--this is first column-->
    <div class="footer-col1">
        <h3>ticketBoo!</h3>
        <a href="/html/HomePage.html" id="move1">
            <h1>Home</h1>
        </a>
        <a href="/html/aboutUs.html" id="move2">
            <h1>About</h1>
        </a>
        <a href="/html/AllDetails.html" id="move3">
            <h1>Concert & Entertainment</h1>
        </a>
    </div>
```

```
2   <!--this is second column-->
    <div class="footer-col2">
        <h3>Hi :) All of Concert Lovers</h3>
        <p>
            Our organization is sincere to provide you with the best service. <br>
            "Life is short, buy the tickets."
        </p>
    </div>
```

```
3   <!--this is third column-->
    <div class="footer-col3">
        <h3>Contact Us</h3>
        <div class="social-icon">
            <a href="javascript:void(0);"><i class="fa fa-facebook"></i></a>
            <a href="javascript:void(0);"><i class="fa fa-twitter"></i></a>
            <a href="javascript:void(0);"><i class="fa fa-instagram"></i></a>
            <a href="javascript:void(0);"><i class="fa fa-linkedin"></i></a>
        </div>


4   </div>
    <div class="cpRight">Copyright &copy 2023 tickeBool!. All Rights Reserved</div>
```



1. This shows the first column that is used to link to another page such as home page, about us, and all details.
2. This second column shows the text about our website content.
3. This third column shows how users can contact us. But in this case, we didn't put any link to each of contact.

File: aboutUs.html

head

```
<head>
    <meta charset="utf-8" />
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
    <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300&display=swap" rel="stylesheet" />
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>About us</title>
    <link rel="stylesheet" href="/style/aboutUs.css" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
</head>
```

1. Name the title of this to be About us.
2. This will link to the css file that is named 'aboutUs.css'. And the other link is for the symbol that we use from external links.

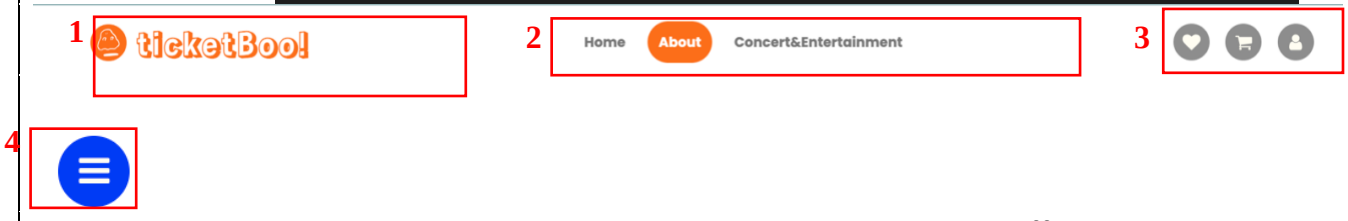
body

<header>

```
<body>
    <header>
1       <a href="/HomePage.html" class="logo"><img src="/logo/TicketBoo_Logo2.png"> ticketBoo!</a>

2       <!--show the text button in the middle-->
        <nav class="bar">
            <a href="/html/HomePage.html">Home</a>
            <a href="/html/aboutUs.html" class="thispage">About</a>
            <a href="/html/AllDetails.html">Concert&Entertainment</a>

        </nav>

3       <!--show the symbol in the right of the page-->
        <div class="iconNav">
            <a href="/html/SearchPage.html"><i class="fa fa-search" id="searchbar"></i></a>
            <a href="#" class="fa fa-heart" id="heartbar"></a>
            <a href="#" class="fa fa-shopping-cart" id="shopbar"></a>
            <a href="/html/Userlogin.html"><i class="fa fa-user" id="acctbar"></i></a>
        </div>

4       <!--this will not show yet at fisrt until the size of screen get smaller-->
        <div class="iconbar" onclick="showBar()">
            <i class="fa fa-bars" id="menubar" ></i>
        </div>
    </header>
```

<script>

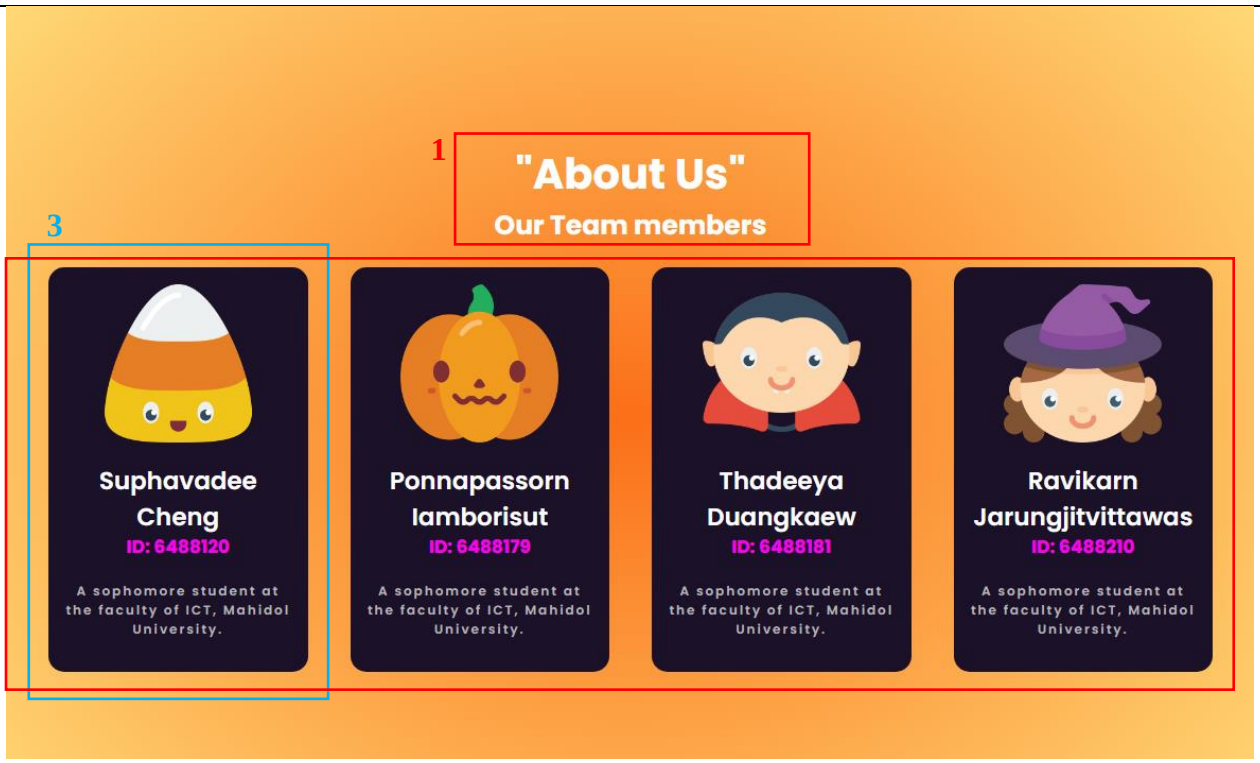
```
/*this use for open and close the bar*/
let show = 0;
function showBar() {
    /*if show is 0, it will open the bar. and 1 will close the bar*/
    /*we will change the color and change the position of the bar and icon too
    (icon will change in the case of the size of screen change to less than 700px*/
    if (show == 0) {
        show = 1;
        document.getElementsByClassName('bar')[0].style.left = "0px";
        document.getElementsByClassName('iconNav')[0].style.left = "0px";
    } else {
        show = 0;
        document.getElementsByClassName('bar')[0].style.left = "-100%";
        document.getElementsByClassName('iconNav')[0].style.left = "-100%";
    }
}
```



This header is the same as the home page. Only the name class in the bar is different as it is this is the about us page. So, this will name the class in the about us link is ‘thispage’ to change the background color to orange.

** **

```
<div class="container">
    <div class="topic">
        <h1>"About Us"</h1>
        <h2>Our Team members</h2>
    </div>
    <div class="ourTeam">
        <div class="memberBox">
            <img src="/aboutUsPic/img/candy-corn.png" alt="candy">
            <h3 class="nameMember">
                Suphavadee Cheng
            </h3>
            <h3 class="number">
                ID: 6488120
            </h3>
            <p class="text-paragraph">
                A sophomore student at the faculty of ICT, Mahidol University.
            </p>
        </div>
        <div class="memberBox">···
        </div>
        <div class="memberBox">···
        </div>
        <div class="memberBox">···
        </div>
    </div>
</div>
```



1. To show what the topic is about.
2. It is a container to show all of the members.
3. For this class of member box will show each of member name, id, and short information about education.

<script>

```
/* fetch to get all of the tickets from the SQL. We fetch the port at 3000 which is the port
    of client side.*/
    fetch('http://localhost:3000/getadmins', {
      method: 'GET',
      headers: {
        'Content-Type': 'application/json'
      }
    })
      .then(response => response.json()) // wait for response
      .then(data => {
        console.log(data)
          document.querySelector("#output").innerHTML = null;  // we point to the output box and ready to do the html
          const jsonData = document.getElementById('output');

          /** we will print out by using innerHTML to show up at the part of the box that has id='output*/
          for (let i = 0; i < data.length; i++) {
```

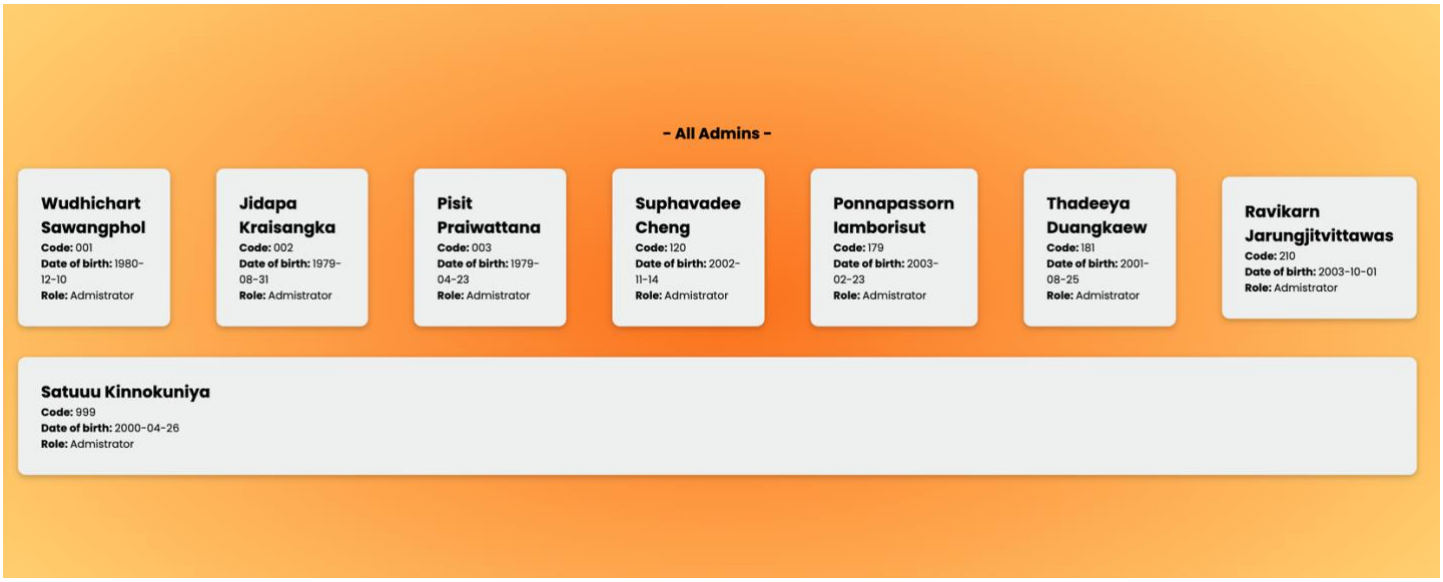
```
                    /** If user click the box it will vall showInfo function and pass the admin_code*/
                    jsonData.innerHTML += `<div class='boxUsers' onclick="showInfo('${data[i].admin_code}')"><h2 class='txt'>` + data[i].fname +
" " + data[i].lname + "</h2>"
                        + "<p class='txt'><b>Code: </b>" + data[i].admin_code + "</p>"
                        + "<p class='txt'><b>Date of birth: </b>" + data[i].dob.substring(0, 10) + "</p>"
                        + "<p><b>Role: </b>" + data[i].role + "</p></div>"
                        /* If user click edit, it will call the function if showEditForm and pass the admin code*/
                     console.log(data[i].admin_code)
                }
            });
/* *This showInfo need to get the information from the server,
         * it receive the admincod and pass it to port of 3000 which specify
         * the adminCode.
        */
        function showInfo(adminCode) {
            console.log(adminCode)
            fetch(`http://localhost:3000/admin_infos/${adminCode}`, {
                method: 'GET',
                headers: {
                    'Content-Type': 'application/json'
                }
            })
                .then(response => response.json())
                .then(data => {                     /* After the response, we get the information now*/
                    console.log(data);
                    let code = data.data.admin_code;
                    let fname = data.data.fname;
                    let lname = data.data.lname;
                    let dob = data.data.dob;
                    let gender = data.data.gender;
                    let pass = data.data.password;
                    let log = data.data.login_log;
                    let role = data.data.role;

                    document.querySelector("#pop").innerHTML = null;  // we will print it here, the part of pop to pop up the box
                    const pop = document.getElementById('pop');
                    // display this pop up box to show the infor
```

```
            document.getElementById('infoPop').style.display = 'block';
            // show all details
            pop.innerHTML += "<h1>" + fname + " " + lname + " [" + code + "] </h1><h3 class='detail'>Date of Birth: " + dob.substring(0, 10)
+ "</h3>";
            pop.innerHTML += "<h3 class='detail'>Gender: " + gender + "</h3>";
            pop.innerHTML += "<h3 class='detail'>Email: " + fname+"."+lname.substring(0,3)+"@gmail.com" + "</h3><h3 class='detail'>Role:
" + role + "</h3>";
            pop.innerHTML += "<div class='pass-link' onclick='closeForm()'><span class='close-symbol'></span></div>";
        })


    }



    // Same as this, this will close the pop up box, to be display as a none again
    function closeForm() {
        const form = document.getElementById('infoPop');
        form.style.display = 'none';


    }
```

</script>

1. This part of the code will be the part that we add, that is, we can see who the admin user is. They are located below our Team members.
2. This section will find a pop-up to view the details of the information.

User Interface:



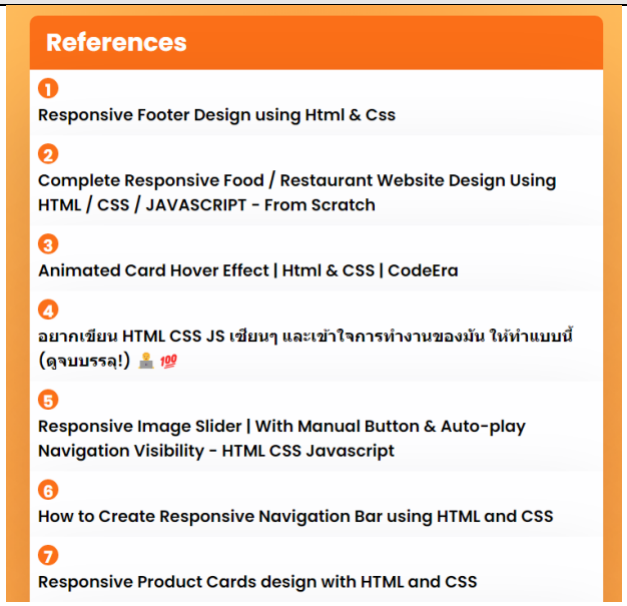
This is all admin

When you click it will show the detail of each admin.

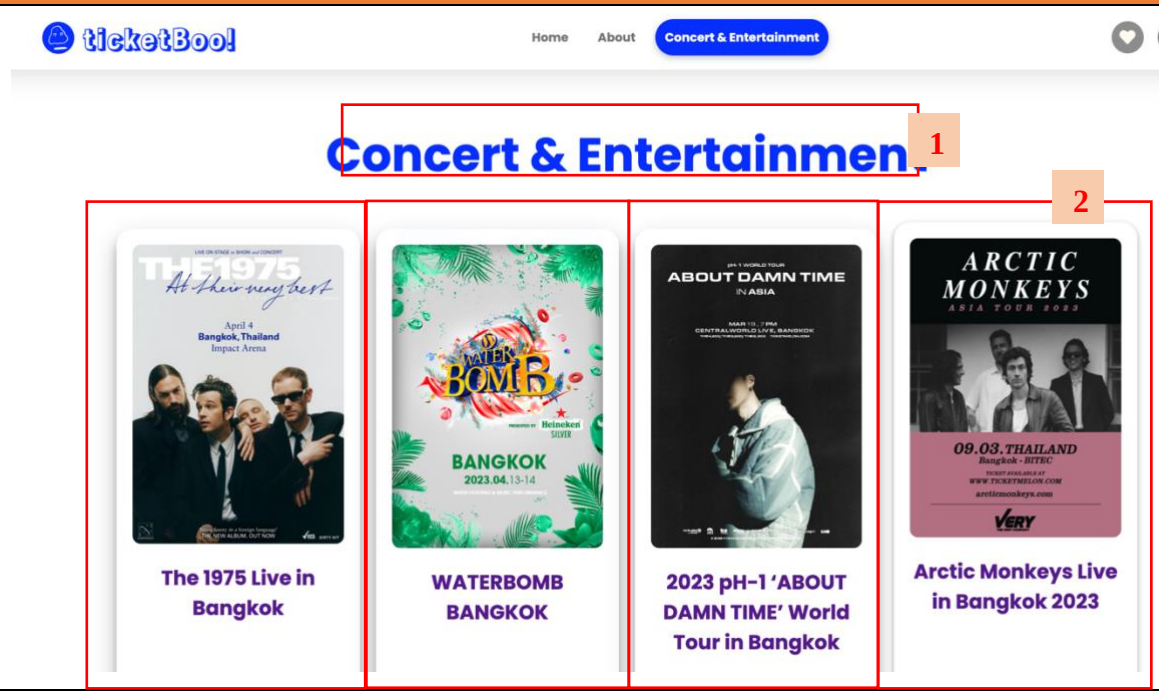
<section class="List">

```
<section class="List">
    <div class="boxOfList">
    <h1>References</h1>
    <ul>
      <li><span>1</span><a href="https://youtu.be/YOb67OKw62s"><h3>Responsive Footer Design using Html & Css</h3></a></li>
      <li><span>2</span><a href="https://youtu.be/MJUssi2c6Ls"><h3>Complete Responsive Food / Restaurant Website Design Using HTML / CSS / JAVASCRIPT - From Scratch</h3></a></li>
      <li><span>3</span><a href="https://youtu.be/t5zFfDdvApE"><h3>Animated Card Hover Effect | Html & CSS | CodeEra</h3></a></li>
```

```
        <li><span>4</span><a href="https://youtu.be/Ei_zX44ItEU"><h3>อยากเขียน HTML CSS JS เขียนๆ และเข้าใจการทำงานของมัน ให้ทำแบบนี้ (ดูจบบรรลุ!)
💯</h3></a></li>
        <li><span>5</span><a href="https://youtu.be/bW8X-tt5AZQ"><h3>Responsive Image Slider | With Manual Button & Auto-play
Navigation Visibility - HTML CSS Javascript</h3></a></li>
        <li><span>6</span><a href="https://youtu.be/oLgtucwjVII"><h3>How to Create Responsive Navigation Bar using HTML and
CSS</h3></a></li>
        <li><span>7</span><a href="https://youtube.com/watch?v=kRs3aTi3pzU&si=EnSIkaIECMiOmarE"><h3>Responsive Product Cards
design with HTML and CSS</h3></a></li>
        <li><span>8</span><a href="https://www.youtube.com/watch?v=RctaFustg5w"><h3>Search Bar Using Html CSS & javascript | CSS
Expandable Search Box</h3></a></li>
        <li><span>9</span><a href="https://youtu.be/H-DvSPRnKWQ"><h3>How To Create Our Team Section Using HTML And CSS |
Responsive HTML CSS Tutorial</h3></a></li>
        <li><span>10</span><a href="https://youtube.com/watch?v=kNO2WkKJzu0&si=EnSIkaIECMiOmarE"><h3>Animated Search Box
using HTML CSS & JavaScript | Elastic Animation on Search Bar</h3></a></li>
        <li><span>11</span><a href="https://www.ticketmelon.com/"><h3>Ticketmelon</h3></a></li>
        <li><span>12</span><a href="https://youtu.be/V8PU_geaCCU"><h3>Animated Login and Registration Form in HTML CSS &
JavaScript</h3></a></li>
      </ul>
    </div>
</section>
```



This is the page that shows references that we used.

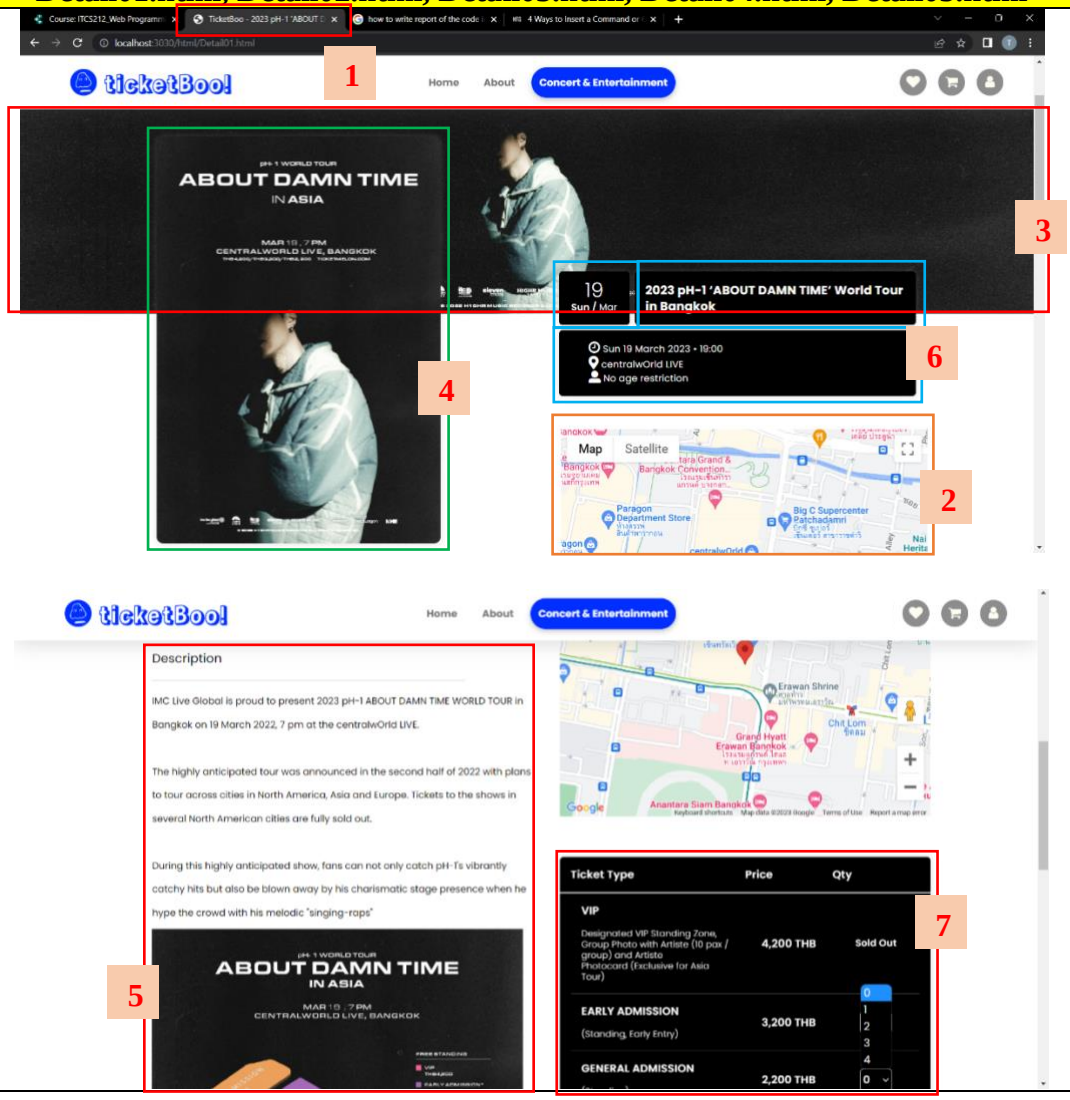
File	AllDetails.html
User Interface	
Code	1 2

```
<div class="contain-box">
        <!-- This class is to display the head content -->
            <h3>Concert & Entertainment</h3>
    </div>

    <!-- class hidden will hide the result until the users click on
    the white search icon at the search bar -->
    <section id="ggg" class="hidden">

        <!-- this class covers all the content of results that is displayed on the page -->
        <div class="gallery">

            <!-- this class covers each box of the results -->
            <div class="content">
                <!-- If this button is clicked, it will show the details of that content
                which are Concert or Entertainment. -->
                <a href="/html/Detail02.html">
                    <img class="detail" src="/images/The1975-Poster.jpg"><br>
                    <h4> The 1975 Live in Bangkok </h4><br>
                    <h7> Tue 4 April 2023 [19:00] </h7>

                    <!-- If this button is clicked, it will show the details of that content
                which are Concert or Entertainment. -->
```

```
            <button class="details"> Detail </button>
        </a>
    </div>
    <div class="content">
        <a href="/html/Detail04.html">
            <img class="detail" src="/images/WaterBomb-Poster.jpg"><br>
            <h4> WATERBOMB BANGKOK </h4><br>
            <h7> 13 April - 14 April 2023 [12:00] </h7>
            <button class="details"> Detail </button>
        </a>
    </div>

    <div class="content">
        <a href="/html/Detail01.html">
            <img class="detail" src="/images/Ph1-Poster.jpg"><br>
            <h4> 2023 pH-1 'ABOUT DAMN TIME' World Tour in Bangkok </h4><br>
            <h7> Sun 19 March 2023 [19:00] </h7>
            <button class="details"> Detail </button>
        </a>
    </div>

    <div class="content">
        <a href="/html/Detail03.html">
            <img class="detail" src="/images/AM-Poster.png"><br>
            <h4> Arctic Monkeys Live in Bangkok 2023 </h4><br>
            <h7> Thu 9 March 2023 [18:00] </h7>
            <button class="details"> Detail </button>
        </a>
    </div>

    <div class="content">
        <a href="/html/Detail05.html">
            <img class="detail" src="/images/Pelupo-Poster.png"><br>
            <h4> Pelupo Festival 2023 </h4><br>
            <h7> Sat 11 March 2023 [15:00] </h7>
            <button class="details"> Detail </button>
        </a>
    </div>

    <div class="content">
        <a href="#">
            <img class="detail" src="/images/Louis-Poster.jpg"><br>
            <h4> LOUIS TOMLINSON FAITH IN THE FUTURE WORLD TOUR 2023 BANGKOK </h4><br>
            <h7> Sat 22 April 2023 [16:00] </h7>
            <button class="details"> Detail </button>
```

```
        </a>
    </div>

    <div class="content">
        <a href="#">
            <img class="detail" src="/images/Harry-Poster.png"><br>
            <h4> HARRY STYLES PRESENTS LOVE ON TOUR 2023 BANGKOK </h4><br>
            <h7> Sat 11 March 2023 [16:00] </h7>
            <button class="details"> Detail </button>
        </a>
    </div>

    <div class="content">
        <a href="#">
            <img class="detail" src="/images/NctDream-Poster.jpg"><br>
            <h4> NCT DREAM TOUR THE DREAM SHOW2 in BANGKOK </h4><br>
            <h7> 10 - 12 March 2023 [18:00] </h7>
            <button class="details"> Detail </button>
        </a>
    </div>

    <div class="content">
        <a href="#">
            <img class="detail" src="/images/Treasure-Poster.png"><br>
            <h4> 2023 TREASURE TOUR [HELLO] IN BANGKOK </h4><br>
            <h7> 31 March - 2 April 2023 [17:00] </h7>
            <button class="details"> Detail </button>
        </a>
    </div>

    <div class="content">
        <a href="#">
            <img class="detail" src="/images/YoungOhm-Poster.png"><br>
            <h4> YOUNGOHM Concert "Sound from Dekwat </h4><br>
            <h7> Mon 25 February 2023 [17:00] </h7>
            <button class="details"> Detail </button>
        </a>
    </div>

    <div class="content">
        <a href="#">
            <img class="detail" src="/images/Wallows-Poster.jpg"><br>
            <h4> WALLOWS TELL ME THAT IT'S OVER TOUR 2023 BANGKOK </h4><br>
            <h7> Mon 20 February 2023 [18:00] </h7>
            <button class="details"> Detail </button>
```

	`` </div> <div class="content"> <img class="detail" src="/images/TheVamps-Poster.png"> <h4> The Vamps Greatest Hits Tour in Bangkok </h4> <h7> Fri 10 February 2023 [19:00] </h7> <button class="details"> Detail </button> </div> </div>```
Explanation	No 1: Display the text as Concert & Entertainment No 2: Display the overall event of every event available on the website. If user click on one event, user will find details of each event. (Events that can be clicked to see details, only of the first row and the second row, the first left-hand event only.)

File	Detail01.html, Detail02.html, Detail03.html, Detail04.html, Detail05.html
User Interface	
Code	`<!DOCTYPE html>` `<html lang="en">` `<head>` `<meta charset="UTF-8">` `<meta http-equiv="X-UA-Compatible" content="IE=edge">` `<meta name="viewport" content="width=device-width, initial-scale=1.0">` `<title>TicketBoo - 2023 pH-1 'ABOUT DAMN TIME' Tour in Bangkok</title>` 1 `<link rel="stylesheet" href="/style/Details.css">` `<script src="https://kit.fontawesome.com/a076d05399.js"></script>` `<link rel='stylesheet' href='https://use.fontawesome.com/releases/v5.7.0/css/all.css'`

```
        integrity='sha384-lZN37f5QGtY3VHgisS14W3ExzMWZxybE1SJSEsQp9S+oqd12jhcu+A56Ebc1zFSJ'
crossorigin='anonymous'>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">

    <script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCheMVFtdq-
toMwD0qa12GHiMvK1W25jbI"></script>
    <script>
        function initMap() {
            var myLatLng = { lat: -34.397, lng: 150.644 };
            var map = new google.maps.Map(document.getElementById('map'), {
                center: { lat: 13.745504247614521, lng: 100.53967390887541 },
                zoom: 16
            });
            var marker = new google.maps.Marker({
                position: { lat: 13.745504247614521, lng: 100.53967390887541 },
                map: map,
                title: 'CentralwOrld Live'
            });
        }
    </script>
</head>

<body onload="initMap()">
```

2

```
<!-- This class to display the banner of the concert -->
    <div class="Banner">
        <img class="banner" src="/images/Ph1-Banner.jpg" alt="ABOUT DAMN TIME">
    </div>
```

3

```
    <!-- This class to display the poster of the concert -->
    <div class="containerLeft">
        <div class="Poster">
            <img class="poster" src="/images/Ph1-Poster.jpg" alt="ABOUT DAMN TIME">
        </div>
    </div>
```

4

```
    <!-- Concert details -->

    <div class="detail-container"> <!-- This class is create for contain the details -->
        <div class="Description">
            <div class="Box">
                <div class="Font deco">
                    <h3>Description</h3>
```

5

```
                <hr class="title-underline">
                <div class="font-view">
                    <p>IMC Live Global is proud to present 2023 pH-1 ABOUT DAMN TIME WORLD TOUR in</p>
                    <p>Bangkok on 19 March 2022, 7 pm at the centralwOrld LIVE.</p>
                    <p>
                        <br>
                    </p>
                    <p>The highly anticipated tour was announced in the second half of 2022 with plans</p>
                    <p>to tour across cities in North America, Asia and Europe. Tickets to the shows in</p>
                    <p>several North American cities are fully sold out.</p>
                    <p>
                        <br>
                    </p>
                    <p>During this highly anticipated show, fans can not only catch pH-1's vibrantly</p>
                    <p>catchy hits but also be blown away by his charismatic stage presence when he</p>
                    <p>hype the crowd with his melodic "singing-raps"</p>
                    <p>
                        <img src="/images/Ph1-Zone.jpg" class="fr-fic fr-dib" alt="ABOUT DAMN TIME">
                    </p>
                    <p>Tickets are available at THB2,200 for General Admission, THB3,200 for General</p>
                    <p>Admission with early entry and THB 4,200 for VIP tickets. VIP ticket holders will</p>
                    <p>have a designated VIP Standing Zone, group photo opportunity with Artiste (10</p>
                    <p>pax / group) and Artiste Photocard exclusive for the Asia tour.  Hurry and grab</p>
                    <p>your tickets now to catch pH-1 live!</p>
                </div>
            </div>
        </div>
    </div>
```

5

```
    <!-- Date, Concert's name, Location -->

    <div class="Buy">
        <div class="First-line"> <!-- Date, Concert's name, -->
            <div class="Event date">
                <span class="day">19</span>
                <br>
                <span>
                    <span class="bold">
                        Sun
                        /
                    </span>
                    Mar
                </span>
            </div>
```

6

```
<div class="Event name">
                    <h3>2023 pH-1 'ABOUT DAMN TIME' World Tour in Bangkok</h3>
                </div>
            </div>

            <div class="Second-line">
                <div class="Event info">
                    <ul>
                        <li class="date-time">
                            <i class="fa fa-clock-o" id="clock"></i>
                            <span>
                                Sun 19 March 2023 • 19:00
                                <br>
                            </span>
                        </li>
                        <li class="location">
                            <i class="fas fa-map-marker-alt" id="pin"></i>
                            <span>centralwOrld LIVE</span>
                        </li>
                        <li class="age">
                            <i class="fas fa-user-alt" id="human"></i>
                            No age restriction
                        </li>
                    </ul>
                </div>
            </div>
        </div>

        <!-- Google maps API -->
        <div id="map" style="height: 505px; width: 530px; position: fixed;"></div>

        <!-- Buy Ticket -->

        <div id="TicketBox" class="ticket">
            <div class="Event nonePadding"> <!-- The buying ticket zone -->
                <div class="type"> <!-- Ticket type head, Price head, Qty head -->
                    <div class="text">
                        <div class="Booking head ticket-type">Ticket Type</div>
                        <div class="Booking head price">Price</div>
                        <div class="Booking head qty">Qty</div>
                    </div>
```

6

2

7

```
        </div>

        <div id="0" class="VIP"> <!-- VIP ticket type, which is shown as sold out -->
            <div class="VIPtext">
                <div class="tic body ticket-type">
                    VIP
                    <span class="ticket-description">
                        Designated VIP Standing Zone, Group Photo with Artiste (10 pax / group) and Artiste
                        Photocard (Exclusive for Asia Tour)
                    </span>
                </div>

                <div class="tic body price center">
                    <div class="pricing">
                        4,200
                        <span>THB</span>
                    </div>
                </div>

                <div class="tic body qty center">
                    <p class="status">Sold Out</p>
                </div>
            </div>
        </div>

        <div id="1" class="early"> <!-- Type of tickets that are not sold out -->
            <div class="earlytext"> <!-- and can choose to buy up to 4 tickets -->
                <div class="tic body ticket-type">
                    EARLY ADMISSION
                    <span class="ticket-description">
                        (Standing, Early Entry)
                    </span>
                </div>

                <div class="tic body price center">
                    <div class="pricing">
                        3,200
                        <span>THB</span>
                    </div>
                </div>

                <div class="tic body qty center">
                    <section id="selected" name="four"> <!-- Select the quantity of the ticket uo to 4-->
                        <select>
                            <option value="0"> 0 </option>
```

7

7

```
                        <option value="1">1</option>
                        <option value="2">2</option>
                        <option value="3">3</option>
                        <option value="4">4</option>
                    </select>
                </section>
            </div>
        </div>
    </div>

    <div id="2" class="general">
        <div class="generaltext">
            <div class="tic body ticket-type">
                GENERAL ADMISSION
                <span class="ticket-description">
                    (Standing)
                </span>
            </div>

            <div class="tic body price center">
                <div class="pricing">
                    2,200
                    <span>THB</span>
                </div>
            </div>

            <div class="tic body qty center">
                <section id="selected" name="four">
                    <select>
                        <option value="0"> 0 </option>
                        <option value="1">1</option>
                        <option value="2">2</option>
                        <option value="3">3</option>
                        <option value="4">4</option>
                    </select>
                </section>
            </div>
        </div>
    </div>

    <hr class="root"> <!-- Break line between the detail price and buy button-->
    <div class="buyTicketArea">
        <div class="buyTicketButton">
            <div class="Button">
                <div data-radium="true"> <!-- manage inline style on react elements-->
```

7

7

	<button id="BooButton-buyticket" class="BooButton">Buy Tickets</button> </body>
Explanation	No 1: The name of an event as displayed on its product page on the Ticket Boo No 2: Create initMap() function to use the google.maps.Map and google.maps.Marker objects to call map and markers on the div with id=”map”. Tag used to include a JavaScript file in an HTML document. Show a map of the event's location. By using the Google maps JavaScript API No 3: Displays an image banner on a web page No 4: Displays an image poster on the left-hand side of a web page No 5: Show the detail of the description of the event and show the image of event zone as well No 6: Show details of the date, event name, event location on the right side No 7: The part where the user will be able to click to buy the product. If the tickets are sold out, it will show `Sold Out`, but if the ticket are not sold out, the user can select up to 4 tickets to buy.

File: UserLogin.html

head

```
<head>
    <meta charset="utf-8" />
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
    <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300&display=swap" rel="stylesheet" />
    <link rel='stylesheet' href='https://use.fontawesome.com/releases/v5.7.0/css/all.css'
      integrity='sha384-lZN37f5QGtY3VHgisS14W3ExzMWZxybE1SJSEsQp9S+oqd12jhcu+A56Ebc1zFSJ' crossorigin='anonymous'>
    <title>Login and Signup</title>
    <link rel="stylesheet" href="/style/Loginpage.css" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
</head>
```

Link to loginpage.css and link to other link to get the symbol

body

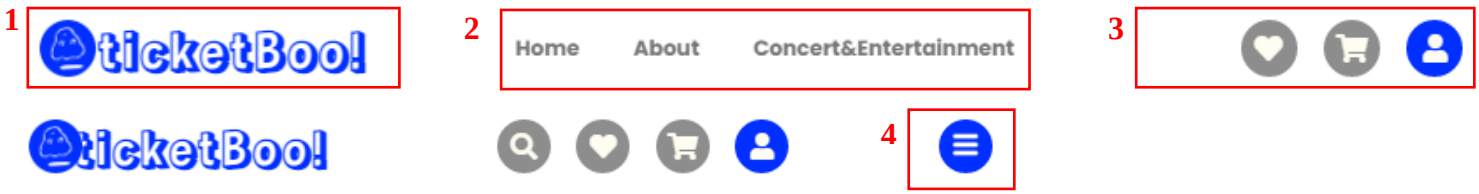
<header>

```
<header>
    <a href="/html/HomePage.html" class="logo"><img src="/homepagePic/logo/TicketBoo_Logo.png"> ticketBoo!</a> <!--show the logo in nav bar-->

    <!--show the text button in the middle-->
    <nav class="bar">
        <a href="/html/HomePage.html">Home</a>
        <a href="/html/aboutUs.html">About</a>
        <a href="/html/AllDetails.html">Concert&Entertainment</a>
    </nav>
    <!--show the symbol in the right of the page-->
    <div class="iconNav">
        <a href="#" class="fa fa-heart" id="heartbar"></a>
        <a href="#" class="fa fa-shopping-cart" id="shopbar"></a>
        <a class="thispage" href="/html/Userlogin.html"><i class="fa fa-user" id="acctbar"></i></a>
```

```
    </div>
    <!--this will not show yet at fisrt until the size of screen get smaller-->
    <div class="iconbar" onclick="showBar()">
        <i class="fa fa-bars" id="menubar" ></i>
    </div>
</header>
```

<script>

```
/* this use for open and close the bar*/
    let show = 0;
    function showBar() {
        /* if show is 0, it will open the bar. and 1 will close the bar*/
        /* we will change the color and change the position of the bar and icon too
        (icon will change in the case of the size of screen change to less than 700px*/
        if (show == 0) {
            show = 1;
            document.getElementsByClassName('bar')[0].style.left = "0px";
            document.getElementsByClassName('iconNav')[0].style.left = "0px";
        } else {
            show = 0;
            document.getElementsByClassName('bar')[0].style.left = "-100%";
            document.getElementsByClassName('iconNav')[0].style.left = "-100%";
        }
    }
```

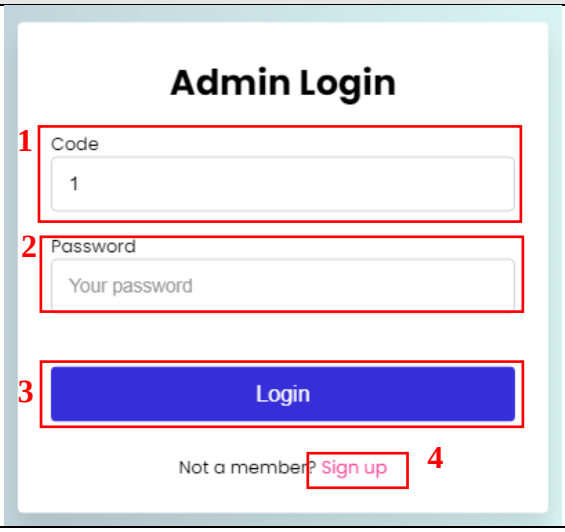


This header is the same as the home page. Only the name class in the bar is different as it is this is the user login page. So, this will name the class in the user login link is ‘thispage’ to change the background color to orange.

** **

```
<div class="wrapper">
    <div class="title">
        <h1>Admin Login</h1>
    </div>
    <div class="form-inner">
        <!-- Form for user to sign up -->
```

```
<form action="/login-submit" method="POST" class="login" >
  <div class="field">
    <label for="code" >Code</label>
    <input type="text" placeholder="Your code"  name="code"/>
  </div>
  <br>
  <div class="field">
    <label for="password" >Password</label>
    <input type="password" placeholder="Your password"  name="password"/>
  </div>
  <br>
  <div class="field btn">
    <div class="btn-layer"></div>
    <input type="submit" value="Login" />
  </div>
  <div class="signup-link">
    Not a member?
    <a href="/html/SignupAdmin.html">Sign up</a>
  </div>
</form>
</div>
</div>
```



If user submit the form it will go to path '/login-submit'

1. Box of text for placing the code of user
2. Box of password for placing the password

3. This is a box for click login to send it to check that do this account is exist or not.
4. If click the sign up, it will show signup admin page.

File: SignupAdmin.html

head

```
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
    <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300&display=swap" rel="stylesheet" />
    <link rel='stylesheet' href='https://use.fontawesome.com/releases/v5.7.0/css/all.css'
      integrity='sha384-lZN37f5QGtY3VHgisS14W3ExzMWZxybE1SJSEsQp9S+oqd12jhcu+A56Ebc1zFSJ' crossorigin='anonymous'>
    <link rel="stylesheet" href="/style/Loginpage.css" />
    <title>Register Admin</title>
</head>
```

Link toLoginpage.css and link to other link to get the symbol

body

** **

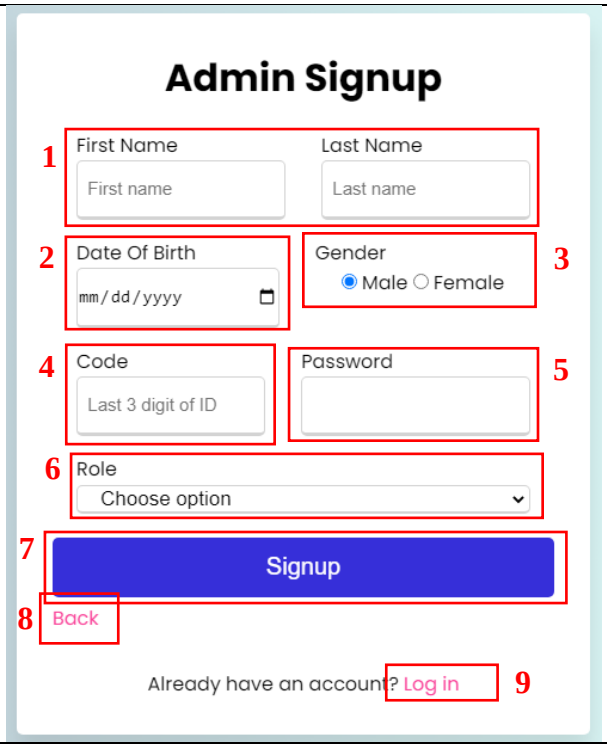
```
<div class="wrapper">
    <h1 class="title">Admin Signup</h1>
    <div class="form inner">
        <form class="signup" action="/signup-submit" method="POST" >
            <div class="row">
                <div class="col box">
                    <label for="fname" class="clabel">First Name</label>
                    <br>
                    <input type="text" class="cinput" name="fname" id="fname" placeholder="  First name">
                </div>
                <div class="col box">
                    <label for="lname" class="clabel">Last Name</label>
                    <br>
                    <input type="text" class="cinput" name="lname" id="lname" placeholder="  Last name">
                </div>
            </div>

            <div class="row">
                <div class="col box">
                    <label for="birthdate" class="clabel">Date Of Birth</label>
                    <br>
                    <input type="date" class="cinput" name="dob" id="birthdate">
                </div>
                <div class="col">

                    <label for="gender" class="clabel">Gender</label>
                    <div class="checkbox">

                        <input type="radio" checked="checked" name="gender" id="gender" value="M">
                        <label class="radio-container">Male</label>

                        <input type="radio" name="gender" id="gender" value="F">
                        <label class="radio-container">Female</label>
                    </div>

                </div>
            </div>
```

1

2

3

```
        <div class="row">
4           <div class="col box">
                <label for="code" class="clabel">Code</label>
                <br>
                <input type="text" class="cinput" name="code" id="code" placeholder="  Last 3 digit of ID">
            </div>
5           <div class="col box">
                <label for="password" class="clabel">Password</label>
                <br>
                <input type="password" class="cinput" name="password" id="password">
            </div>
        </div>

        <div class="lastcol">
6           <label for="role" class="clabel">Role</label>
            <br>
            <select name="role" id="role" class="selectrole">
                <option disabled="disabled" selected="selected">Choose option</option>
                <option>Administrator</option>
            </select>
        </div>
7       <div class="field btn">
            <div class="btn-layer"></div>
            <input type="submit" value="Signup" />
        </div>
8       <div class="pass-link">
            <a href="/html/Userlogin.html">
                Back
            </a>
        </div>
9       <div class="signup-link">
            Already have an account?
            <a href="/html/Userlogin.html">Log in</a>
        </div>
    </form>
  </div>
</div>
```

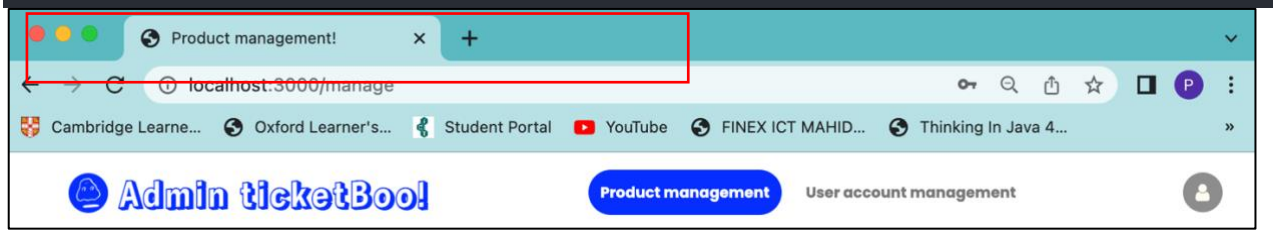


If the user submits their information, it will send the information to the SQL to add more accounts by going to the path of '/signup-submit'. All of the information that is in this form will be passed.

1. This box is the type of text to fill in the first name and last name.
2. This box is the type of date to fill in the date.
3. This box uses radio to choose the gender.
4. This box is the type of text to fill in the code.
5. This box is the type of password so when user fill in the password, it will be hidden.
6. This is the box for selecting the role.
7. This is the signup button with the type of submit.
8. Press the back button will link to Userlogin.html page.
9. Press the Login button will link to Userlogin.html page.

File	Product-management.html
Code and user interface	

```
//Head part that tell the name of this html and to link with css or script.
<head>
    <title>Product management!</title>
    <link rel="stylesheet" href="/style/management.css">
    <script src="https://kit.fontawesome.com/a076d05399.js"></script>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
</head>
```



<Header part>

```
<header>
        <a href="/html/HomePage.html" class="logo"><img src="/homepagePic/logo/TicketBoo_Logo.png"> Admin
ticketBoo!</a>
        <!--show the logo in nav bar-->

        <!--show the text button in the middle-->
        <nav class="bar">
            <a class="thispage" href="/html/Product-management.html">Product management</a>
            <a href="/html/User-account-management.html">User account management</a>

        </nav>

        <!--show the symbol in the right of the page-->
        <div class="iconNav">
            <a href="/html/Userlogin.html"><i class="fa fa-user" id="acctbar"></i></a>
        </div>

        <!--this will not show yet at fisrt until the size of screen get smaller-->
        <div class="iconbar" onclick="showBar()">
            <i class="fa fa-bars" id="menubar"></i>
        </div>
```

```
</header>
```

<script>

```
let show = 0;
    function showBar() {
        /* if show is 0, it will open the bar. and 1 will close the bar*/
        /* we will change the color and change the position of the bar and icon too
        (icon will change in the case of the size of screen change to less than 700px*/
        if (show == 0) {
            show = 1;
            document.getElementsByClassName('bar')[0].style.left = "0px";
            document.getElementsByClassName('iconNav')[0].style.left = "0px";
        } else {
            show = 0;
            document.getElementsByClassName('bar')[0].style.left = "-100%";
            document.getElementsByClassName('iconNav')[0].style.left = "-100%";
        }
    }
```

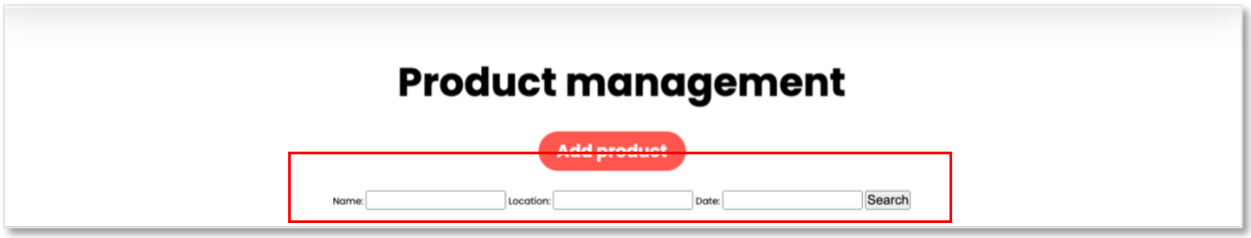
</script>

<html and css part>

This part is built a form to add product and search:

```
<!--this section use to show the overall concert-->
    <div class="title-page">
        <h1>Product management</h1>
    </div>
    <!-- This contain-box was created to move the class of customize to the center. -->
    <div class="contain-box">
        <!-- This class is to store data for adding a new products -->
        <div class="customize">
            <a href="/product">
                <h3>Add product</h3>
            </a>
        </div>
```

```
</div>
<div class="contain-box">
    <form type="normal" action="/search-product-submit" method="POST" id="search-form">
        <label>Name:</label>
        <input type="text" name="name" id="name"></input>
        <label>Location:</label>
        <input type="text" name="location" id="loca"></input>
        <label>Date:</label>
        <input type="text" name="date" id="date"></input>
        <input type="submit" value="Search"></input>
    </form>
</div>
```



<script>

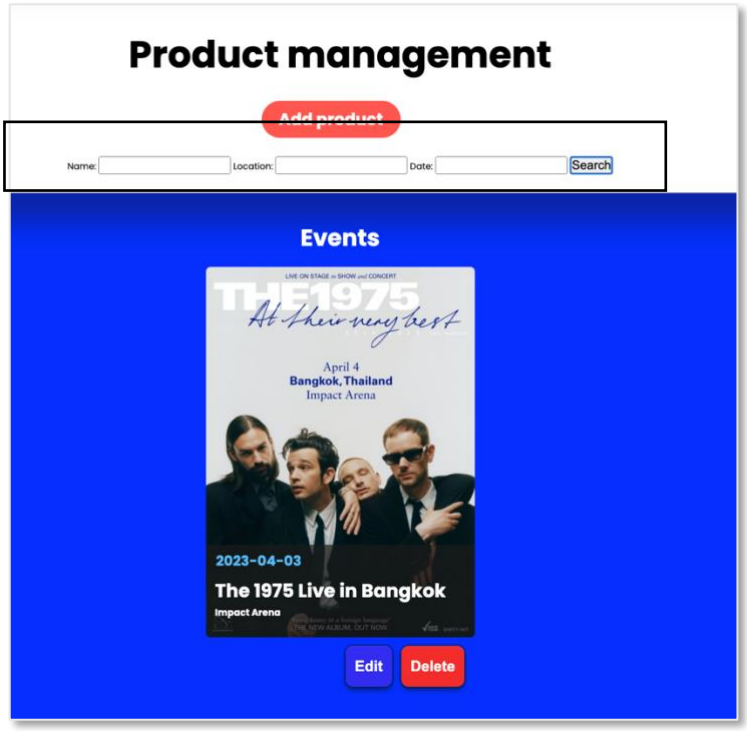
This part is about search by name, location, and date that we have to fetch API from port = 3030 that connect with database.

```
/* This Code will be used when user click submit the form of search the product, it will
        keep the value from the input that has id of 'name', 'loca', and ;'date'.*/
    const searchForm = document.getElementById('search-form');
    searchForm.addEventListener('submit', event => {
        event.preventDefault();
        // console.log("here")
        const name = document.getElementById('name').value;
        const location = document.getElementById('loca').value;
        const date = document.getElementById('date').value;

        /* fetch the data from this port*/
        fetch('http://localhost:3000/search-product-submit', {
            method: 'POST',
            headers: {
```

```
                'Content-Type': 'application/json'
            },
            body: JSON.stringify({ name, location, date })  /* this send the name, location, and date to port of 3000*/
        })
            .then(response => response.json())
            .then(data => {
                document.querySelector("#output").innerHTML = null;
                const jsonData = document.getElementById('output');  // we will post all of the admin in this id
                for (let i = 0; i < data.length; i++) {              // this will show the ticket card and it can edit and delete
                    jsonData.innerHTML +=  `<div class='card' id='${data[i].ticket_code}'
onclick="showInfo('${data[i].ticket_code}')"">`
                        + "<img src='" + data[i].ticket_img + "' class='card-img' />"
                        + "<div class='info'>"
                        + "<h3 class='card-date'>" + data[i].ticket_date.substring(0, 10) + "</h3>"
                        + "<h1 class='card-name'>" + data[i].ticket_title + "</h1>"
                        + "<h3 class='card-loca'>" + data[i].ticket_loca + "</h3></div></div>"
                        + "<div class='button-container'>"
                        + `<button class='button-edit' role='button' data-code="${data[i].ticket_code}"
onclick="showEditForm('${data[i].ticket_code}')" id='edit'>Edit</button>`
                        + "<button class='button-del' role='button' data-code='" + data[i].ticket_code + "'
id='delete'>Delete</button>"
                        + "</div>";
                    console.log(" onclick='showEditForm(" + data[i].ticket_code + ")'")
                }
```

</script>

This is a user interface of this part.



This part for edit Product.

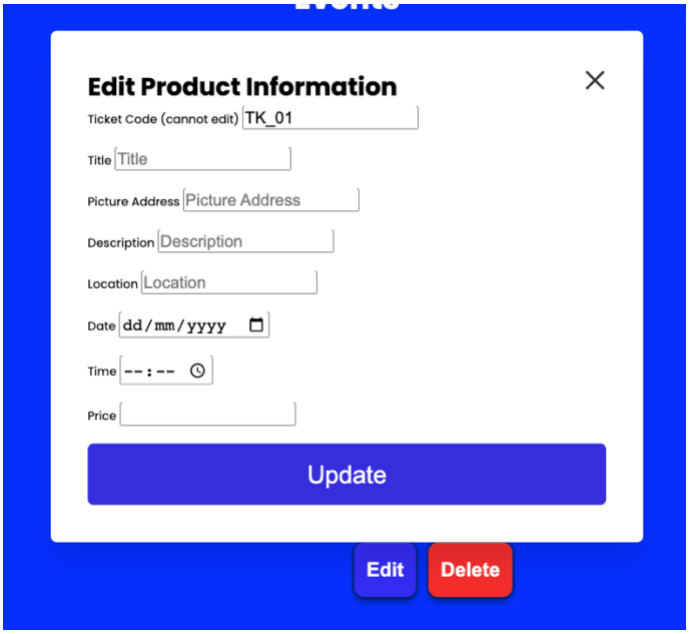
1. You have to fill in every box to update the product.
2. When you click edit the `Ticket_code` will not change because it will let you know which ticket you are editing.

```
<div class="wrapper" id="editForm" style="display:none;">
    <!-- Form for user to edit -->
    <form class="product" action="/editproduct" method="POST">
      <!--send all of this input to this path /editproduct-->
      <h1 class="topic">Edit Product Information</h1>
      <!--Class field is to hold all the value from the input.-->
      <div class="field">
        <label for="ticcode">Ticket Code (cannot edit)</label>
        <input type="text" id="ticcode" name="t_code" placeholder="TK_xx" readonly />
      </div>
      <br>
      <div class="field">
        <label for="titletxt">Title</label>
        <input type="text" id="titletxt" name="t_title" placeholder="Title" required/>
```

```
</div>
<br>
<div class="field">
    <label for="Pic">Picture Address</label>
    <input type="text" id="Pic" name="t_img" title="Picture Address" placeholder="Picture Address" required/>
</div>
<br>
<div class="field">
    <label for="destxt">Description</label>
    <input type="text" id="destxt" name="t_desc" placeholder="Description" required/>
</div>
<br />
<div class="field">
    <label for="locatxt">Location</label>
    <input type="text" id="locatxt" name="t_loca" placeholder="Location" required/>
</div>
<br />
<div class="field">
    <label for="dateCon">Date</label>
    <input type="date" id="dateCon" name="t_date" required/>
</div>
<br />
<div class="field">
    <label for="timeCon">Time</label>
    <input type="time" id="timeCon" name="t_time" required/>
</div>
<br />
<div class="field">
    <label for="priceCon">Price</label>
    <input type="number" id="priceCon" name="t_price" title="number" required/>
</div>
<br>

<div class="total">
    <div class="pass-link" onclick="closeForm()"> <!--If click this close button, it will close the form-->
        <span class="close-symbol"></span>
    </div>
```

```
            <div class="field btn">
                <div class="btn-layer" id="ad"></div>
                <input type="submit" value="Update" />
            </div>
        </div>

    </form>
  </div>
<!-- This will show the pop up box when user click the ticket card.-->
    <div id="infoPop" class="wrapper" style="display:none;">
        <p id="pop"></p>
    </div>
```

This is User interface of this code:



<script>

```
/* *This function is to pop up the box of edit form
     * we also set the value of input of the ticket code
     * so user cannot modify this part. and show it by
     * using the style.display
    */
    function showEditForm(Code) {
        const form = document.getElementById('editForm');
```

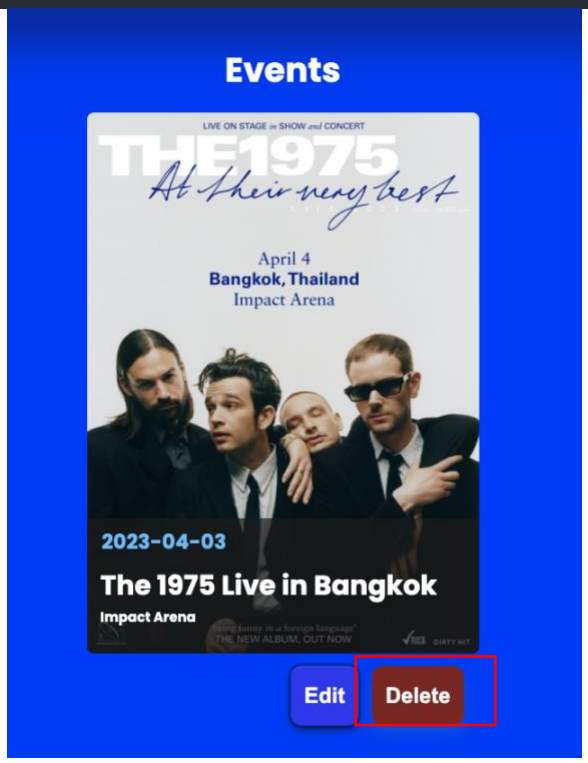
```
        const codeInput = document.getElementById('ticcode');
        codeInput.value = Code;
        form.style.display = 'block'; // Show the form
    }
/* *Function of closeform is make the pop up box
     * to disappear again by useing style to close this
    */
    function closeForm() {
        const form = document.getElementById('editForm');
        form.style.display = 'none';


    }
```

</script>

This part is for deleting each ticket. When you want to delete, it will send the information to frontend.js or port 3000 and it will delete the information.

```
// If click the delete button, it will come to this code
            document.addEventListener('click', function (event) {
                // The button that has the class name 'button-del is a delete button'
                if (event.target.classList.contains('button-del')) {
                    const ticketCode = event.target.dataset.code;
                    console.log('Ticket Code to be deleted: ' + ticketCode);
            // fetch from this port in the method of delete and send the t_code to
            // use it for delete the information in the SQL
                    fetch(`http://localhost:3000/deleteProduct/`, {
                        method: 'DELETE',
                        headers: {
                            'Content-Type': 'application/json'
                        },
                        body: JSON.stringify({ t_code: ticketCode })
                    })
                        .then(response => response.json())
                        .then(data => {
                            console.log(data.error)
                            if (data.error === false) {
```

```
            // If error==false that mean it's work
                    window.location.reload(true);
                // it will reload the page to update the value
                    }
                })
                .catch(error => console.error(error));
            }
        });
```

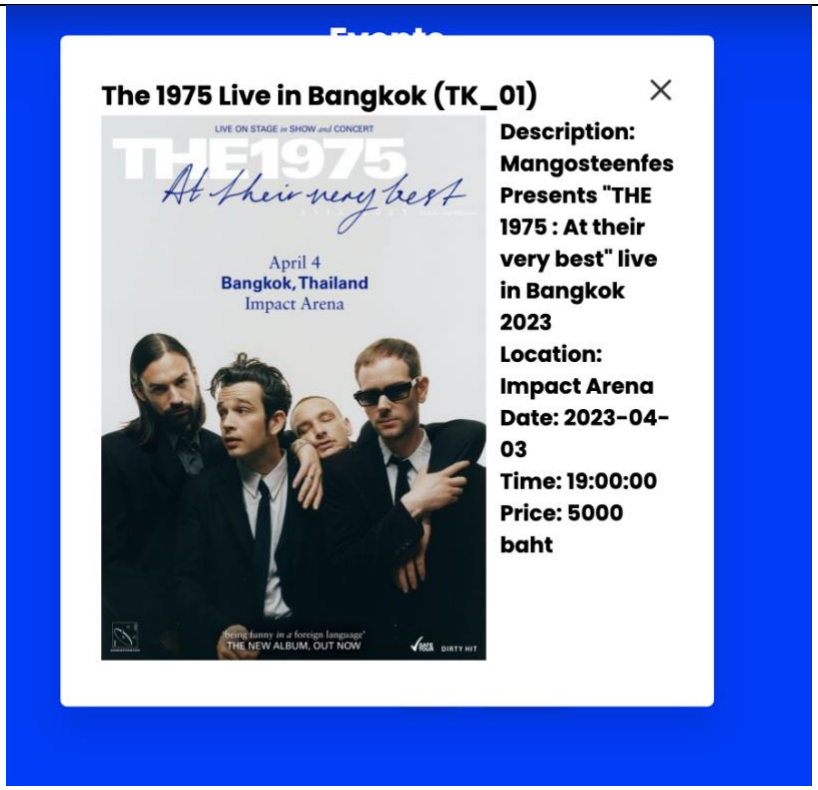


This part is to **view** each tickets that contain their information. To get the ticket information, we should fetch from the frontend.js or port 3000 by ticket_code.

```
/* this function will be called when user click the ticket card. It will fetch to port of 3000
    and useing the ticketCode to get the information of that card*/
    function showInfo(ticketCode) {
     //console.log(ticketCode)
     fetch(`http://localhost:3000/getticket/${ticketCode}`, {
        method: 'GET',
        headers: {
            'Content-Type': 'application/json'
        }
     })
```

```
            .then(response => response.json())
            .then(data => {                    // we get all of data of that card
                console.log(data);
                let code = data.ticket_code;
                let title = data.ticket_title;
                let img = data.ticket_img;
                let desc = data.ticket_desc;
                let loca = data.ticket_loca;
                let date = data.ticket_date;
                let time = data.ticket_time;
                let price = data.ticket_price;
                document.querySelector("#infoPop").innerHTML = null;  // add more html to this id
                const pop = document.getElementById('infoPop');
                pop.style.display = 'block';                    // at first it doesn't show so we need to set to play again


                pop.innerHTML += "<h1>"+title+" "+"("+code+") </h1>"
                    + "<div class='box-container'><div class='box-left'><img src='"+img+"' class='picpop' id='picpop'
style='height: 400px; width: 290;'/></div>"
                    + "<div class='box-right'><h3 class='detail'>Description: "+desc+"</h3>"
                    + "<h3 class='detail'>Location: "+loca+"</h3>"
                    + "<h3 class='detail'>Date: "+date.substring(0,10)+"</h3>"
                    + "<h3 class='detail'>Time: "+time+"</h3>"
                    + "<h3 class='detail'>Price: "+price+" baht</h3></div></div>"
                    + "<div class='pass-link' onclick='closeFormPop()'><span class='close-symbol'></span></div>";
            })
```

This is a user interface:

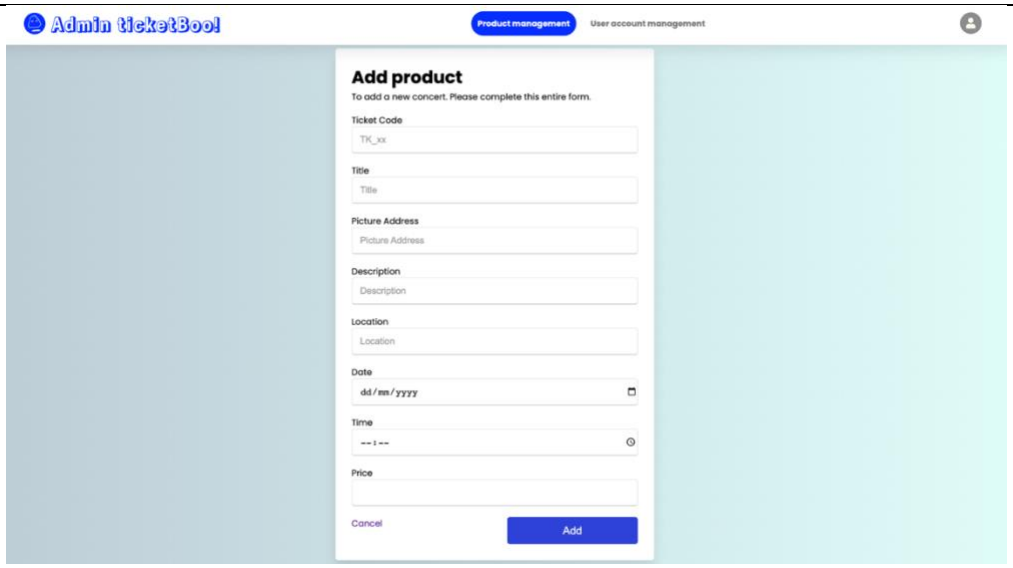
	 <script> `// this will close the pop up box which is the box that has the detail of that ticket` `function closeFormPop() {` `const form = document.getElementById('infoPop');` `form.style.display = 'none';` `}` This code is to close the pop-up that pop out. </script>

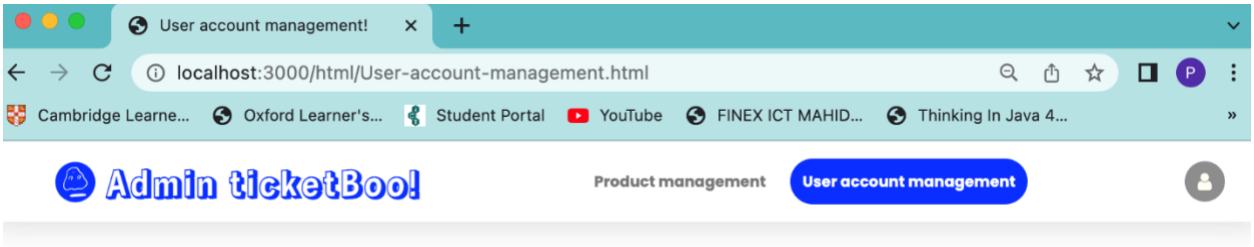
File	product.html
Code and User-Interface	This part is html part that we create a form to let user entire the form of add new product. [For the nav bar we use as the same code] This class is to keep everything in one box. `<div class="wrapper">` `<!-- This is a section to let people who have forgotten their password know what to enter. -->`

```
        <div class="head-text">
            <h1>Add product</h1>
            <p>To add a new concert. Please complete this entire form.</p>
        </div>
        <!--this will keep every box form-->
        <div class="form-inner">
            <!-- Form for user to sign up -->
            <form class="product" action="/addProduct" method="POST">
                <!--Class field is to hold all the value from the input.-->
                <div class="field">
                    <label for="ticcode">Ticket Code</label>
                    <input type="text" id="ticcode" name = "t_code" placeholder="TK_xx" required/>
                </div>
                <br>
                <div class="field">
                    <label for="titletxt">Title</label>
                    <input type="text" id="titletxt" name = "t_title" placeholder="Title" required/>
                </div>
                <br>
                <div class="field">
                    <label for="Pic">Picture Address</label>
                    <input type="text" id="Pic" name="t_img" title="Picture Address" placeholder="Picture Address"
required/>
                </div>
                <br>
                <div class="field">
                    <label for="destxt">Description</label>
                    <input type="text" id="destxt" name = "t_desc" placeholder="Description" required/>
                </div>
                <br />
                <div class="field">
                    <label for="locatxt">Location</label>
                    <input type="text" id="locatxt" name = "t_loca" placeholder="Location" required/>
                </div>
                <br />
                <div class="field">
                    <label for="dateCon">Date</label>
```

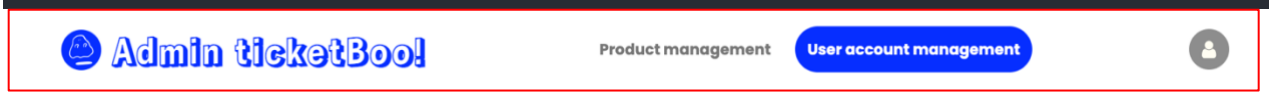
```
            <input type="date" id="dateCon"  name = "t_date" required/>
        </div>
        <br />
        <div class="field">
            <label for="timeCon">Time</label>
            <input type="time" id="timeCon" name = "t_time" required/>
        </div>
        <br />
        <div class="field">
            <label for="priceCon">Price</label>
            <input type="number" id="priceCon" name = "t_price" title="number" required/>
        </div>
        <br>

        <div class="total">
            <div class="field btn">
                <a href="/html/Product-management.html">Cancel</a>
            </div>
            <div class="field btn">
                <div class="btn-layer" id="ad"></div>
                <input type="submit" value="Add" />
            </div>
        </div>

    </form>
  </div>
</div>
```

This is User-Interface of this produce.html:



File	User-management.html
Code and User-Interface	First, head these HTML tags provide the necessary resources for styling the web page and adding iconography to it. <head> `<head>` `  <title>User account management!</title>` `  <link rel="stylesheet" href="/style/management.css">` `  <script src="https://kit.fontawesome.com/a076d05399.js"></script>` `  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">` `</head>` </head>  <header> `<header>` `  <a href="/html/HomePage.html" class="logo"><img src="/homepagePic/logo/TicketBoo_Logo.png" alt=""> Admin`

```
        ticketBoo!</a>
    <!--show the logo in nav bar-->

    <!--show the text button in the middle-->
    <nav class="bar">
        <a href="/html/Product-management.html">Product management</a>
        <a class="thispage" href="#">User account management</a>

    </nav>

    <!--show the symbol in the right of the page-->
    <div class="iconNav">
        <a href="/html/Userlogin.html">
            <i class="fa fa-user" id="acctbar"></i></a>
    </div>

    <!--this will not show yet at fisrt until the size of screen get smaller-->
    <div class="iconbar" onclick="showBar()">
        <i class="fa fa-bars" id="menubar"></i>
    </div>

</header>
```

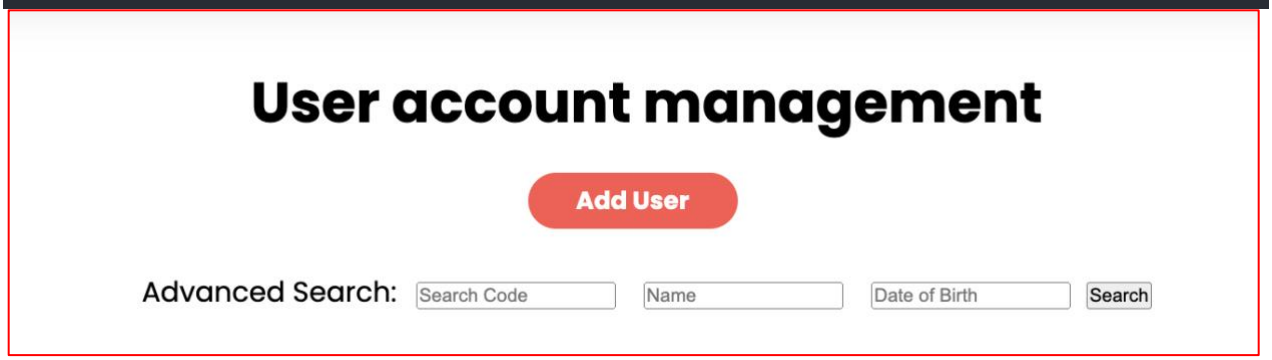


</header>

This part is html part that will have the name and search box already created, will retrieve the value in SQL.

```
<!--this section use to show the recommend concert-->
  <div class="title-page">
    <h1>User account management</h1>
  </div>
  <div class="contain-box">
    <!-- This class is to store data for adding a new products -->
    <div class="customize">
        <a href="/user">
```

```
            <h3>Add User</h3>
        </a>
    </div>
</div>
<form type="normal" class="searchUser" action="/search-user-submit" method="POST" id="search-form">
    <label class="topic">Advanced Search: </label>
    <div class="eachsearch">
        <input id="code" type="search" placeholder="Search Code" list="list" name="code">
    </div>
    <div class="eachsearch">
        <input id="name" type="search" placeholder="Name" list="list" name="name">
    </div>
    <div class="eachsearch">
        <input id="date" type="search" placeholder="Date of Birth" list="list" name="date">
    </div>
    <input type="submit" value="Search"></input>
</form>
```



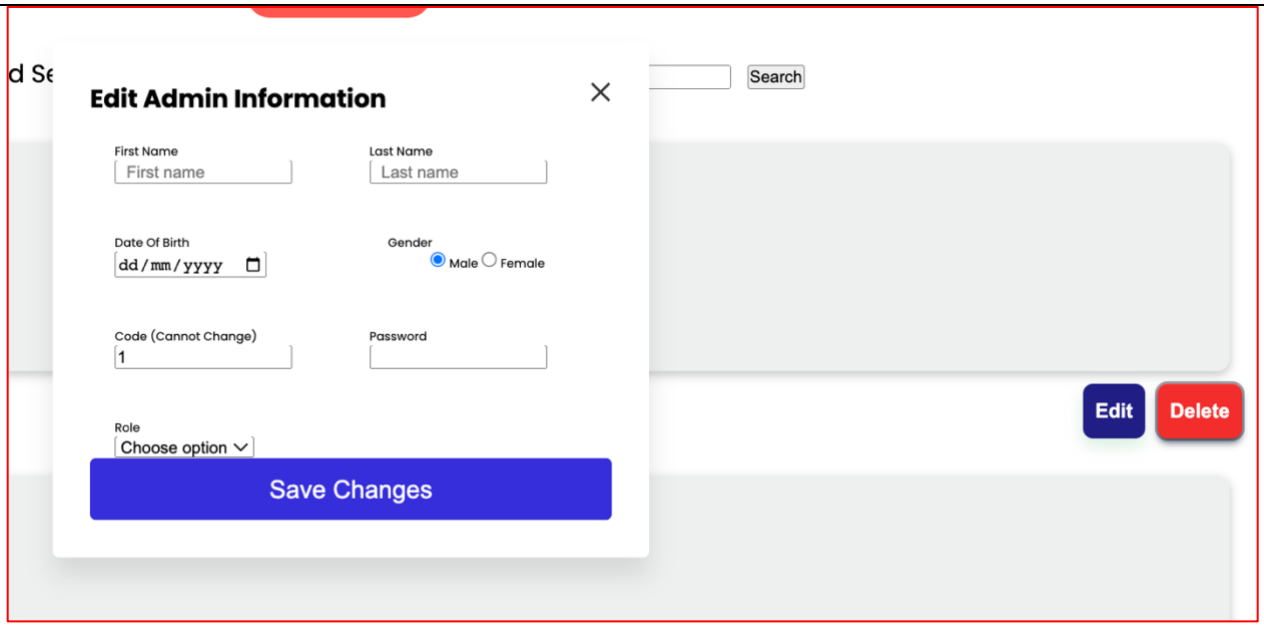
This part is to edit user by create form that fix the `**admin_code**` to let user know that which user that you going to edit.

<script>

```
<!-- This is the edit user information form which will display as a none-->
    <div class="wrapper" id="editForm" style="display:none;">
        <div>
            <h1 class="topic">Edit Admin Information</h1>
            <form action="/edituser" method="POST">
                <input type="hidden" name="admin_code" id="adminCodeInput">
```

```
<div class="row">
    <div class="col box">
        <label for="fname" class="clabel">First Name</label>
        <br>
        <input type="text" class="cinput" name="fname" id="fname" placeholder="  First name" required>
    </div>
    <div class="col box">
        <label for="lname" class="clabel">Last Name</label>
        <br>
        <input type="text" class="cinput" name="lname" id="lname" placeholder="  Last name" required>
    </div>
</div>

<div class="row">
    <div class="col box">
        <label for="birthdate" class="clabel">Date Of Birth</label>
        <br>
        <input type="date" class="cinput" name="dob" id="birthdate" required>
    </div>
    <div class="col">

        <label for="gender" class="clabel">Gender</label>
        <div class="checkbox">

            <input type="radio" checked="checked" name="gender" id="gender" value="M" required>
            <label class="radio-container">Male</label>

            <input type="radio" name="gender" id="gender" value="F" required>
            <label class="radio-container">Female</label>
        </div>

    </div>
</div>

<div class="row">
    <div class="col box">
        <!--The code will be only for read, user cannot change the code-->
```

```
                <label for="code" class="clabel">Code (Cannot Change)</label>
                <br>
                <input type="text" class="cinput" name="code" id="codeAd" readonly>
              </div>
              <div class="col box">
                <label for="password" class="clabel">Password</label>
                <br>
                <input type="password" class="cinput" name="password" id="password" required>
              </div>
            </div>

            <div class="lastcol">
              <label for="role" class="clabel">Role</label>
              <br>
              <select name="role" id="role" class="selectrole" required>
                <option disabled="disabled" selected="selected">Choose option</option>
                <option>Administrator</option>
              </select>
            </div>
            <div class="field btn">
              <div class="btn-layer"></div>
              <input type="submit" value="Save Changes" />
            </div>
            <div class="pass-link" onclick="closeFormEdit()">
              <span class="close-symbol"></span>
            </div>
          </form>

      </div>
    </div>
```
</script>




```
/** To close the form edit is to close the edit form by useing the style.display = none*/
    function closeFormEdit() {
        const form = document.getElementById('editForm');
        form.style.display = 'none';
    }
```

This function is to close the form that pop-up.

<script>

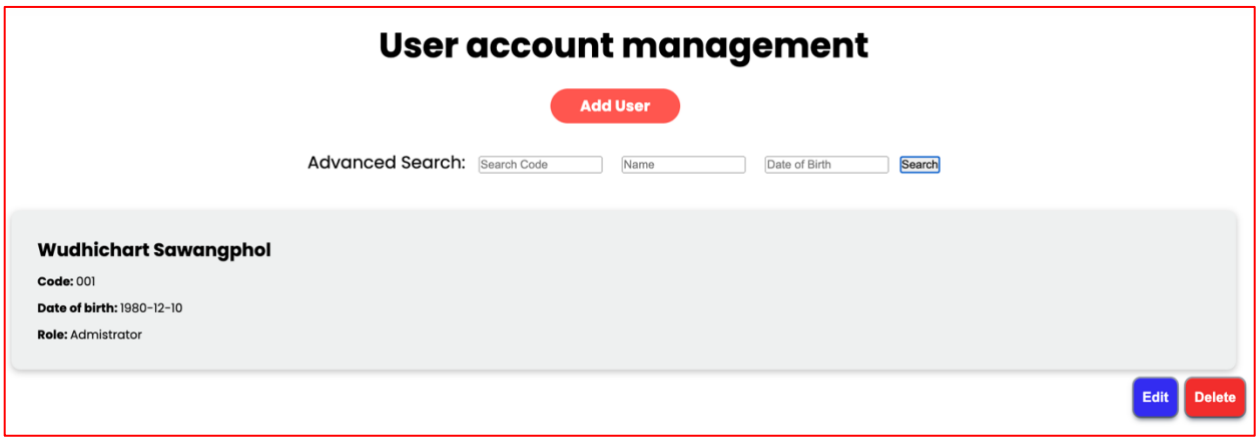
This part is for search form that can search by code, name, and date. To search data we choose POST method and fetch from frontend.js.

```
const searchForm = document.getElementById('search-form');
    /** If user submit the button of search the admin
     * it will com to this
    */
    searchForm.addEventListener('submit', event => {
        event.preventDefault();
        // Keep all of the input value from the box input that has id of code,name,and date
        const code = document.getElementById('code').value;
        const name = document.getElementById('name').value;
        const date = document.getElementById('date').value;

        /** fetch to the port of 3000 which is the client side to send the code, name, and date to be
         * as a query. If the form is empty, it will show everything admin. It fetch with the ,method of 'POST'*/
```

```
        fetch('http://localhost:3000/search-user-submit', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify({ code, name, date }) // convert to JSON and send it
        })
            .then(response => response.json())
            .then(data => {
                document.querySelector("#output").innerHTML = null;  // we point to the output box and ready to do the html
                const jsonData = document.getElementById('output');

                /** we will print out by using innerHTML to show up at the part of the box that has id='output*/
                for (let i = 0; i < data.length; i++) {
                    /** If user click the box it will vall showInfo function and pass the admin_code*/
                    jsonData.innerHTML += `<div class='boxUsers' onclick="showInfo('${data[i].admin_code}')"><h2 class='txt'>` + data[i].fname + " " + data[i].lname + "</h2>"
                        + "<p class='txt'><b>Code: </b>" + data[i].admin_code + "</p>"
                        + "<p class='txt'><b>Date of birth: </b>" + data[i].dob.substring(0, 10) + "</p>"
                        + "<p><b>Role: </b>" + data[i].role + "</p></div>"
                        /* If user click edit, it will call the function if showEditForm and pass the admin code*/
                        + "<div class='button-container'><button class='button-edit' role='button'  data-code='" + data[i].admin_code + "' onclick='showEditForm(" + data[i].admin_code + ")' id='edit'>Edit</button>"
                        + "<button class='button-del' role='button' data-code='" + data[i].admin_code + "' id='delete'>Delete</button></div>";
                }

                document.addEventListener('click', function (event) {
                    if (event.target.classList.contains('button-del')) {
                        /** when user click delete button ,it will come to this code.
                         * First it keep the adminCode to pass to the client side
                         * so when serrver side receive that , they can use it to delete
                         * that information.
                        */
                        const adminCode = event.target.dataset.code;
                        console.log('User Code to be deleted: ' + adminCode);
```

```
                /** Then fetch to client side with the port of 3000 to delete and send the admin code*/
                fetch(`http://localhost:3000/deleteAdmin/`, {
                    method: 'DELETE',
                    headers: {
                        'Content-Type': 'application/json'
                    },
                    body: JSON.stringify({ admin_code: adminCode })
                })
                    .then(response => response.json())
                    .then(data => {
                        console.log(data);
                        if (data === false) {           // If it's work, it will reload the page to update value
                            window.location.reload();
                        }
                    })
                    .catch(error => console.error(error));
            }
        });


    })
    .catch(error => console.error(error));
});
```

</script>



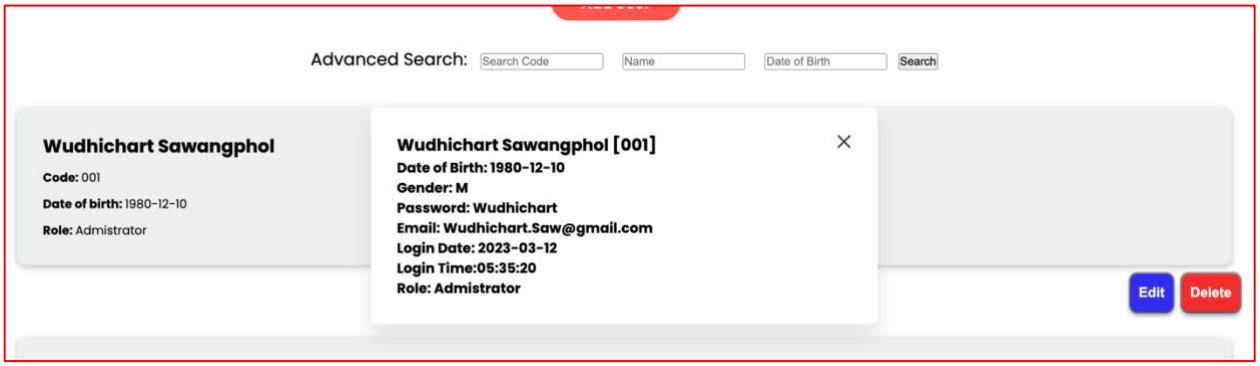
<script>

This part is **`view`** part that we have to fetch **`admin_infos`** from **`frontend.js`** to display the information of each user and make it pop out so the difference can be seen.

```
/* *This showInfo need to get the information from the server,
        * it receive the admincod and pass it to port of 3000 which specify
        * the adminCode.
       */
      function showInfo(adminCode) {
          console.log(adminCode)
          fetch(`http://localhost:3000/admin_infos/${adminCode}`, {
              method: 'GET',
              headers: {
                  'Content-Type': 'application/json'
              }
          })
              .then(response => response.json())
              .then(data => {                    /* After the response, we get the information now*/
                  console.log(data);
                  let code = data.data.admin_code;
                  let fname = data.data.fname;
                  let lname = data.data.lname;
                  let dob = data.data.dob;
                  let gender = data.data.gender;
                  let pass = data.data.password;
                  let log = data.data.login_log;
                  let role = data.data.role;


                  document.querySelector("#pop").innerHTML = null;  // we will print it here, the part of pop to pop up the box
                  const pop = document.getElementById('pop');
                  // display this pop up box to show the infor
                  document.getElementById('infoPop').style.display = 'block';
                  // show all details
                  pop.innerHTML += "<h1>" + fname + " " + lname + " [" + code + "] </h1><h3 class='detail'>Date of Birth: " + dob.substring(0, 10) + "</h3>";
                  pop.innerHTML += "<h3 class='detail'>Gender: " + gender + "</h3><h3 class='detail'>Password: " + pass + "</h3>";
```

```
                pop.innerHTML += "<h3 class='detail'>Email: " + fname+"."+lname.substring(0,3)+"@gmail.com" +
"</h3>";
                pop.innerHTML += "<h3 class='detail'>Login Date: " + log.substring(0, 10) + "</h3><h3
class='detail'>Login Time:" + log.substring(11, 19) + "</h3><h3 class='detail'>Role: " + role + "</h3>";
                pop.innerHTML += "<div class='pass-link' onclick='closeForm()'><span class='close-
symbol'></span></div>";
            })
```

</script>




```
// Same as this, this will close the pop up box, to be display as a none again
    function closeForm() {
        const form = document.getElementById('infoPop');
        form.style.display = 'none';
    }
```

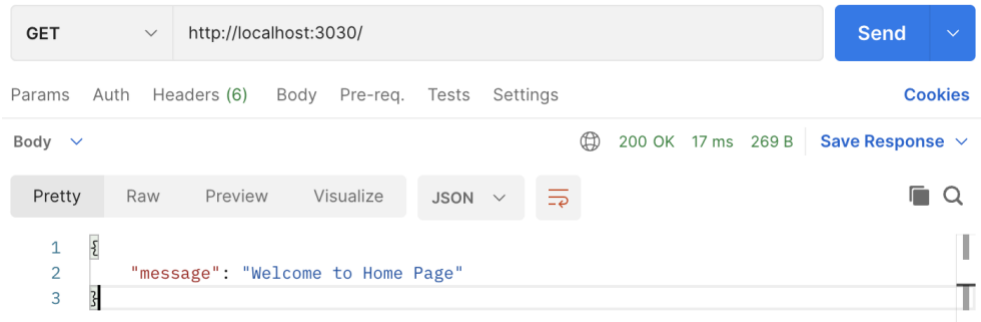
This function is for close form that pop up.

File	user.html
Code and User-Interface	This part is html part that we create a form to let user entire the form of add new user. [For the nav bar we use as the same code] This class is to keep everything in one box. `<div class="wrapper">` `<!-- This is a section to let people who have forgotten their password know what to enter. -->`

```
<div class="head-text">
    <h1>Add User</h1>
    <p>To add a new user. Please complete this entire form.</p>
</div>
<!--this will keep every box form-->
<div class="form-inner">
    <!-- Form for user to sign up -->
    <form action="/signup-submit" method="POST" class="product">
        <!--Class field is to hold all the value from the input.-->
        <div class="field">
            <label for="titletxt">First Name</label>
            <input type="text" id="titletxt"  name="fname" placeholder="First Name of User" required/>
        </div>
        <br>
        <div class="field">
            <label for="destxt">Last Name</label>
            <input type="text" id="destxt"  name="lname" placeholder="Last Name of User" required/>
        </div>
        <br>

        <div class="field">
            <label for="date">Date Of Birth</label>
            <input type="date" id="date" name="dob"  placeholder="DD-MM-YYYY" required/>
        </div>
        <br />
        <div class="field">
            <label for="code">Gender</label>
            <div class="checkbox">
                <label class="radio-container"> - Male</label>
                <input type="radio" name="gender"  checked="checked" id="gender" value="M" required>
                <label class="radio-container"> - Female</label>
                <input type="radio" name="gender" id="gender" value="F" required>
            </div>
        </div>
        <br /><br />
        <div class="field">
            <label for="code">Code</label>
```

```
            <input type="text" id="code" name="code" placeholder="xxx" required/>
        </div>
        <br />
        <div class="field">
            <label for="pass">Password</label>
            <input type="text" id="pass" name="password"  title="password" required/>
        </div>
        <br>
         <div class="field">
            <label for="role">Role</label>
            <select name="role" id="role" class="selectrole" required>
                <option disabled="disabled" selected="selected">Choose option</option>
                <option>Administrator</option>
            </select>
        </div>
        <br>

        <div class="total">
            <div class="field btn">
                <a href="/html/User-account-management.html">Cancel</a>
            </div>
            <div class="field btn">
                <div class="btn-layer" id="ad"></div>
                <input type="submit" value="Add" />

            </div>
        </div>

    </form>
  </div>
</div>
```

This is User-Interface:

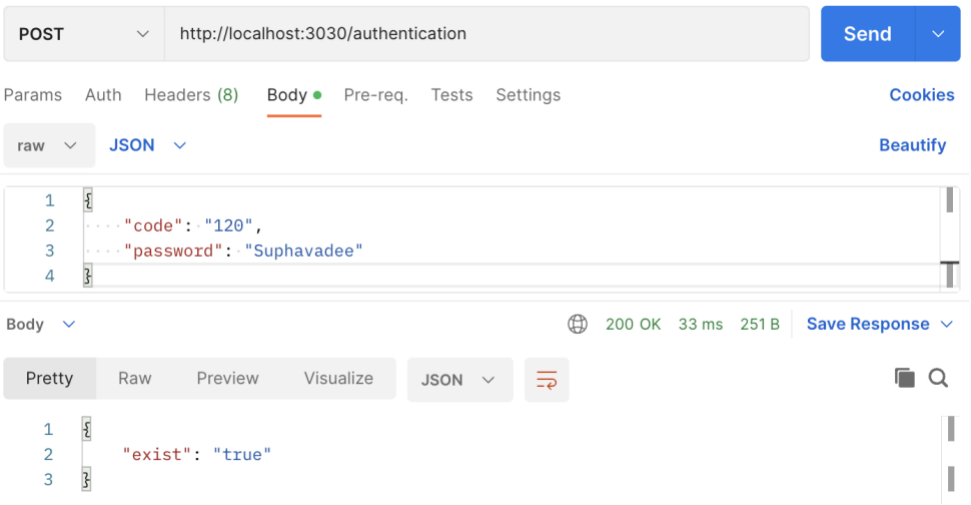
File: Backend.js [Web Service: Administrator]	
Code	`/* a route handler for HTTP GET requests to the root path '/' of the server, using the get method of the router instance. */` `router.get('/', (req, res) => {` `  console.log('Request at /');` `  return res.send({ message: 'Welcome to Home Page' })` `})`
Test case:	TEST CASE: method: GET URL: http://localhost:3030/
Postman:	
Explanation	This is handling the get request at '/' the root path and responding the message "Welcome to Home Page".

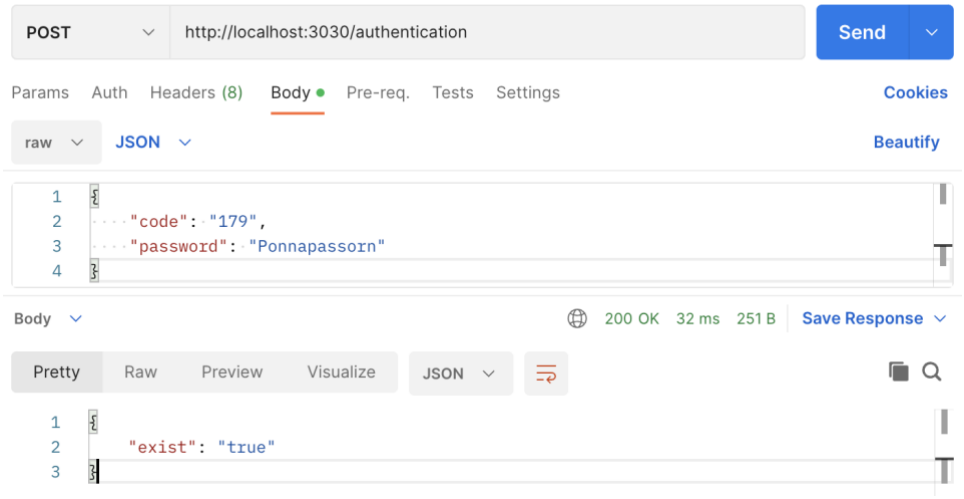
File: Backend.js [Web Service: Administrator] **[SIGN IN]**	
Code	

```
/* TASK 2.1: FOR SIGN IN */
router.post('/authentication', (req, res) => {
    console.log('Request at /login-submit');
    console.log("Login by " + req.body.code);//
    let code = req.body.code;
    let pass = req.body.password;

    let sql = `select * from admin_info where (admin_code="` + code + `" && password="` + pass +
`");`;

    connection.query(sql, function (error, results) {
        if (error) throw error;
        // console.log(`${results.length} row returned`);

        if (results.length === 0) {
            console.log("Not Found");
            return res.json({ "exist": "false" });
        } else {
            console.log("Found");
            return res.json({ "exist": "true" });
        }
    });
})
```

Test case 1:	

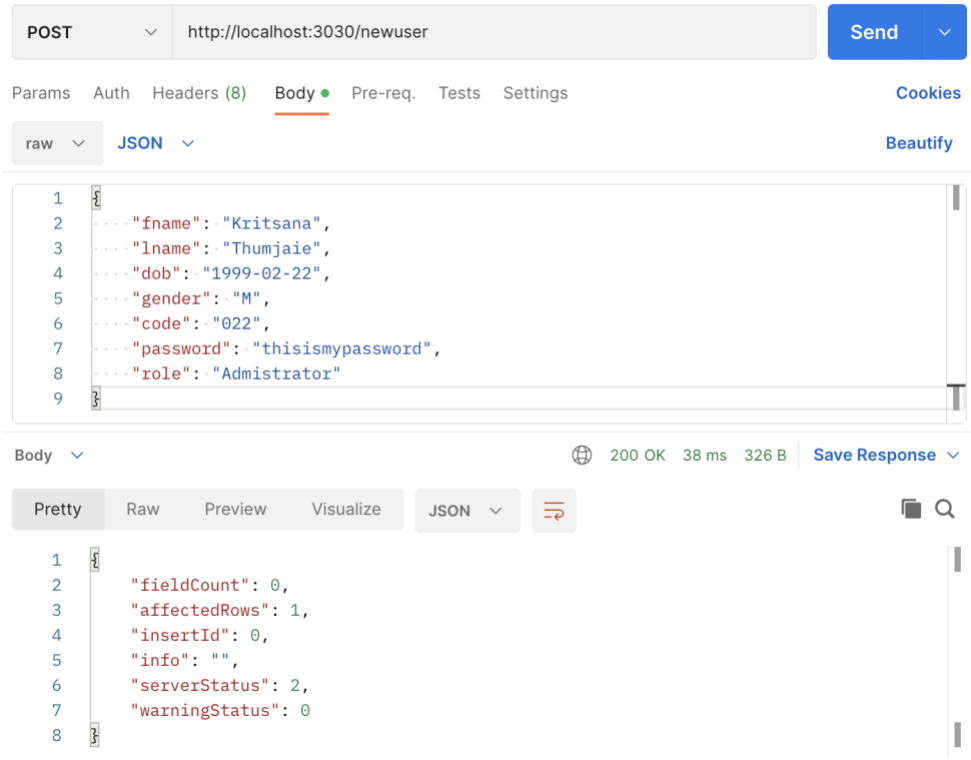
```
TEST CASE 1:
method: POST
```

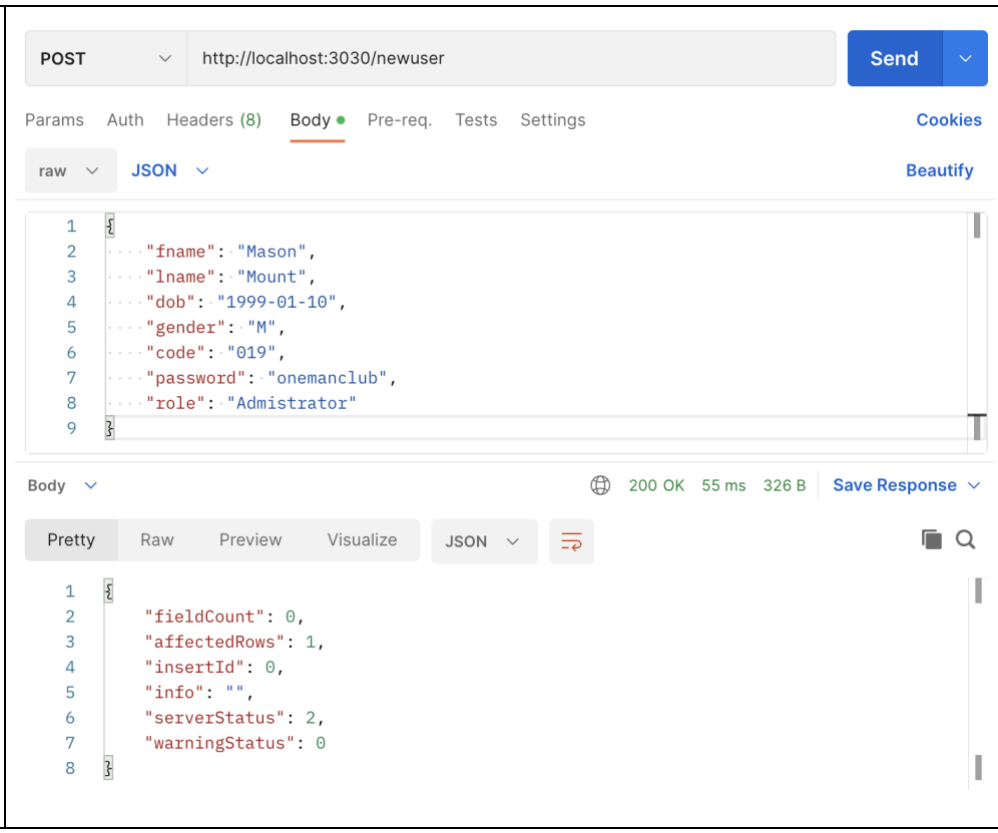
	URL: http://localhost:3030/authentication Body: raw JSON { "code": "120", "password": "Suphavadee" }
Test case 2:	TEST CASE 2: method: POST URL: http://localhost:3030/authentication Body: raw JSON { "code": "179", "password": "Ponnapassorn" }
Postman 1:	

Postman 2:	
Explanation	This is handling the post request at path '/authentication'. It retrieves value of code and password, then matches with sql table admin_info if code equals to admin_code, and pass equals to password. Lastly, if the length of the result equals to zero, that means there's no result, so it will shows "Not Found". Otherwise, it will shows "Found".

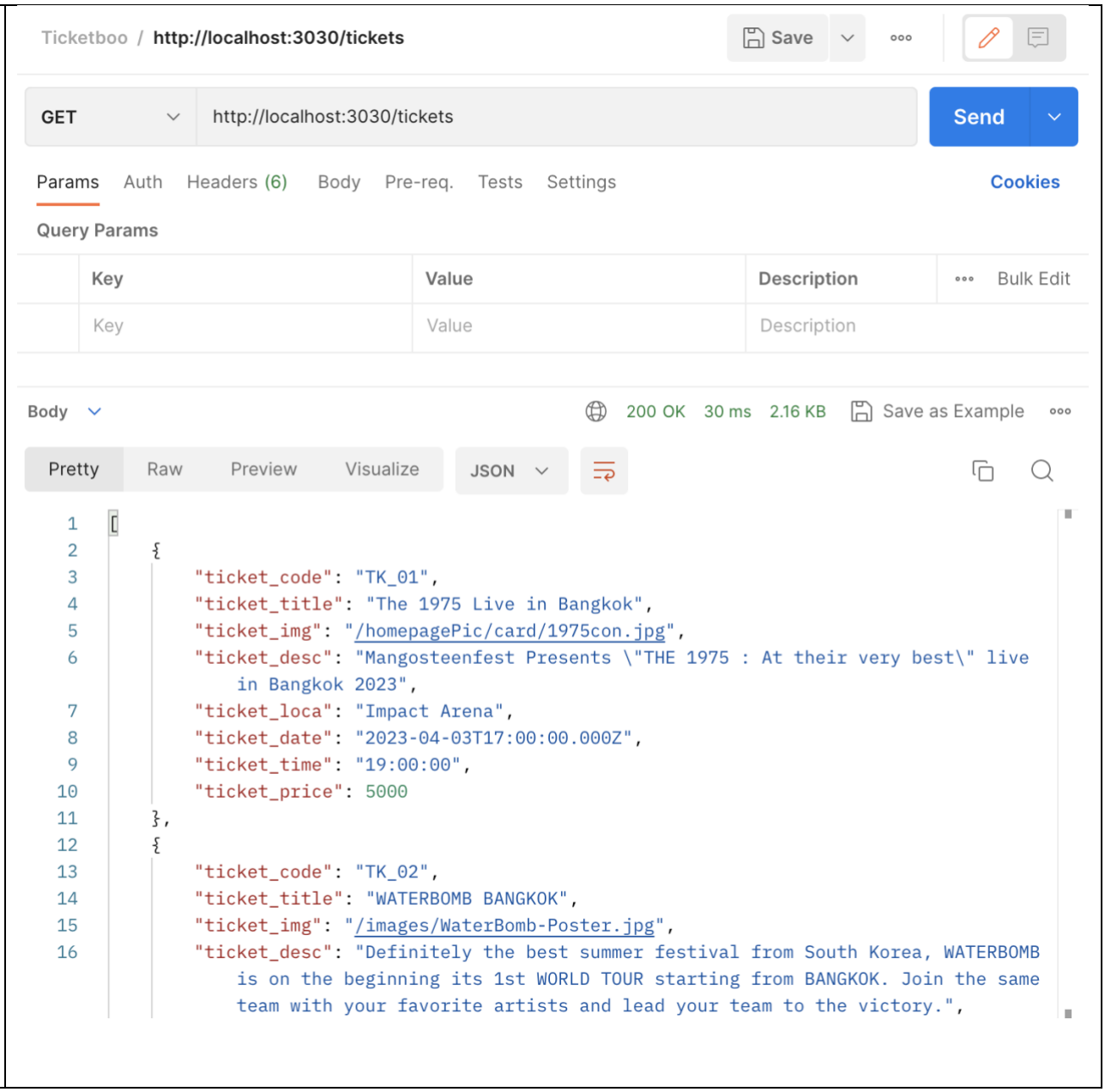
File: Backend.js [Web Service: Administrator] **[SIGN UP]**	
Code	```/* TASK 2.1: FOR SIGN UP */ router.post('/newuser', (req, res) => { console.log('Request at /signin-submit'); console.log("Form submitted by " + req.body.code);// let fname = req.body.fname; let lname = req.body.lname; let dob = req.body.dob; let gen = req.body.gender; let code = req.body.code;```

	`` let pass = req.body.password; let role = req.body.role; const currentDate = new Date(); const date = currentDate.getFullYear() + "-" + (currentDate.getMonth() + 1) + "-" currentDate.getDate(); const time = currentDate.getHours() + ":" + currentDate.getMinutes() + ":" + currentDate.getSeconds(); console.log(gen) console.log(role) console.log(`INSERT INTO admin_info VALUES(` + code + `', '` + fname + `', '` + lname + `', '` + dob + `', '`+ gen + `', '` + pass + `', '` + date + " " + time + `', '` + role + `');`); console.log("Not Found"); let sql = `INSERT INTO admin_info VALUES(` + code + `', '` + fname + `', '` + lname + `', '` + dob + `', '`+ gen + `', '` + pass + `', '` + date + " " + time + `', '` + role + `');`; connection.query(sql, function (error, results) { if (error) throw error; return res.send(results);; }); }) ```
Explanation	This is handling the post request at path '/newuser'. It retrieves fname, lname, dob, gender, code, password, and role. After that, inserting the new value into sql in table admin_info including current ate and time.
Test case 1:	TEST CASE 1: method: POST URL: http://localhost:3030/newuser Body: raw JSON { "fname": "Kritsana", "lname": "Thumjaie", "dob": "1999-02-22", "gender": "M", "code": "022", "password": "thisismypassword", "role": "Admistrator" }

Postman 1:	
Test case 2:	TEST CASE 2: method: POST URL: http://localhost:3030/newuser Body: raw JSON { "fname": "Mason", "lname": "Mount", "dob": "1999-01-10", "gender": "M", "code": "019", "password": "onemanclub", "role": "Admistrator" }

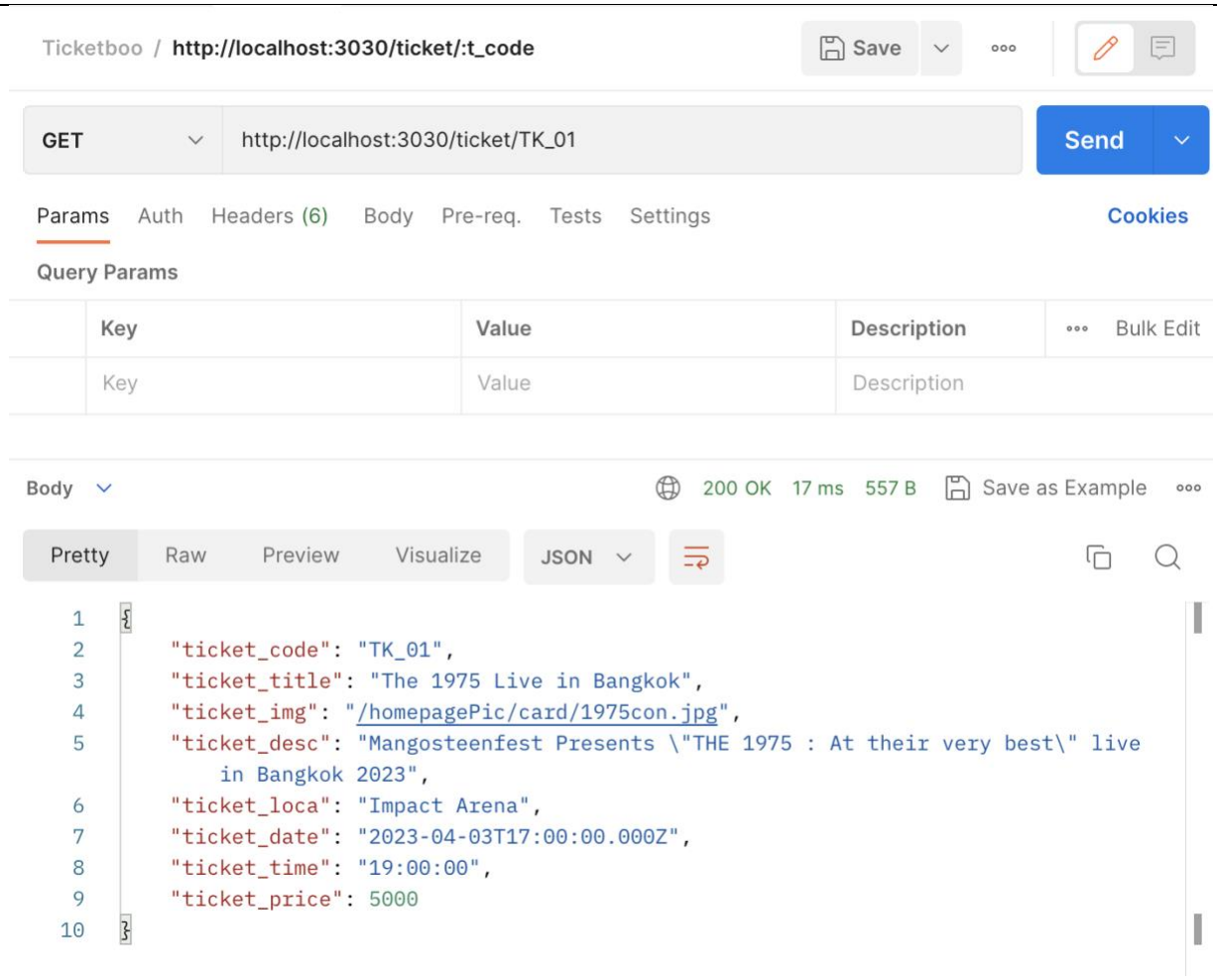
Postman 2:	 POST http://localhost:3030/newuser Send Params Auth Headers (8) Body Pre-req. Tests Settings Cookies raw JSON Beautify { "fname": "Mason", "lname": "Mount", "dob": "1999-01-10", "gender": "M", "code": "019", "password": "onemanclub", "role": "Admistrator" } Body 200 OK 55 ms 326 B Save Response Pretty Raw Preview Visualize JSON { "fieldCount": 0, "affectedRows": 1, "insertId": 0, "info": "", "serverStatus": 2, "warningStatus": 0 }

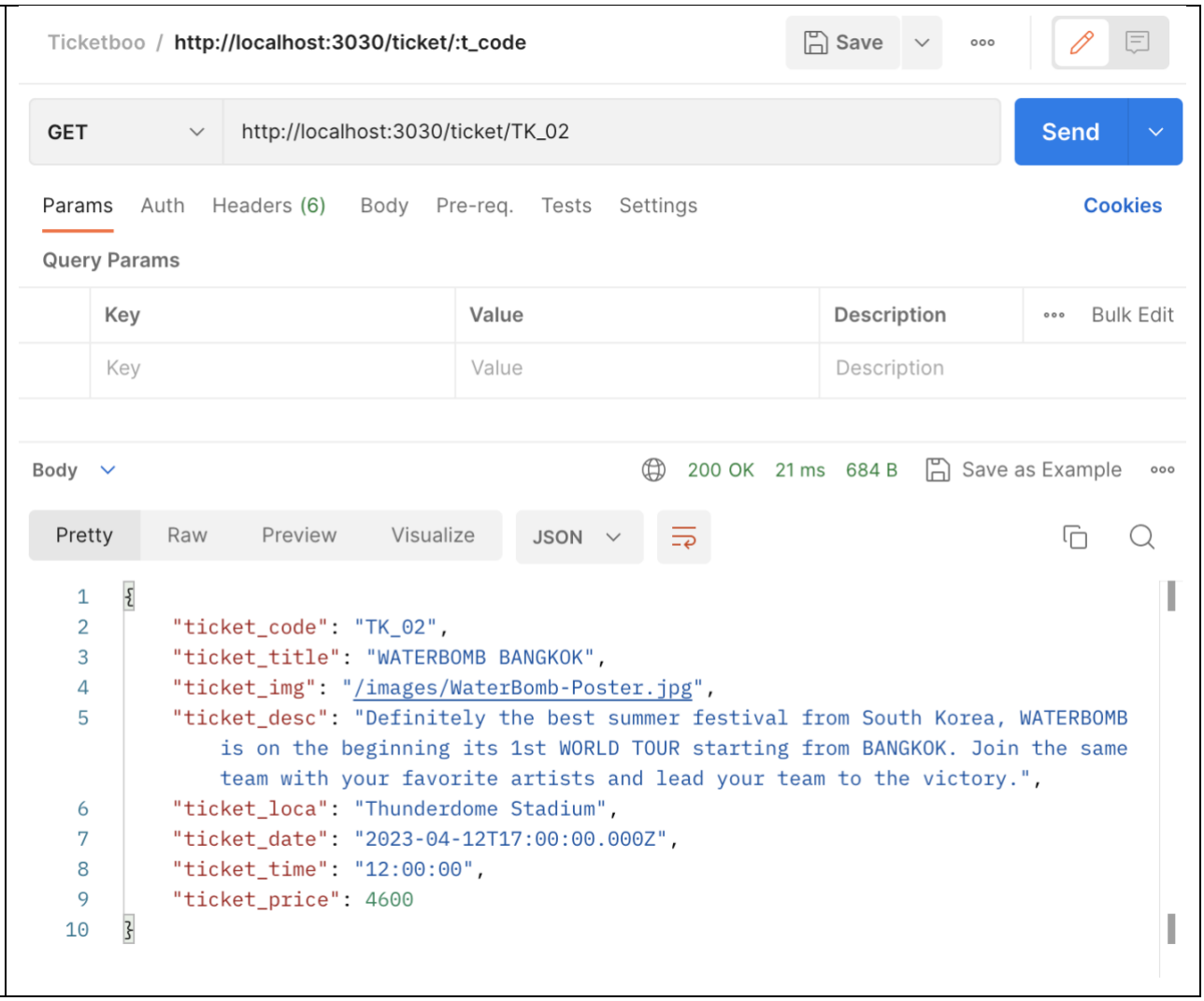
File	Backend.js (Web Service: Product) [Select (get) all Product]
Code	`/* THIS PATH FOR SELECT ALL TICKET */` `router.get('/tickets', function (req, res) {` `  /* The command we want to select all data from the database. */` `  const sql = "SELECT * FROM ticket"` `  connection.query(sql, (err, results) => {` `  if (err) return res.json(err);` `  return res.send(results);` `  });` `});`
Explanation	This is a route handler for the GET method on the "/tickets" of a web application. When a user sends a GET request to the "/tickets". The function executes a SQL query to retrieve all rows from a "ticket" table in a database. If an error occurs while executing the query, the function sends a JSON response containing information about the error to the client. Otherwise, the function sends a JSON response containing the results of the query to the client.
Test case in Postman	`// TEST CASE:` `// method: GET` `// URL: http://localhost:3030/tickets`


Ticketboo / http://localhost:3030/tickets
Save
GET
http://localhost:3030/tickets
Send
Params
Auth
Headers (6)
Body
Pre-req.
Tests
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
Key
Value
Description
Body
200 OK
30 ms
2.16 KB
Save as Example
Pretty
Raw
Preview
Visualize
JSON
[
{
"ticket_code": "TK_01",
"ticket_title": "The 1975 Live in Bangkok",
"ticket_img": "/homepagePic/card/1975con.jpg",
"ticket_desc": "Mangosteenfest Presents \"THE 1975 : At their very best\" live in Bangkok 2023",
"ticket_loca": "Impact Arena",
"ticket_date": "2023-04-03T17:00:00.000Z",
"ticket_time": "19:00:00",
"ticket_price": 5000
},
{
"ticket_code": "TK_02",
"ticket_title": "WATERBOMB BANGKOK",
"ticket_img": "/images/WaterBomb-Poster.jpg",
"ticket_desc": "Definitely the best summer festival from South Korea, WATERBOMB is on the beginning its 1st WORLD TOUR starting from BANGKOK. Join the same team with your favorite artists and lead your team to the victory.",

```
            team with your favorite artists and lead your team to the victory.",
        "ticket_loca": "Thunderdome Stadium",
        "ticket_date": "2023-04-12T17:00:00.000Z",
        "ticket_time": "12:00:00",
        "ticket_price": 4600
    },
    {
        "ticket_code": "TK_03",
        "ticket_title": "ABOUT DAMN TIME",
        "ticket_img": "/images/Ph1-Poster.jpg",
        "ticket_desc": "IMC Live Global is proud to present 2023 pH-1 ABOUT DAMN TIME
            WORLD TOUR in Bangkok on 19 March 2022, 7 pm at the centralwOrld LIVE.",
        "ticket_loca": "CentralwOrld LIVE",
        "ticket_date": "2023-03-18T17:00:00.000Z",
        "ticket_time": "19:00:00",
        "ticket_price": 3800
    },
    {
        "ticket_code": "TK_04",
        "ticket_title": "Arctic Monkeys Live in Bangkok 2023",
        "ticket_img": "/homepagePic/card/arcMon.png",
        "ticket_desc": "The wait is over! Get ready for the indie rock icon of the era
            and their first live show in Thailand “ARCTIC MONKEYS LIVE IN BANGKOK”, 9
            March 2023 at Bitec Hall.",
        "ticket_loca": "BITEC Bangna",
        "ticket_date": "2023-03-08T17:00:00.000Z",
        "ticket_time": "18:00:00",
        "ticket_price": 6000
    },
    {
        "ticket_code": "TK_05",
        "ticket_title": "Pelupo Festival 2023",
        "ticket_img": "/images/Pelupo-Poster.png",
        "ticket_desc": "Leave your worries behind you and prepare to join us at PELUPO
            International Music Festival at The Fields at Siam Country Club in Chonburi
            on March 11, 2023.",
        "ticket_loca": "Siam Country Club, Pattaya, Thailand",
        "ticket_date": "2023-03-10T17:00:00.000Z",
        "ticket_time": "15:00:00",
        "ticket_price": 4000
    }
]
```

File	Backend.js (Web Service: Product) [Select (get) a Product by ticket_code]
Code	`/* THIS PATH FOR SELECT A TICKET BASED ON ticket_code*/` `router.get('/ticket/:ticket_code', function (req, res) {` `  let ticket_code = req.params.ticket_code;` `  if (!ticket_code) {` `  return res.status(404).send({ error: true, message: 'Please provide Ticket information.' });` `  }` `  /* The command we want to select data by ticket_code from the database. */` `  connection.query('SELECT * FROM ticket where ticket_code=?', ticket_code, function (err, results) {` `  if (err) throw err;` `  res.json(results[0]);` `  console.log(`Sending product result of ticket_code = ${ticket_code}`);` `  });` `});`
Explanation	This route can be used to retrieve **a single ticket record** from the "**ticket"** table in database, based on the `**ticket_code**` provided in the URL parameter. The ticket details are returned as a JSON object in the response body. If the `**ticket_code**` parameter is exist, the function executes a SQL query to retrieve the row in the "**ticket**" table where the `**ticket_code**` column matches the value of the `**ticket_code**` parameter.
Test case 1 in Postman	`// TEST CASE: 1` `// method: GET` `// URL: http:///localhost:3030/ticket/TK_01` `// OUTPUT: It will print result of ticket_code = TK_01`

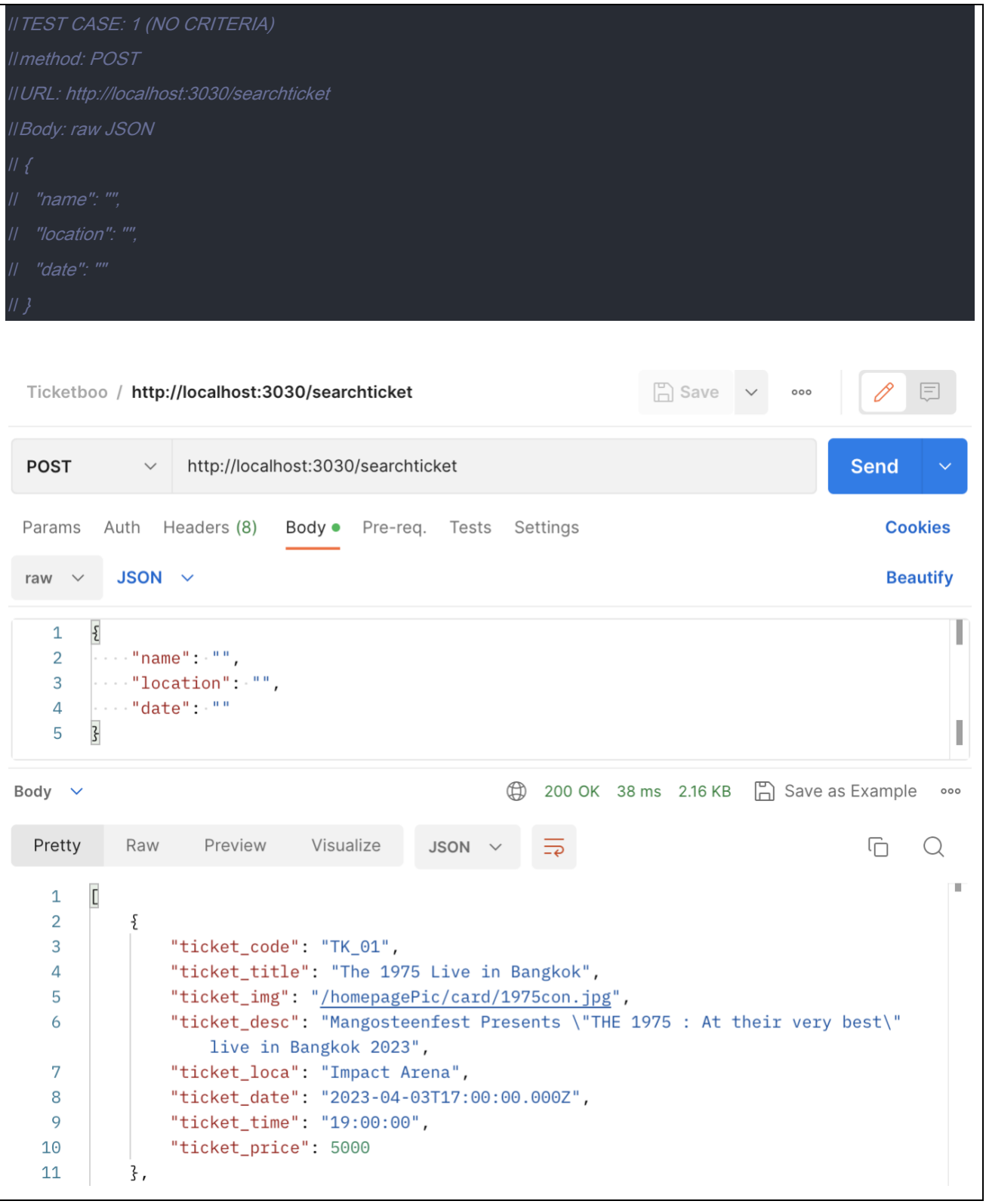
	
Test case 2 in Postman	*// TEST CASE: 2* *// method: GET* *// URL: http://localhost:3030/ticket/TK_02* *// OUTPUT: It will print result of ticket_code = TK_02*


Ticketboo / http://localhost:3030/ticket/:t_code
Save
GET
http://localhost:3030/ticket/TK_02
Send
Params
Auth
Headers (6)
Body
Pre-req.
Tests
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
Key
Value
Description
Body
200 OK
21 ms
684 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
{
"ticket_code": "TK_02",
"ticket_title": "WATERBOMB BANGKOK",
"ticket_img": "/images/WaterBomb-Poster.jpg",
"ticket_desc": "Definitely the best summer festival from South Korea, WATERBOMB is on the beginning its 1st WORLD TOUR starting from BANGKOK. Join the same team with your favorite artists and lead your team to the victory.",
"ticket_loca": "Thunderdome Stadium",
"ticket_date": "2023-04-12T17:00:00.000Z",
"ticket_time": "12:00:00",
"ticket_price": 4600
}

File	Backend.js (Web Service: Product) [Search: a product in each criterion]
Code	

```
/* TASK 2.2: Part 1 */
//SEARCH for TICKET in each criteria => By name, location, and date
router.post('/searchticket', (req, res) => {
    console.log("::Welcome to path search ticket::");
    console.log("--------------------------------");

    /* This is the request object, which represents the HTTP request sent by the client or frontend. */
    let ticket_title = req.body.name;
    let ticket_loca = req.body.location;
    let ticket_date = req.body.date;
    console.log("Search by name: " + ticket_title + "| Search by location: " + ticket_loca + "| Search by: date: " +
ticket_date);

    /* The command we want to select data from the database by name, location, and date. */
    let queryString = `select * from ticket where ticket_title like '%${ticket_title}%' and ticket_loca like '%${ticket_loca}%'
and ticket_date like '%${ticket_date}%';`;
    // check if each query attribute has a value or not
    connection.query(queryString, (error, results) => {
        if (error) throw error;
        console.log(`${results.length} result(s) found`);
        console.log("--------------------------------");
        return res.send(results);
    });
});
```

Explanation	This route is when a user sends a **POST** request to the "**/searchticket**", the handler function executes. The function expects a JSON payload in the request body that contains optional "name", "location", and "date" properties, which represent search parameters for tickets. It constructs a SQL query string that searches the "ticket" table for rows where the **"ticket_title", "ticket_loca",** and **"ticket_date"** columns contain substrings that match the search parameters.

Test case 1 in Postman [Search: No criteria]	

```
    {
        "ticket_code": "TK_02",
        "ticket_title": "WATERBOMB BANGKOK",
        "ticket_img": "/images/WaterBomb-Poster.jpg",
        "ticket_desc": "Definitely the best summer festival from South Korea,
            WATERBOMB is on the beginning its 1st WORLD TOUR starting from BANGKOK.
            Join the same team with your favorite artists and lead your team to the
            victory.",
        "ticket_loca": "Thunderdome Stadium",
        "ticket_date": "2023-04-12T17:00:00.000Z",
        "ticket_time": "12:00:00",
        "ticket_price": 4600
    },
    {
        "ticket_code": "TK_03",
        "ticket_title": "ABOUT DAMN TIME",
        "ticket_img": "/images/Ph1-Poster.jpg",
        "ticket_desc": "IMC Live Global is proud to present 2023 pH-1 ABOUT DAMN TIME
            WORLD TOUR in Bangkok on 19 March 2022, 7 pm at the centralwOrld LIVE.",
        "ticket_loca": "CentralwOrld LIVE",
        "ticket_date": "2023-03-18T17:00:00.000Z",
        "ticket_time": "19:00:00",
        "ticket_price": 3800
    },
    {
        "ticket_code": "TK_04",
        "ticket_title": "Arctic Monkeys Live in Bangkok 2023",
        "ticket_img": "/homepagePic/card/arcMon.png",
        "ticket_desc": "The wait is over! Get ready for the indie rock icon of the era
            and their first live show in Thailand “ARCTIC MONKEYS LIVE IN BANGKOK”, 9
            March 2023 at Bitec Hall.",
        "ticket_loca": "BITEC Bangna",
        "ticket_date": "2023-03-08T17:00:00.000Z",
        "ticket_time": "18:00:00",
        "ticket_price": 6000
    },
```

<table>
<tr>
<td></td>
<td>

```
    {
        "ticket_code": "TK_05",
        "ticket_title": "Pelupo Festival 2023",
        "ticket_img": "/images/Pelupo-Poster.png",
        "ticket_desc": "Leave your worries behind you and prepare to join us at PELUPO
            International Music Festival at The Fields at Siam Country Club in Chonburi
            on March 11, 2023.",
        "ticket_loca": "Siam Country Club, Pattaya, Thailand",
        "ticket_date": "2023-03-10T17:00:00.000Z",
        "ticket_time": "15:00:00",
        "ticket_price": 4000
    }
]
```

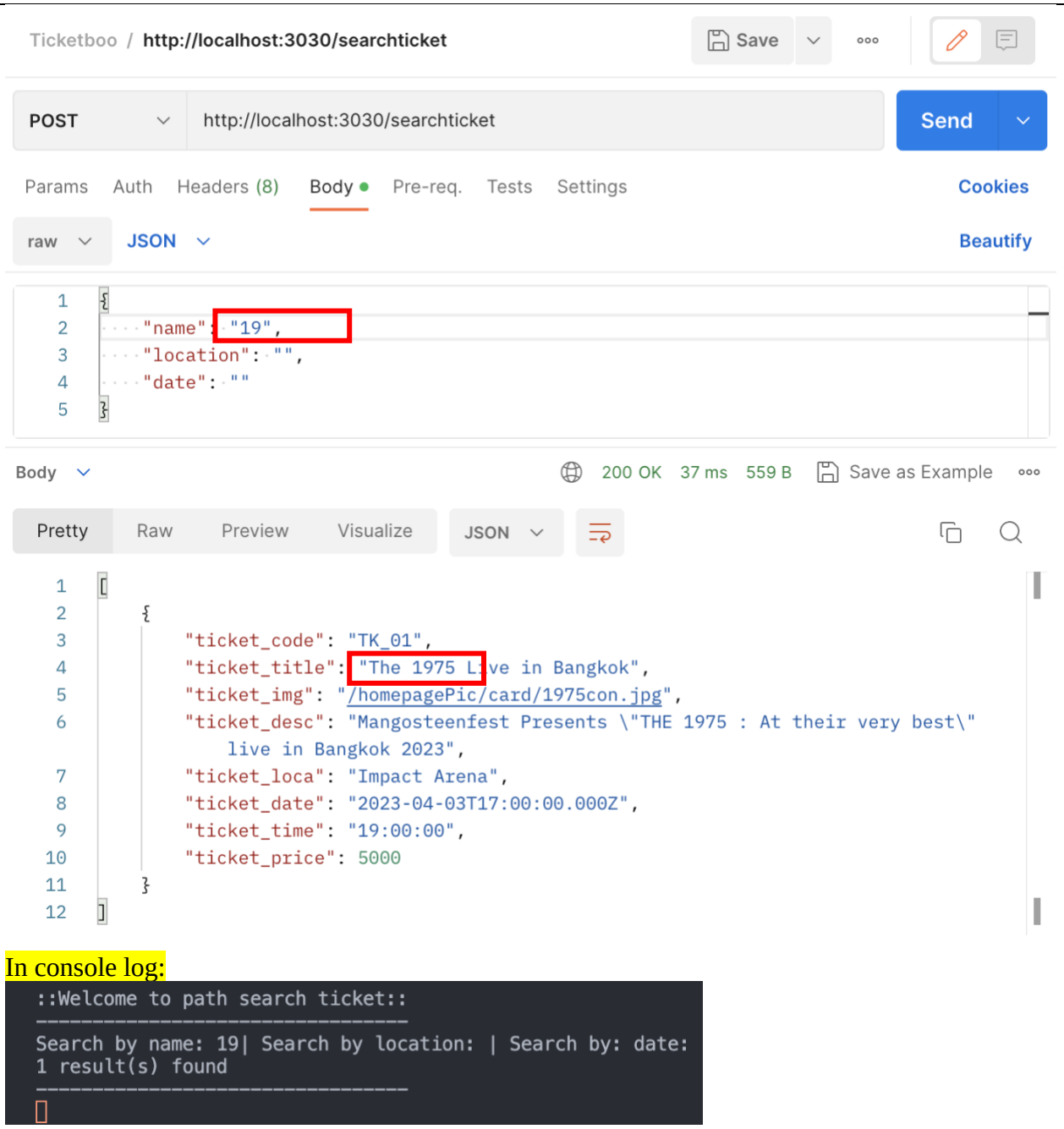
In console log:

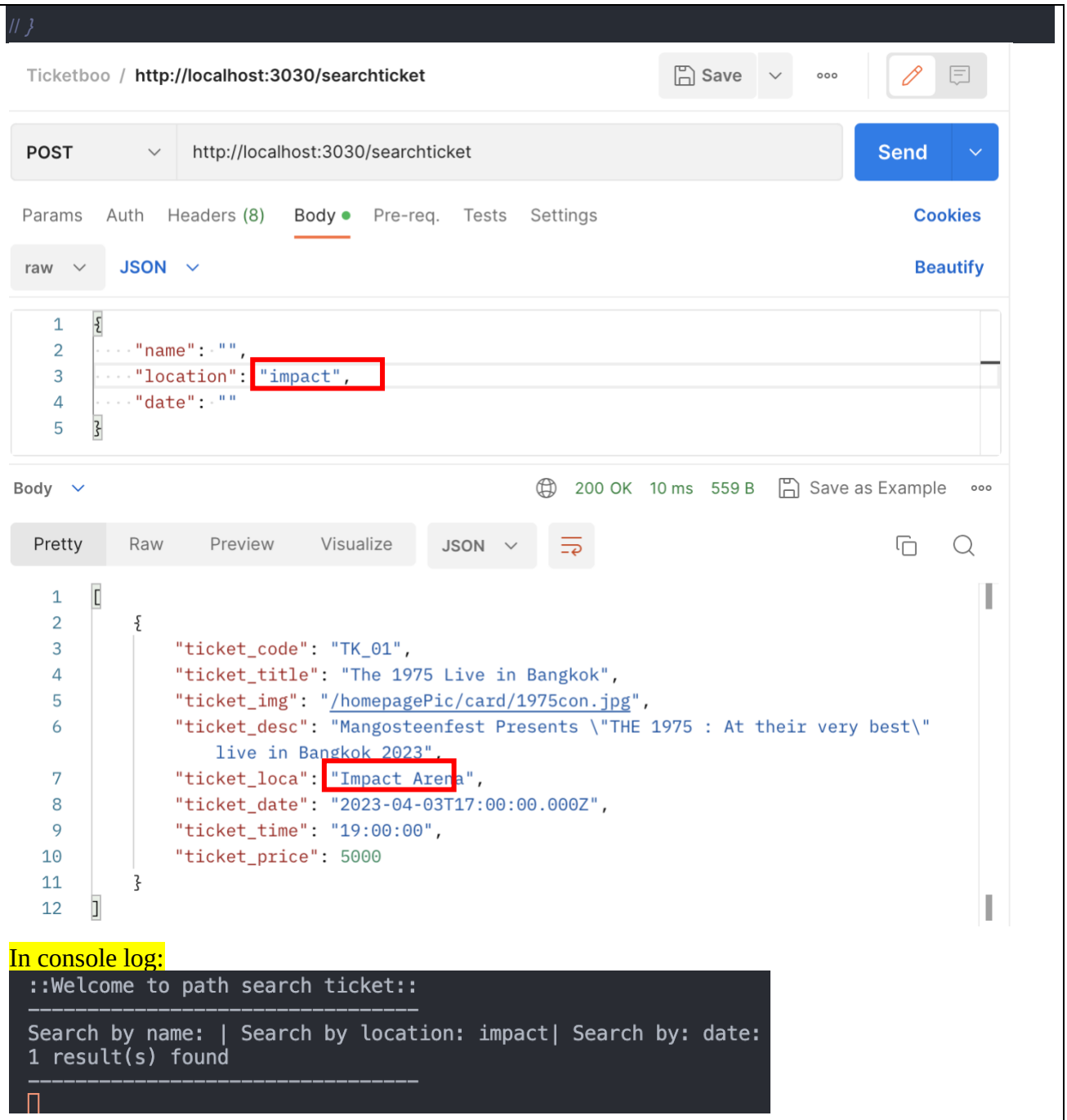
```
Server listening at Port 3030
Connected DB: ticketboo
::Welcome to path search ticket::
---------------------------------
Search by name: | Search by location: | Search by: date:
5 result(s) found
---------------------------------
```

</td>
</tr>
<tr>
<td>Test case 2 in Postman [Search: by name]</td>
<td>

```
// TEST CASE: 2 (SEARCH BY NAME)
// method: POST
// URL: http://localhost:3030/searchticket
// Body: raw JSON
// {
//   "name": "19",
//   "location": "",
//   "date": ""
// }
```

</td>
</tr>
</table>

	 In console log: ::Welcome to path search ticket:: Search by name: 19| Search by location: | Search by: date: 1 result(s) found
Test case 3 in Postman [Search: by location]	// TEST CASE: 3 (SEARCH BY LOCATION) // method: POST // URL: http://localhost:3030/searchticket // Body: raw JSON // { // "name": "", // "location": "impact", // "date": ""

	 In console log: ::Welcome to path search ticket:: Search by name: | Search by location: impact| Search by: date: 1 result(s) found
Test case 4 in Postman [Search: by date]	// TEST CASE: 4 (SEARCH BY DATE) // method: POST // URL: http://localhost:3030/searchticket // Body: raw JSON // { // "name": "", // "location": "",

```
//   "date": "2023-04"
// }
```

Ticketboo / **http://localhost:3030/searchticket**

Save

POST http://localhost:3030/searchticket **Send**

Params Auth Headers (8) Body Pre-req. Tests Settings Cookies

raw JSON Beautify

```
{
    "name": "",
    "location": "",
    "date": "2023-04"
}
```

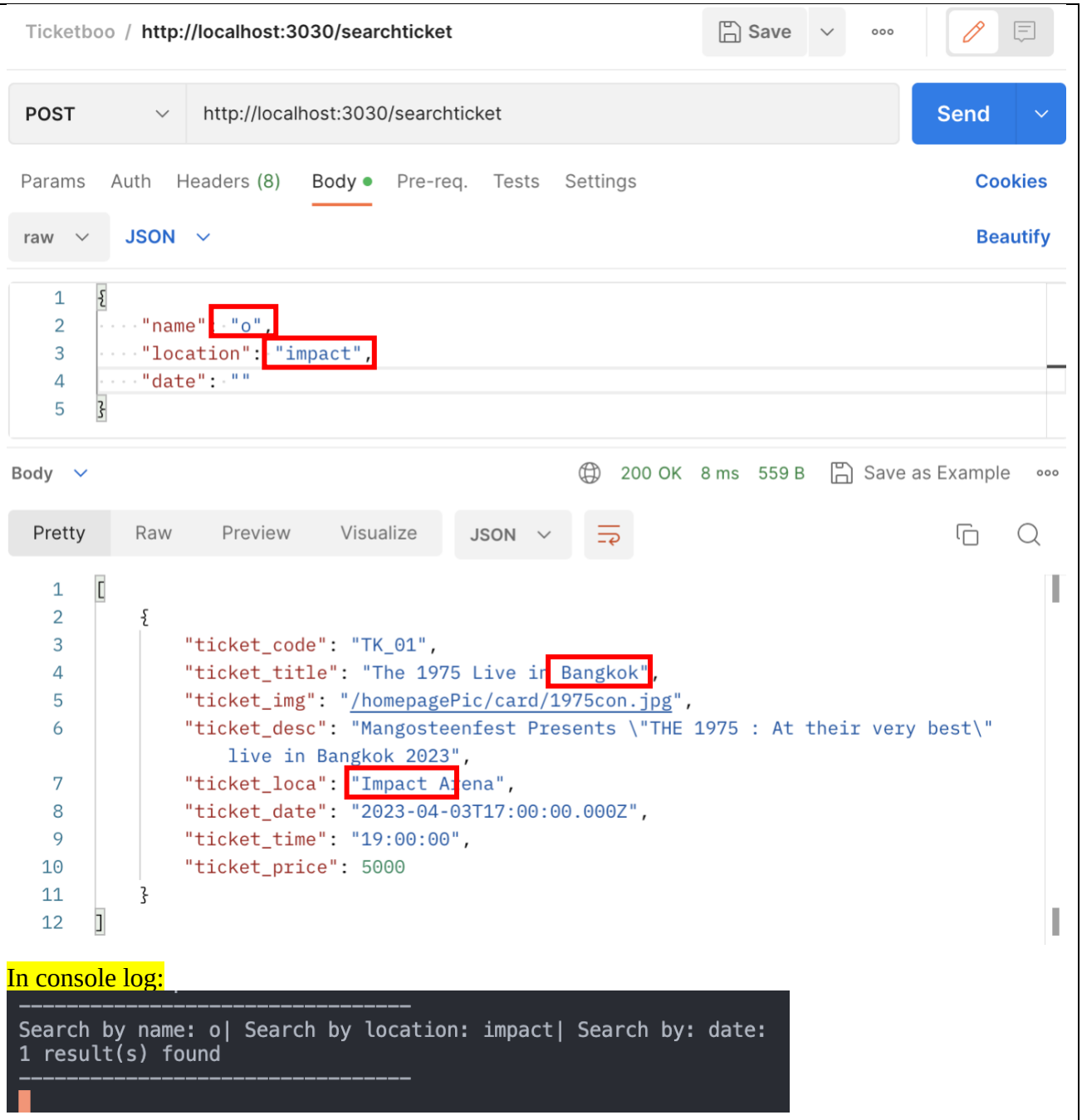
Body 200 OK 10 ms 1007 B Save as Example

Pretty Raw Preview Visualize JSON


```
[
    {
        "ticket_code": "TK_01",
        "ticket_title": "The 1975 Live in Bangkok",
        "ticket_img": "/homepagePic/card/1975con.jpg",
        "ticket_desc": "Mangosteenfest Presents \"THE 1975 : At their very best\"
            live in Bangkok 2023",
        "ticket_loca": "Impact Arena",
        "ticket_date": "2023-04-03T17:00:00.000Z",
        "ticket_time": "19:00:00",
        "ticket_price": 5000
    },
    {
        "ticket_code": "TK_02",
        "ticket_title": "WATERBOMB BANGKOK",
        "ticket_img": "/images/WaterBomb-Poster.jpg",
        "ticket_desc": "Definitely the best summer festival from South Korea,
            WATERBOMB is on the beginning its 1st WORLD TOUR starting from BANGKOK.
            Join the same team with your favorite artists and lead your team to the
            victory.",
        "ticket_loca": "Thunderdome Stadium",
        "ticket_date": "2023-04-11T17:00:00.000Z",
        "ticket_time": "12:00:00",
        "ticket_price": 4600
    },
```


In console log:

	`` ::Welcome to path search ticket:: _____ Search by name: | Search by location: | Search by: date: 2023-04 2 result(s) found _____ ```
Test case 5 in Postman [Search: by name and location]	``` // TEST CASE: 5 (SEARCH BY NAME and LOCATION) // method: POST // URL: http://localhost:3030/searchticket // Body: raw JSON // { // "name": "o", // "location": "impact", // "date": "" // } ```



In console log:

```
--------------------------------
Search by name: o| Search by location: impact| Search by: date:
1 result(s) found
--------------------------------
```

Test case 6 in Postman [Search: by name and date]

```
// TEST CASE: 6 (SEARCH BY NAME and DATE)
// method: POST
// URL: http://localhost:3030/searchticket
// Body: raw JSON
// {
//   "name": "o",
//   "location": "",
```

```
//   "date": "2023"
// }
```

Ticketboo / **http://localhost:3030/searchticket**

Save

POST http://localhost:3030/searchticket **Send**

Params Auth Headers (8) Body Pre-req. Tests Settings Cookies

raw JSON Beautify

```
{
    "name": "o",
    "location": "",
    "date": "2023"
}
```

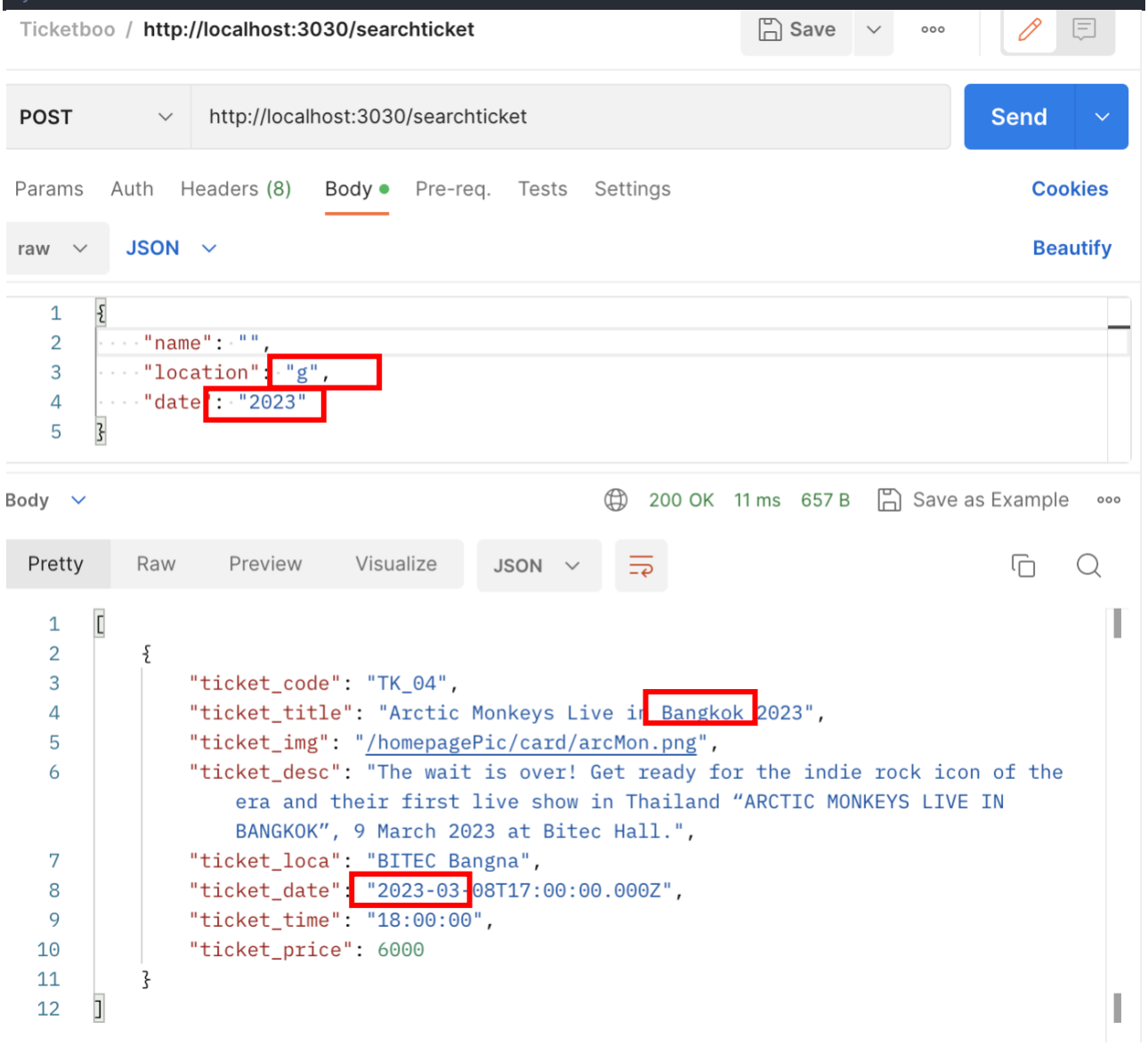
Body 200 OK 10 ms 2.16 KB Save as Example

Pretty Raw Preview Visualize JSON


```
[
    {
        "ticket_code": "TK_01",
        "ticket_title": "The 1975 Live in Bangkok",
        "ticket_img": "/homepagePic/card/1975con.jpg",
        "ticket_desc": "Mangosteenfest Presents \"THE 1975 : At their very best\"
            live in Bangkok 2023",
        "ticket_loca": "Impact Arena",
        "ticket_date": "2023-04-03T17:00:00.000Z",
        "ticket_time": "19:00:00",
        "ticket_price": 5000
    },
    {
        "ticket_code": "TK_02",
        "ticket_title": "WATERBOMB BANGKOK",
        "ticket_img": "/images/WaterBomb-Poster.jpg",
        "ticket_desc": "Definitely the best summer festival from South Korea,
            WATERBOMB is on the beginning its 1st WORLD TOUR starting from BANGKOK.
            Join the same team with your favorite artists and lead your team to the
            victory.",
        "ticket_loca": "Thunderdome Stadium",
        "ticket_date": "2023-04-12T17:00:00.000Z",
        "ticket_time": "12:00:00",
        "ticket_price": 4600
    },
```

<table>
<tr>
<td></td>
<td>
<pre>
22   {
23   "ticket_code": "TK_03",
24   "ticket_title": "ABOUT DAMN TIME",
25   "ticket_img": "/images/Ph1-Poster.jpg",
26   "ticket_desc": "IMC Live Global is proud to present 2023 pH-1 ABOUT DAMN TIME
              WORLD TOUR in Bangkok on 19 March 2022, 7 pm at the centralwOrld LIVE.",
27   "ticket_loca": "CentralwOrld LIVE",
28   "ticket_date": "2023-03-18T17:00:00.000Z",
29   "ticket_time": "19:00:00",
30   "ticket_price": 3800
31   },
32   {
33   "ticket_code": "TK_04",
34   "ticket_title": "Arctic Monkeys Live in Bangkok 2023",
35   "ticket_img": "/homepagePic/card/arcMon.png",
36   "ticket_desc": "The wait is over! Get ready for the indie rock icon of the era
              and their first live show in Thailand “ARCTIC MONKEYS LIVE IN BANGKOK”, 9
              March 2023 at Bitec Hall.",
37   "ticket_loca": "BITEC Bangna",
38   "ticket_date": "2023-03-08T17:00:00.000Z",
39   "ticket_time": "18:00:00",
40   "ticket_price": 6000
41   },
42   {
43   "ticket_code": "TK_05",
44   "ticket_title": "Pelupo Festival 2023",
45   "ticket_img": "/images/Pelupo-Poster.png",
46   "ticket_desc": "Leave your worries behind you and prepare to join us at PELUPO
              International Music Festival at The Fields at Siam Country Club in Chonburi
              on March 11, 2023.",
47   "ticket_loca": "Siam Country Club, Pattaya, Thailand",
48   "ticket_date": "2023-03-10T17:00:00.000Z",
49   "ticket_time": "15:00:00",
50   "ticket_price": 4000
51   }
52]
</pre>
In console log:
<pre>
Search by name: o| Search by location: | Search by: date: 2023
5 result(s) found
</pre>
</td>
</tr>
<tr>
<td>Test case 7 in Postman [Search: by location and date]</td>
<td>
<pre>
// TEST CASE: 7 (SEARCH BY LOCATION and DATE)
// method: POST
// URL: http://localhost:3030/searchticket
// Body: raw JSON
// {
//   "name": "",
</pre>
</td>
</tr>
</table>

```
//   "location": "g",
//   "date": "2023"
// }
```



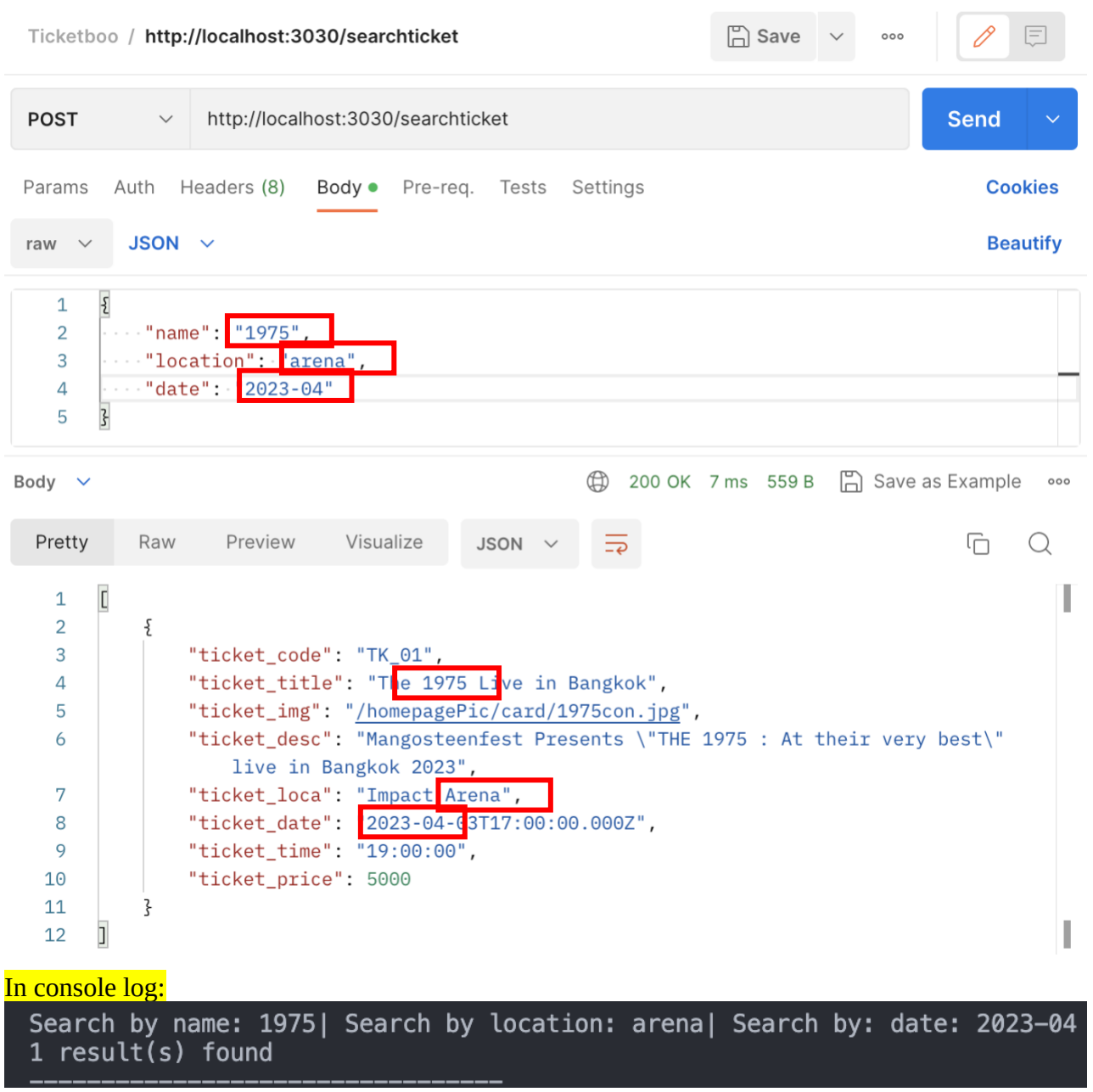
In console log:

```
---------------------------------
Search by name: | Search by location: g| Search by: date: 2023
1 result(s) found
---------------------------------
```

Test case 8 in Postman [Search: by name and	// TEST CASE: 8 (SEARCH BY NAME and LOCATION and DATE) // method: POST // URL: http://localhost:3030/searchticket // Body: raw JSON // {

location and date]

```
//   "name": "1975",
//   "location": "arena",
//   "date": "2023-04"
// }
```



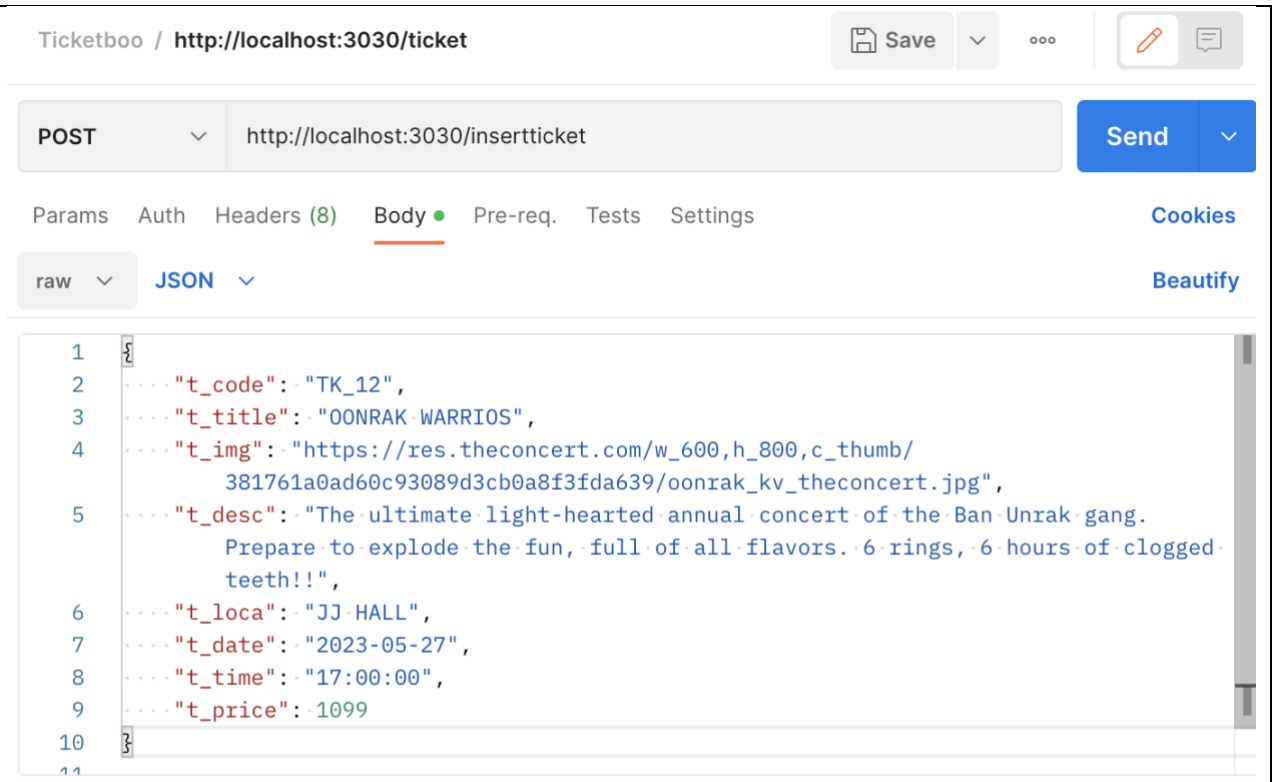
In console log:

```
Search by name: 1975| Search by location: arena| Search by: date: 2023-04
1 result(s) found
_____________________________________
```

File	Backend.js (Web Service: Product) [Insert Product]
Code	

```
/* TASK 2.2: Part 2*/
/* INSERT TICKET*/
router.post('/insertticket', function (req, res) {
    /* This is the request object, which represents the HTTP request sent by the client or frontend. */
    let t_code = req.body.t_code;
    let t_title = req.body.t_title;
    let t_img = req.body.t_img;
    let t_desc = req.body.t_desc;
    let t_loca = req.body.t_loca;
    let t_date = req.body.t_date;
    let t_time = req.body.t_time;
    let t_price = req.body.t_price;
    console.log(t_code + " " + t_title + " " + t_img + " " + t_desc + " " + t_loca + " " + t_date + " " + t_time + " " + t_price)
    /* The command we want to add data to the database. */
    let insertSql = `INSERT INTO ticket VALUES(` + t_code + `, '` + t_title + `', '` + t_img + `', '` + t_desc + `', '` + t_loca
+ `', '` + t_date + `', '` + t_time + `', '` + t_price + `');`;
    connection.query(insertSql, function (error, insertResults) {
        if (error) throw error;
        if (insertResults.affectedRows === 0) {
            console.log("No rows inserted.");
            return res.status(500).send("No rows inserted.");
        }
        console.log(`${insertResults.affectedRows} row(s) inserted`);
        res.json({ "message": "Ticket has been updated successfully." });
    });
});
```

Explanation	This is a route handler for the POST method on the "/insertticket" of a web application. When a user sends a POST request to the "/insertticket" endpoint, the handler function executes. The function expects a JSON payload in the request body that contains values for the properties "t_code", "t_title", "t_img", "t_desc", "t_loca", "t_date", "t_time", and "t_price". Then constructs an SQL INSERT statement that inserts a new row into the "ticket" table with the provided property values. If the INSERT statement is successful, the function returns a success response to the client.
Test case 1 in Postman	*// TEST CASE: 1* *// method: POST* *// URL: http://localhost:3030/insertticket* *// Body: raw JSON* *// {* *// "t_code": "TK_12",* *// "t_title": "OONRAK WARRIOS",* *// "t_img": "https://res.theconcert.com/w_600,h_800,c_thumb/381761a0ad60c93089d3cb0a8f3fda639/oonrak_kv_theconcert.jpg",* *// "t_desc": "The ultimate light-hearted annual concert of the Ban Unrak gang.Prepare to explode the fun, full of all flavors. 6 rings, 6 hours of clogged teeth!!",* *// "t_loca": "JJ HALL",* *// "t_date": "2023-05-27",* *// "t_time": "17:00:00",* *// "t_price": 1099* *// }*



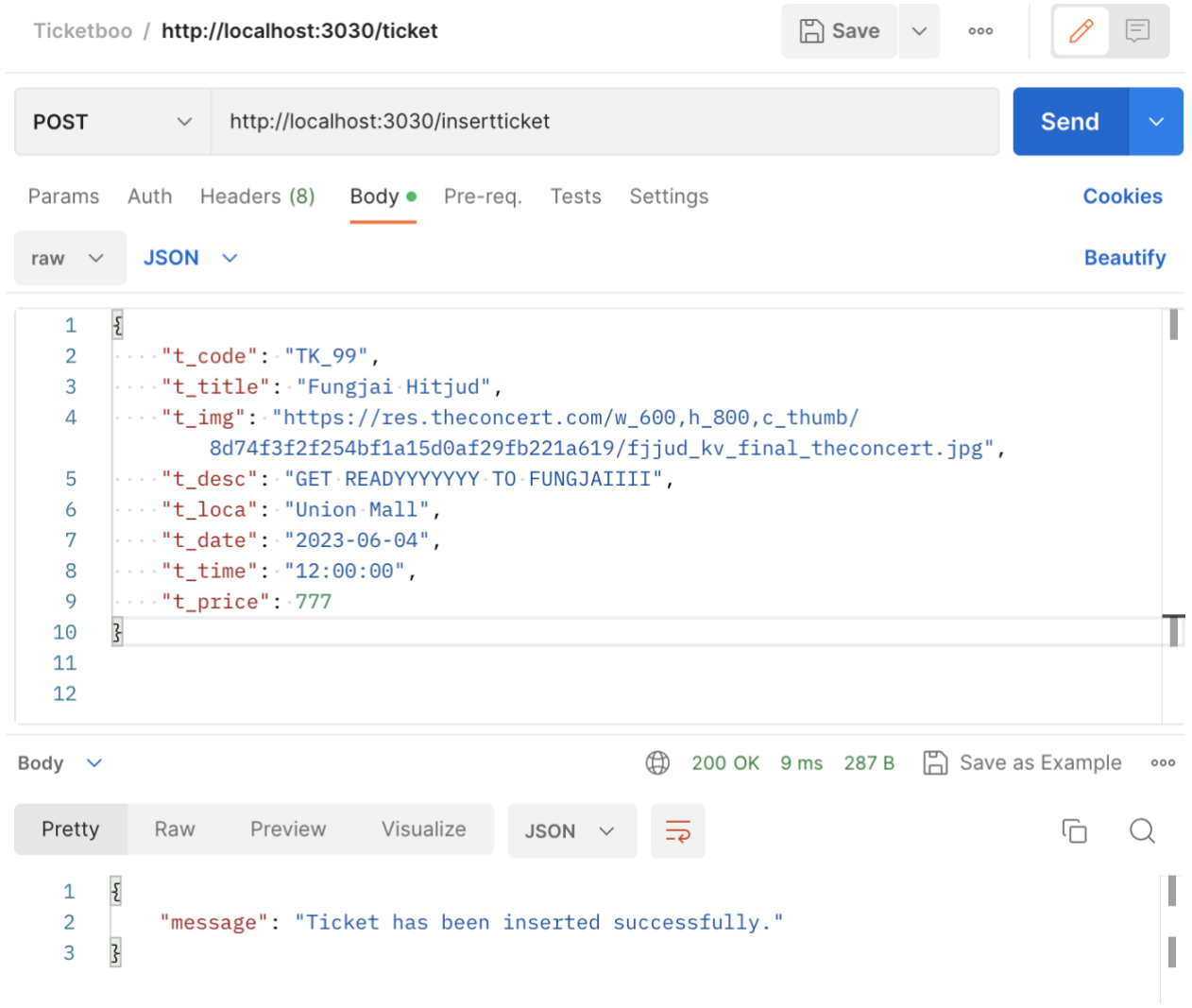
In console log

```
TK_12 OONRAK WARRIOS https://res.theconcert.com/w_600,h_800,c_thumb/381761a0ad60c93089d3cb0a8f3fda639/oonrak_kv_theconcert.jpg The ultimate light-hearted annual concert of the Ban Unrak gang.Prepare to explode the fun, full of all flavors. 6 rings, 6 hours of clogged teeth!! JJ HALL 2023-05-27 17:00:00 1099
1 row(s) inserted
```

Test case 2 in Postman

```
// TEST CASE: 2
// method: POST
// URL: http://localhost:3030/insertticket
// Body: raw JSON
// {
//     "t_code": "TK_99",
//     "t_title": "Fungjai Hitjud",
//     "t_img": "https://res.theconcert.com/w_600,h_800,c_thumb/8d74f3f2f254bf1a15d0af2
//9fb221a619/fjjud_kv_final_theconcert.jpg",
```

```
//     "t_desc": "GET READYYYYYYY TO FUNGJAIIII",
//     "t_loca": "Union Mall",
//     "t_date": "2023-06-04",
//     "t_time": "12:00:00",
//     "t_price": 777
// }
```



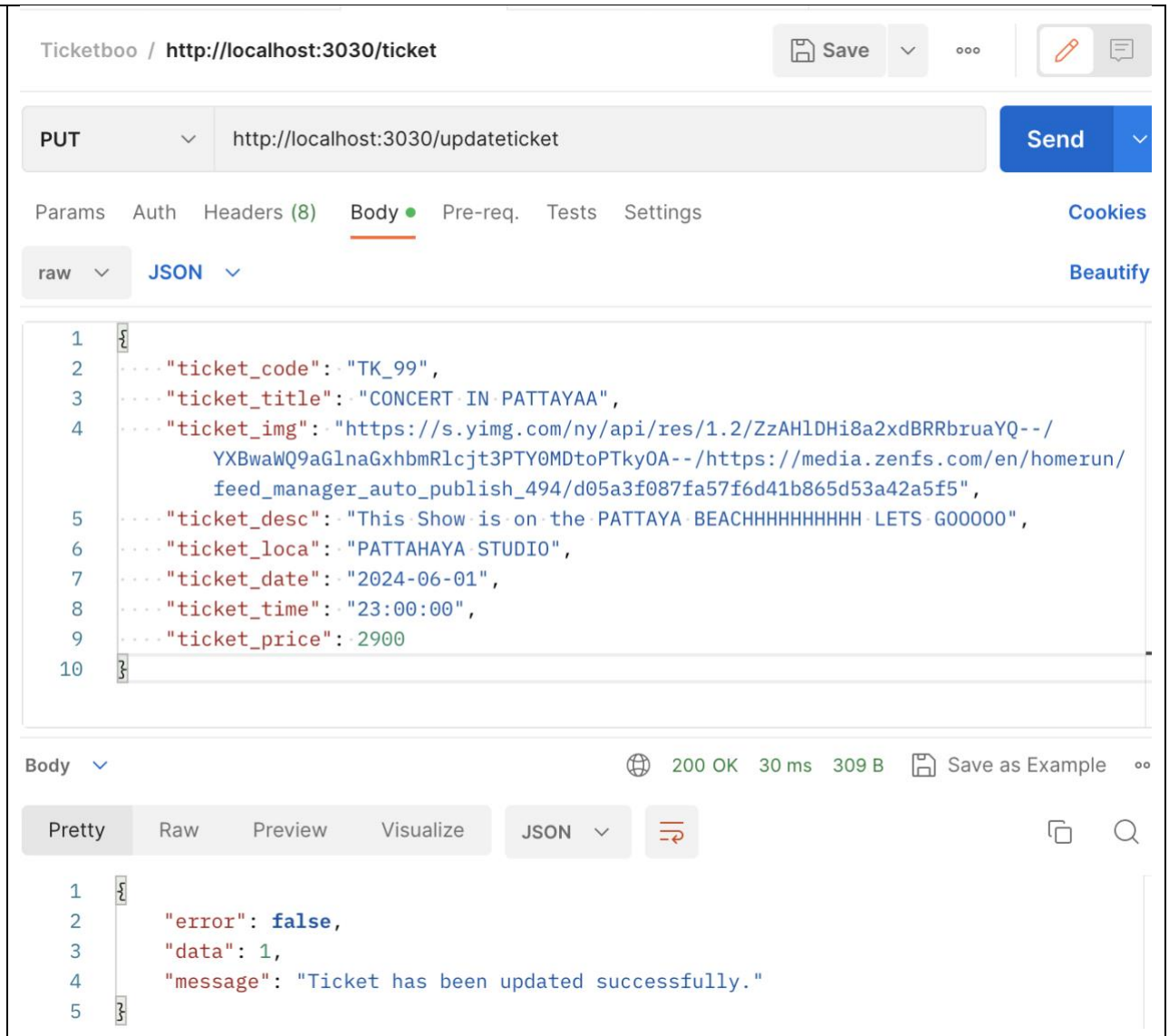
In console log

```
TK_99 Fungjai Hitjud https://res.theconcert.com/w_600,h_800,c_thumb/8d74f3f2f254bf1a15d
0af29fb221a619/fjjud_kv_final_theconcert.jpg GET READYYYYYYY TO FUNGJAIIII Union Mall 2
023-06-04 12:00:00 777
1 row(s) inserted
```

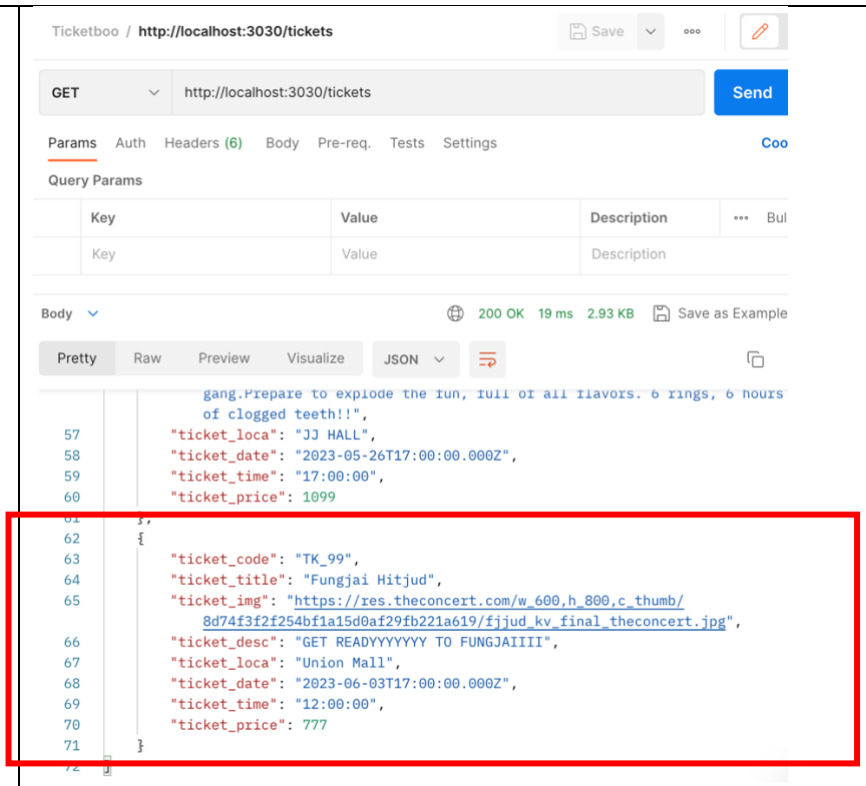
File	Backend.js (Web Service: Product) [Update Product]
Code	

```
/* TASK 2.3: Part 3 */
router.put('/updateticket', function (req, res) {
    /* This is the request object, which represents the HTTP request sent by the client or frontend. */
    let ticket_code = req.body.ticket_code;
    let ticket_title = req.body.ticket_title;
    let ticket_img = req.body.ticket_img;
    let ticket_desc = req.body.ticket_desc;
    let ticket_loca = req.body.ticket_loca;
    let ticket_date = req.body.ticket_date;
    let ticket_time = req.body.ticket_time;
    let ticket_price = req.body.ticket_price;

    if (!ticket_code) {
        return res.status(404).send({ error: ticket, message: 'Please provide ticket information' });
    }
    console.log(ticket_code)
    /* The command we want to update data to the database. */
    connection.query("UPDATE ticket SET ticket_title='" + ticket_title + "', ticket_img='" + ticket_img + "',ticket_desc='" +
ticket_desc + "',ticket_loca='" + ticket_loca + "',ticket_date='" + ticket_date + "',ticket_time='" + ticket_time +
"',ticket_price='" + ticket_price + "' WHERE ticket_code = '" + ticket_code + "'", function (err, results) {
        if (err) throw err;
        return res.send({ error: false, data: results.affectedRows, message: 'Ticket has been updated successfully.' })
    });
});
```

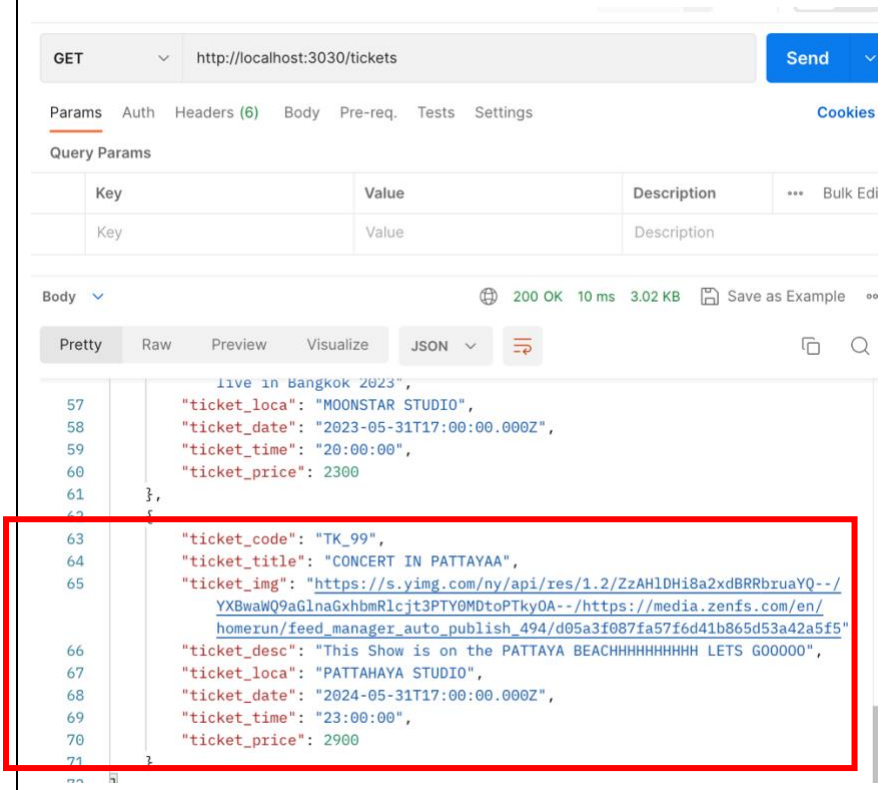
Explanation	This is a route handler for the PUT method on the "/updateticket" of a web application by extracting the necessary data from the request body using the "req.body" object. The extracted data includes the ticket code, title, image, description, location, date, time, and price. Then, it will check whether the ticket code is present in the request. If it's not, it returns a 404-status code with an error message. Next, the route performs an SQL update query to update the ticket in the database. It uses the extracted data to set the new values for the corresponding columns in the "ticket" table. The query updates the ticket with the matching ticket code.
Test case 1 in Postman	*// TEST CASE: 1* *// method: PUT* *// URL: http://localhost:3030/updateticket* *// Body: raw JSON* *// {* *// "ticket_code": "TK_99",* *// "ticket_title": "CONCERT IN PATTAYAA",* *// "ticket_img": "https://s.yimg.com/ny/api/res/1.2/ZzAHlDHi8a2xdBRRbruaYQ--* */YXBwaWQ9aGlnaGxhbmRlcjt3PTY0MDtoPTkyOA--* */https://media.zenfs.com/en/homerun/feed_manager_auto_publish_494/d05a3f087fa* *//57f6d41b865d53a42a5f5",* *// "ticket_desc": "This Show is on the PATTAYA BEACHHHHHHHHHH LETS GOOOOO",* *// "ticket_loca": "PATTAHAYA STUDIO",* *// "ticket_date": "2024-06-01",* *// "ticket_time": "23:00:00",* *// "ticket_price": 2900* *// }*



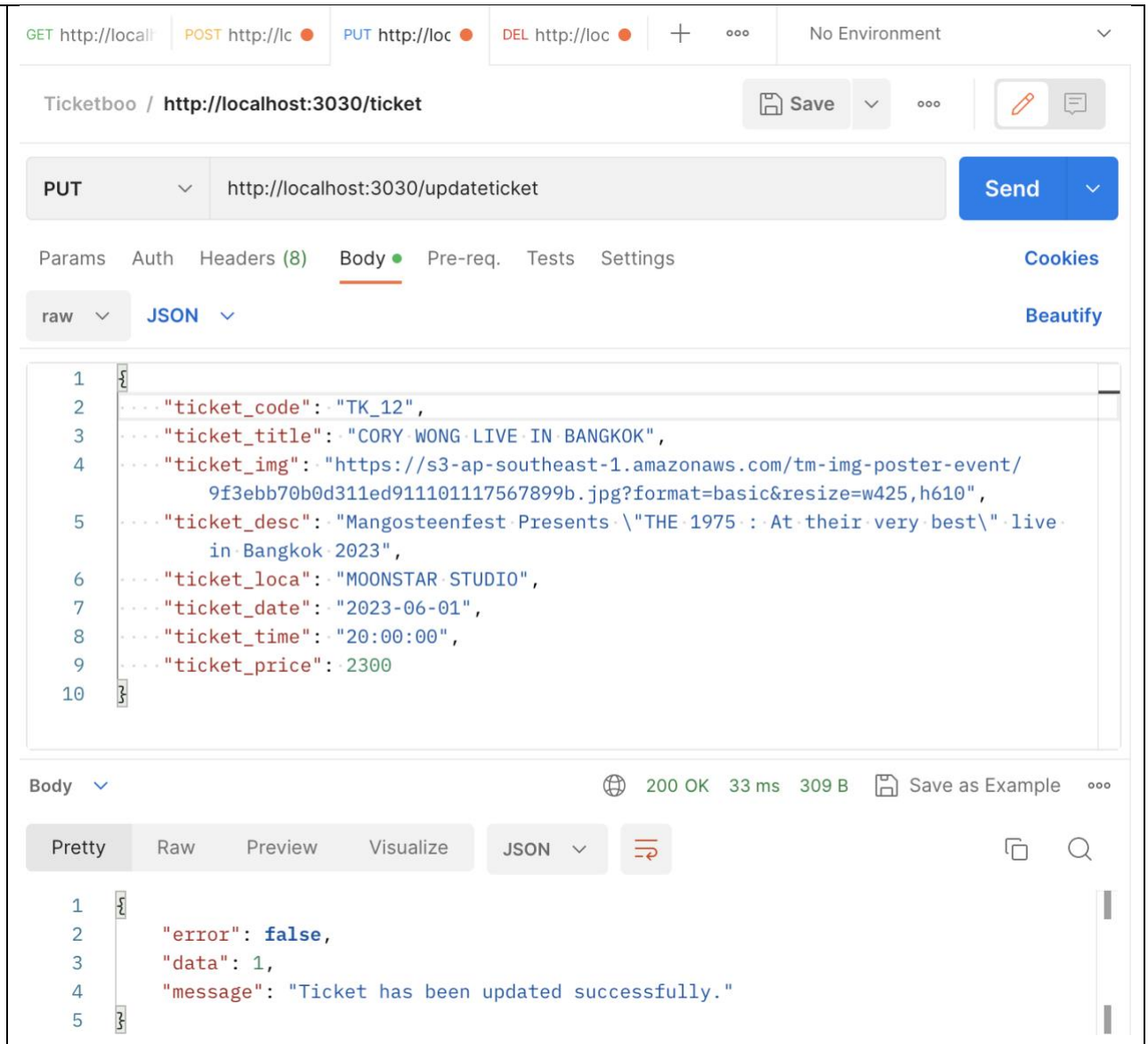
[Before update]



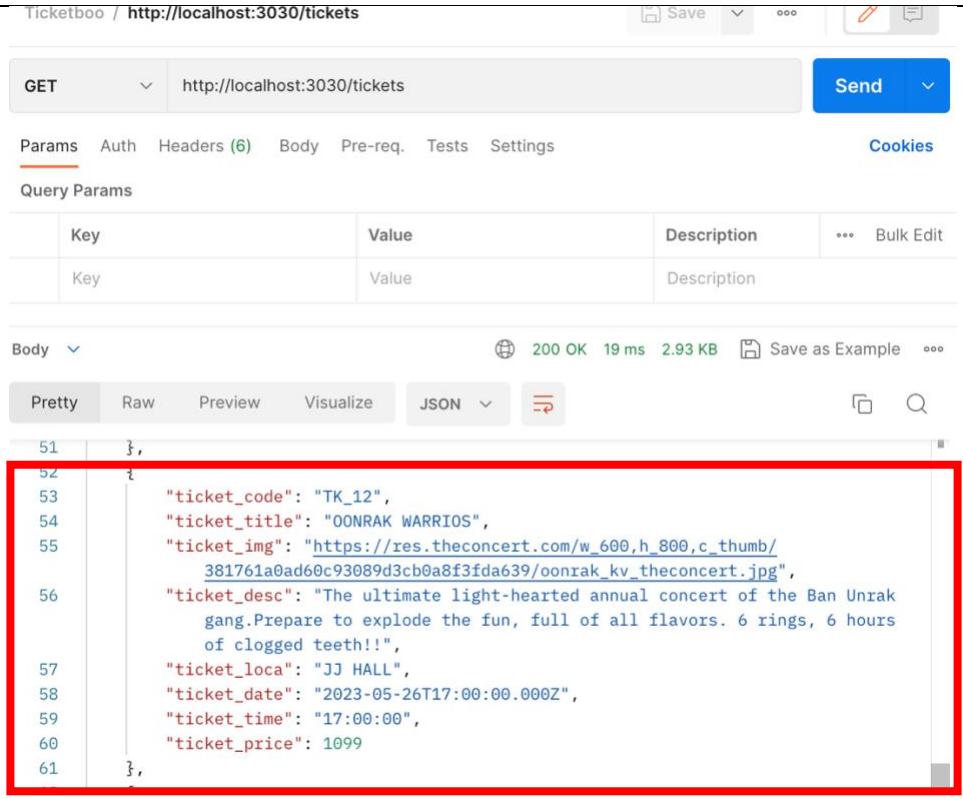
[After update]



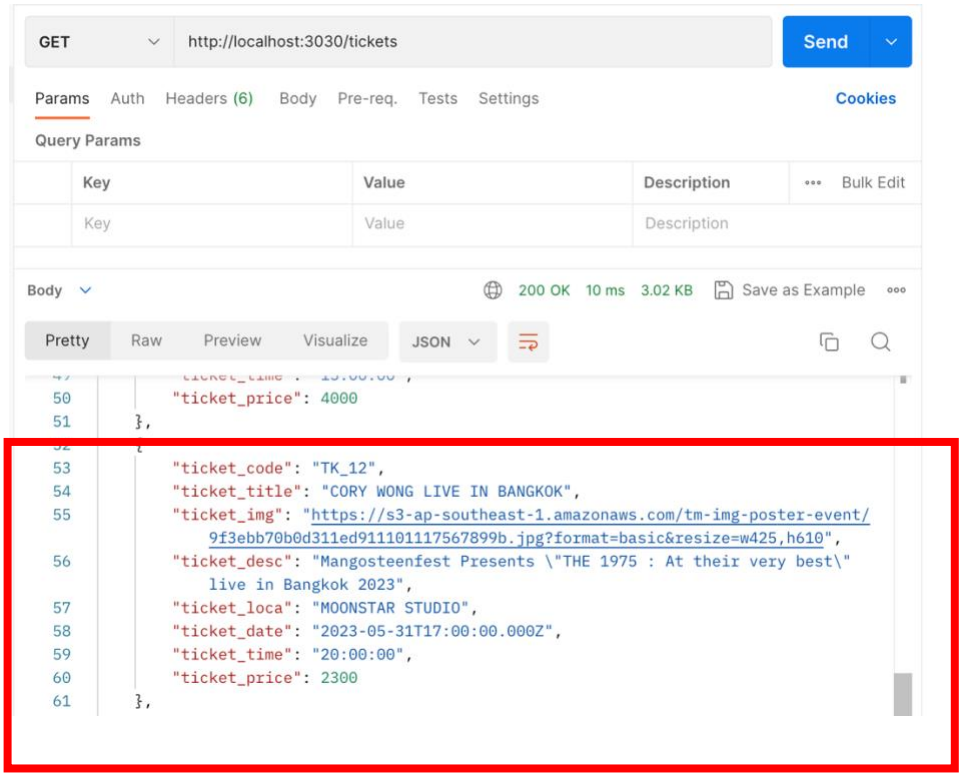
Test case 2 in Postman	// TEST CASE: 2 // method: PUT // URL: http://localhost:3030/updateticket // Body: raw JSON // { // "ticket_code": "TK_12", // "ticket_title": "CORY WONG LIVE IN BANGKOK", // "ticket_img": "https://s3-ap-southeast-1.amazonaws.com/tm-img-poster-event/9f3ebb70b0d311ed911101117567899b.jpg?format=basic&resize=w425,h610", // "ticket_desc": "Mangosteenfest Presents \"THE 1975 : At their very best\" live in Bangkok 2023", // "ticket_loca": "MOONSTAR STUDIO", // "ticket_date": "2023-06-01", // "ticket_time": "20:00:00", // "ticket_price": 2300 // }

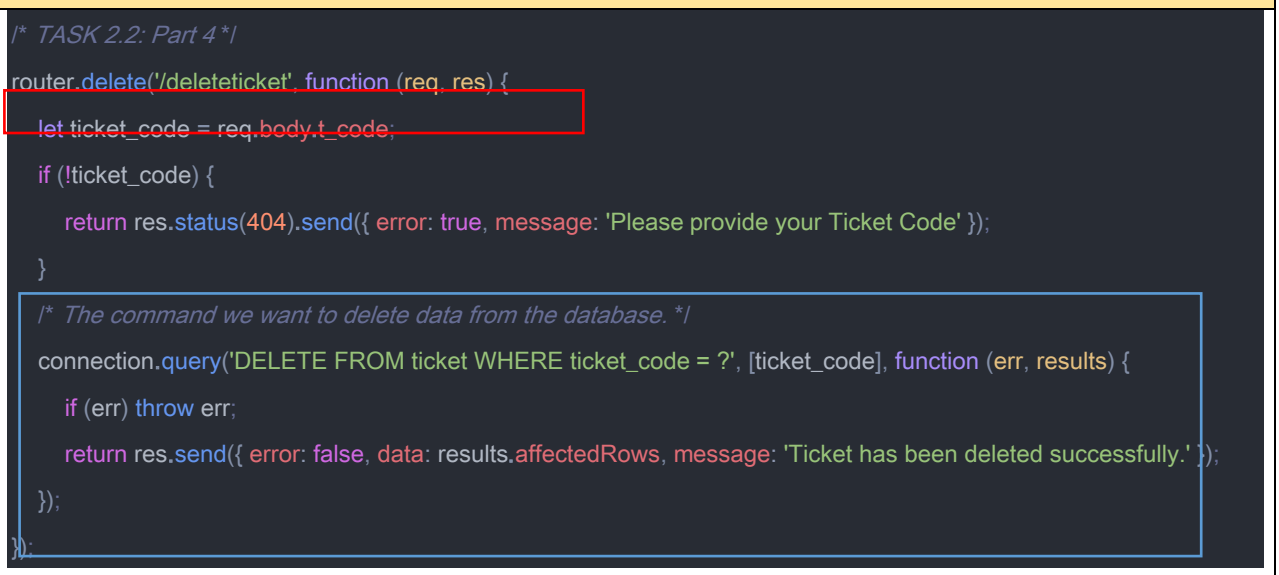


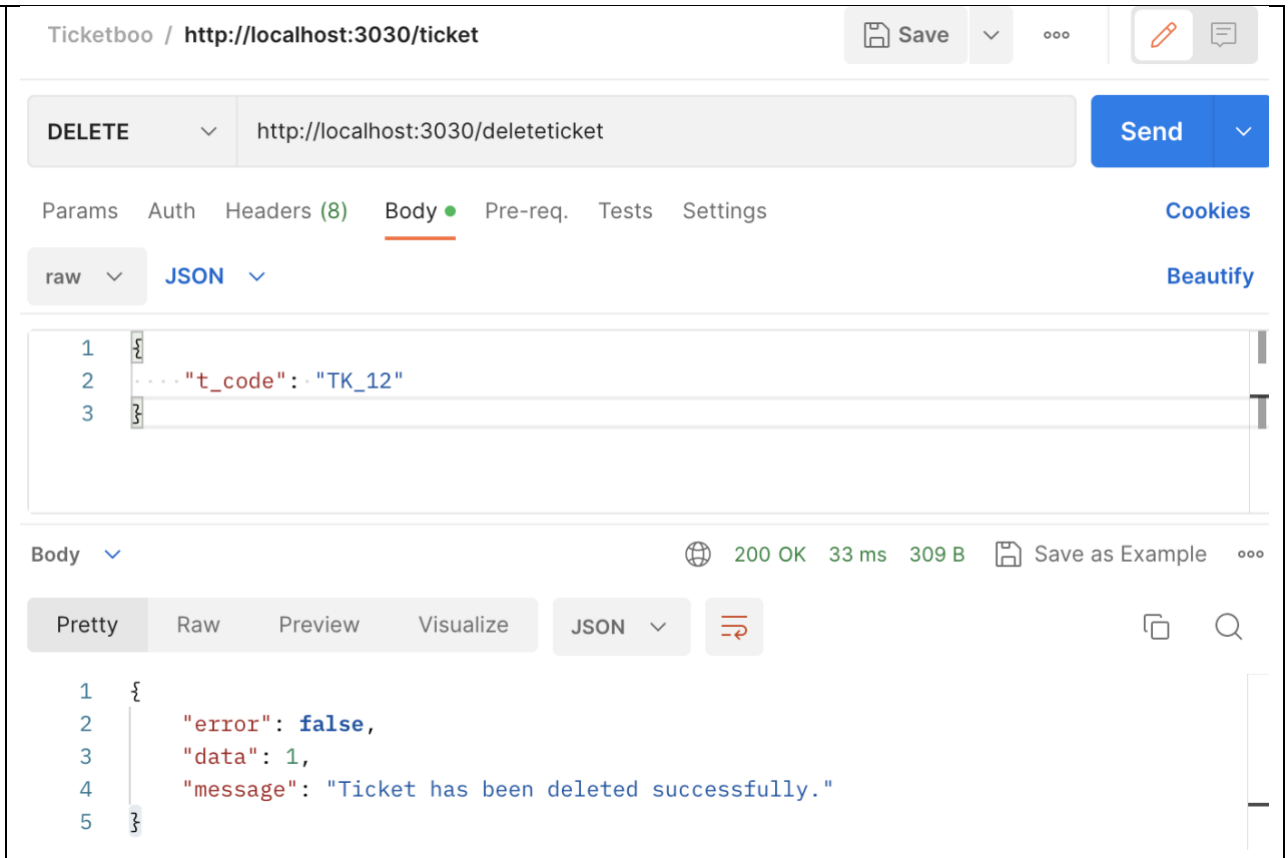
[Before update]

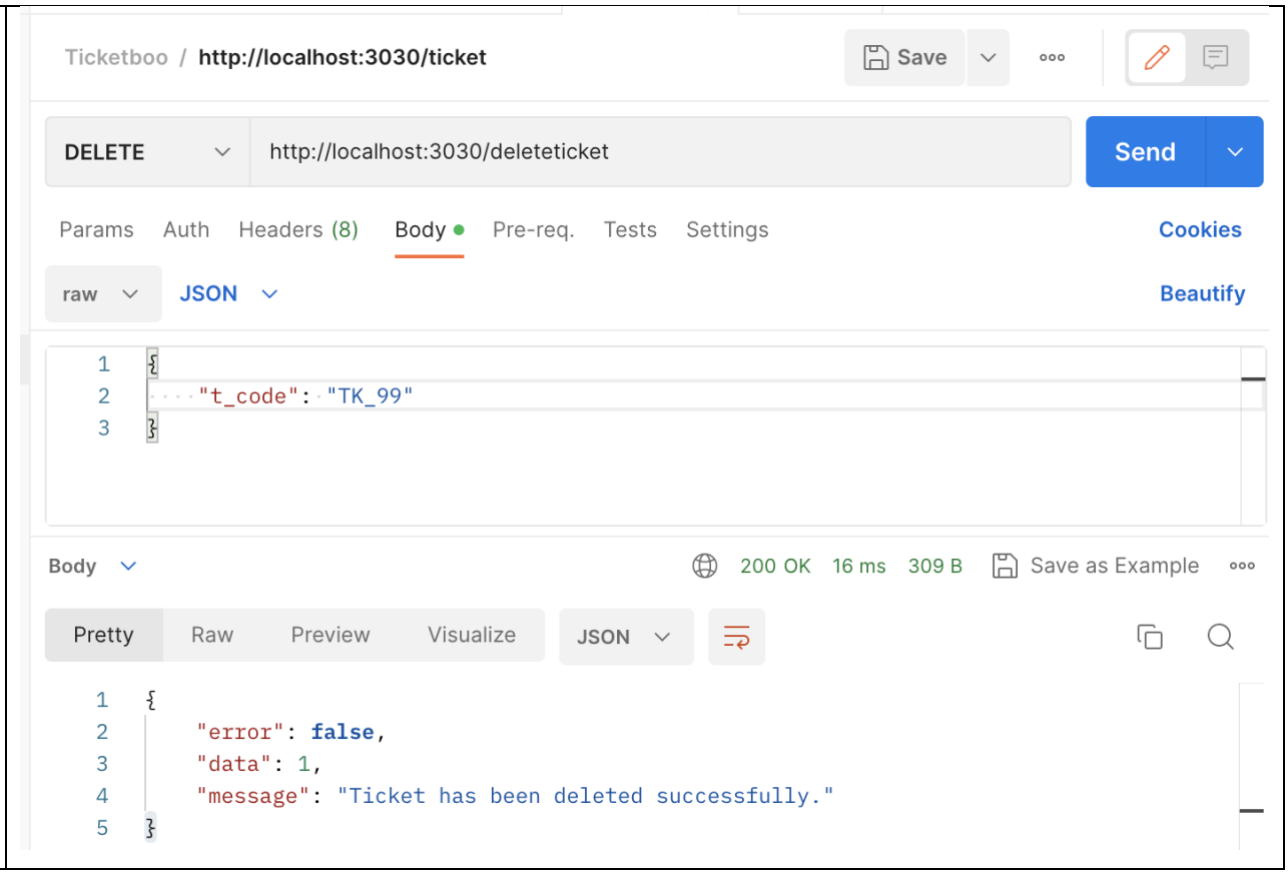


[After update]



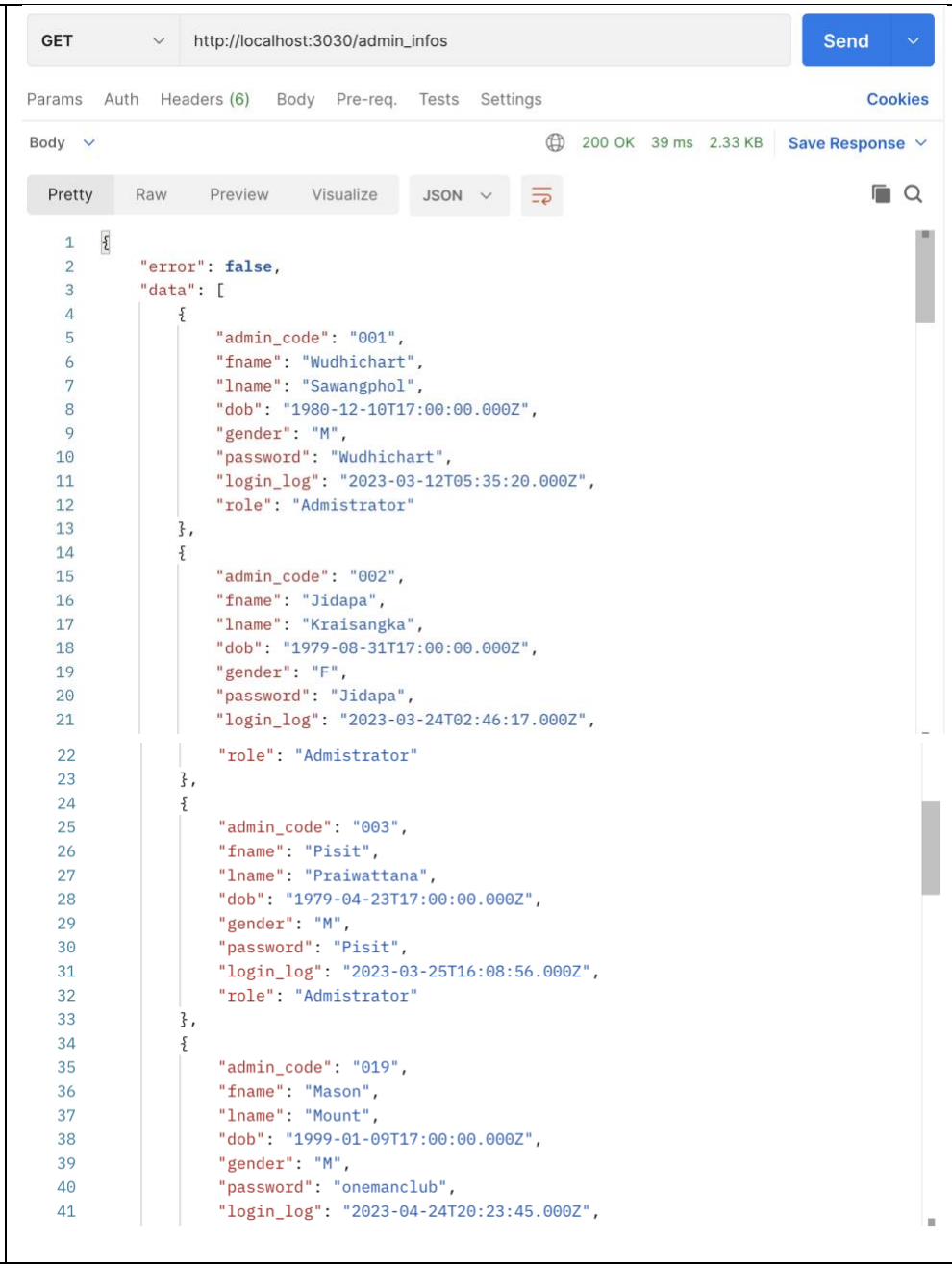
File	Backend.js (Web Service: Product) [Delete Product]
Code	 ``` /* TASK 2.2: Part 4 */ router.delete('/deleteticket', function (req, res) { let ticket_code = req.body.t_code; if (!ticket_code) { return res.status(404).send({ error: true, message: 'Please provide your Ticket Code' }); } /* The command we want to delete data from the database. */ connection.query('DELETE FROM ticket WHERE ticket_code = ?', [ticket_code], function (err, results) { if (err) throw err; return res.send({ error: false, data: results.affectedRows, message: 'Ticket has been deleted successfully.' }); }); }); ```
Explanation	This is a route handler for the DELETE method on the "/deleteticket" of a web application. When a user sends a DELETE request to the "/deleteticket", the handler function executes. This route can be used to delete a ticket record from the database, based on the `**ticket_code**` provided in the request body. If the "t_code" property is present, the function executes a SQL query to delete the row in the "ticket" table where the "ticket_code" column matches the value of the "t_code" property. If an error occurs while executing the query, the function throws an error. Otherwise, the function sends a JSON response containing the number of affected rows and a success message indicating that the ticket has been deleted successfully.
Test case 1 in Postman	 ``` // TEST CASE: 1 //method: DELETE //URL: http://localhost:3030/deleteticket // Body: raw JSON // { // "t_code": "TK_12" // } ```

	
Test case 2 in Postman	// TEST CASE: 2 // method: DELETE // URL: http://localhost:3030/deleteticket // Body: raw JSON // { // "t_code": "TK_99" // }


Ticketboo / http://localhost:3030/ticket
Save
DELETE
http://localhost:3030/deleteticket
Send
Params
Auth
Headers (8)
Body
Pre-req.
Tests
Settings
Cookies
raw
JSON
Beautify
{
"t_code": "TK_99"
}
Body
200 OK
16 ms
309 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
{
"error": false,
"data": 1,
"message": "Ticket has been deleted successfully."
}

File: Backend.js [Web Service: Administrator] [Select All Admin user]	
Code	`//GET ADMIN USER` `router.get('/admin_infos', function (req, res) {` `  connection.query('SELECT * FROM admin_info', (err, results) => {` `  if (err) throw err;` `  return res.send({ error: false, data: results, message: 'Admin User list.' });` `  });` `});`
Explanation	This is handling the get method at path '/admin_infos' and responding by shows all the admin as following the table admin_info.
Test case:	TEST CASE: method: GET URL: http://localhost:3030/admin_infos

Postman:


GET
http://localhost:3030/admin_infos
Send
Params Auth Headers (6) Body Pre-req. Tests Settings
Cookies
Body
200 OK 39 ms 2.33 KB
Save Response
Pretty Raw Preview Visualize JSON
{
"error": false,
"data": [
{
"admin_code": "001",
"fname": "Wudhichart",
"lname": "Sawangphol",
"dob": "1980-12-10T17:00:00.000Z",
"gender": "M",
"password": "Wudhichart",
"login_log": "2023-03-12T05:35:20.000Z",
"role": "Admistrator"
},
{
"admin_code": "002",
"fname": "Jidapa",
"lname": "Kraisangka",
"dob": "1979-08-31T17:00:00.000Z",
"gender": "F",
"password": "Jidapa",
"login_log": "2023-03-24T02:46:17.000Z",
"role": "Admistrator"
},
{
"admin_code": "003",
"fname": "Pisit",
"lname": "Praiwattana",
"dob": "1979-04-23T17:00:00.000Z",
"gender": "M",
"password": "Pisit",
"login_log": "2023-03-25T16:08:56.000Z",
"role": "Admistrator"
},
{
"admin_code": "019",
"fname": "Mason",
"lname": "Mount",
"dob": "1999-01-09T17:00:00.000Z",
"gender": "M",
"password": "onemanclub",
"login_log": "2023-04-24T20:23:45.000Z",

File: Backend.js
[Web Service: Administrator]
[Select an Admin user by admin_code]

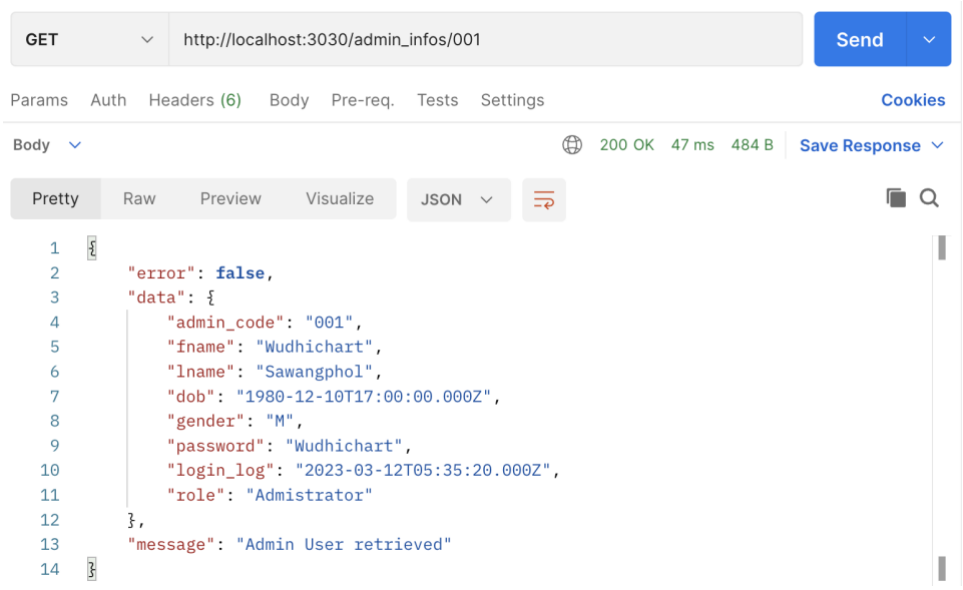
Code

```
//SELECT a ADMIN USER
router.get('/admin_infos/:admin_code', function (req, res) {
    let admin_code = req.params.admin_code;
    if (!admin_code) {
        return res.status(404).send({ error: true, message: 'Please provide admin code id.' });
    }
    connection.query('SELECT * FROM admin_info where admin_code=?', admin_code, (err, results) => {
        if (err) throw err;
        res.send({ error: false, data: results[0], message: 'Admin User retrieved' });
        console.log(`Sending admin result of admin_code = ${admin_code}`);
    });
});
```

Test case 1:

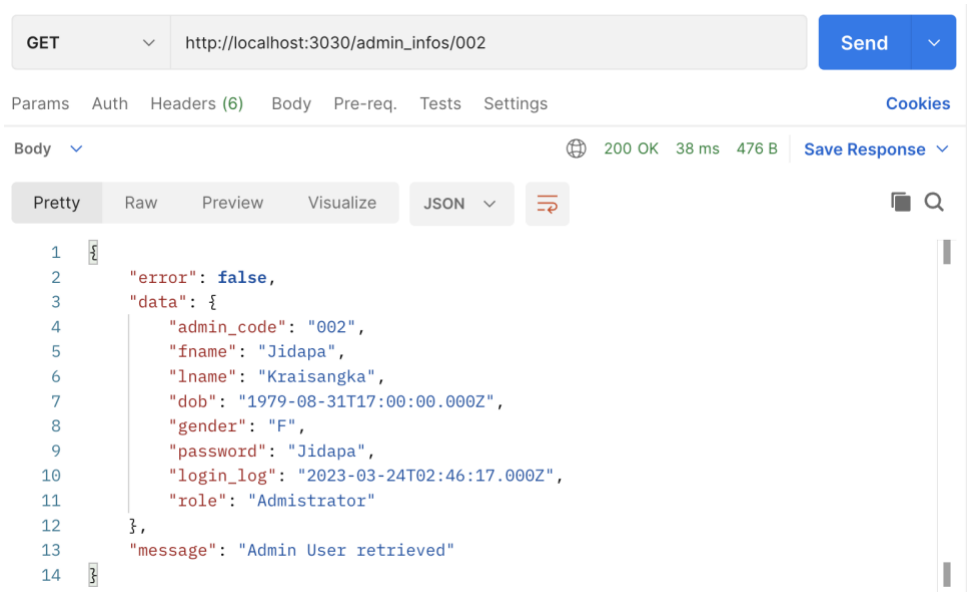
```
TEST CASE: 1
method: GET
URL: http://localhost:3030/admin_infos/001
OUTPUT: It will print result of admin_code = 001
```

Postman 1:

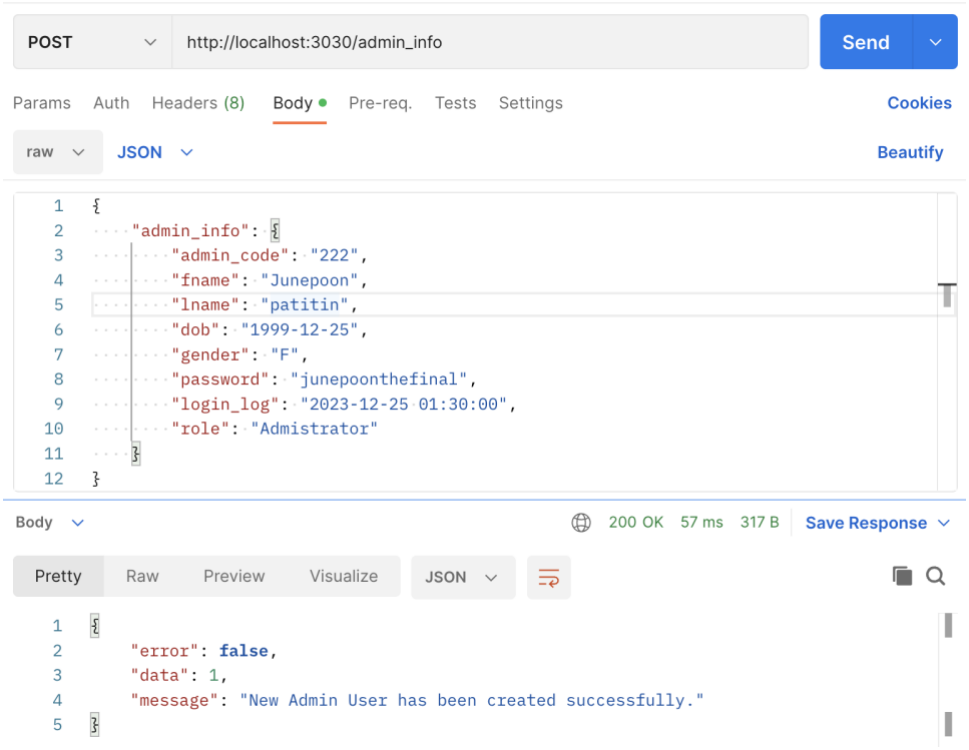


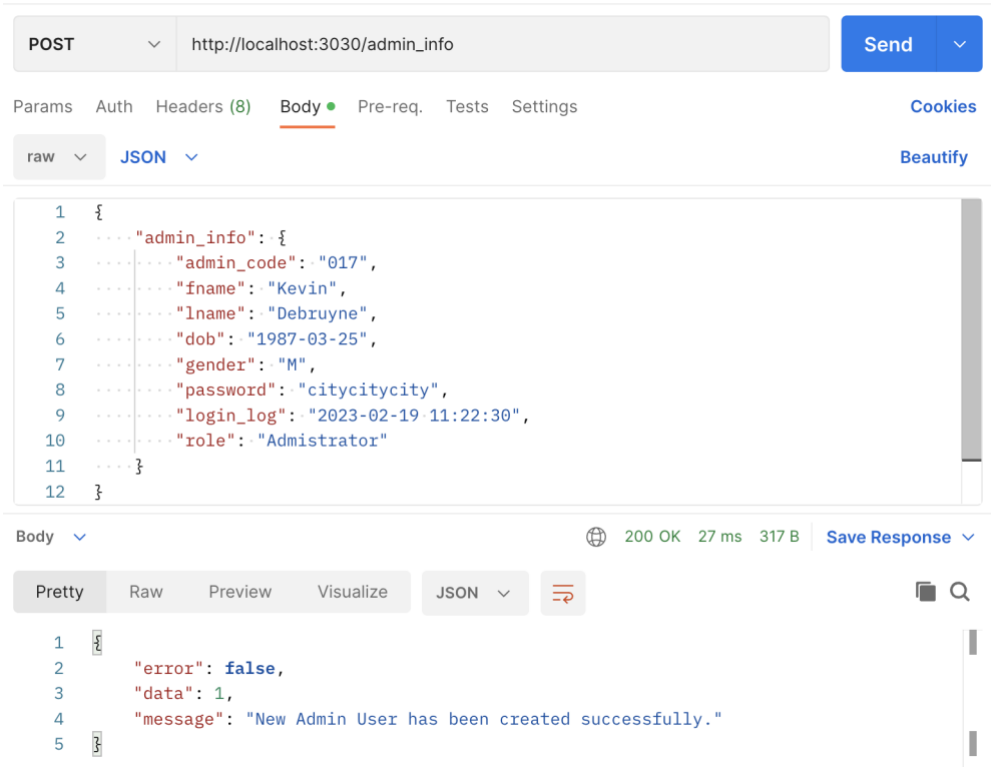
Test case 2:

```
TEST CASE: 2
method: GET
```

	URL: http://localhost:3030/admin_infos/002 OUTPUT: It will print result of admin_code = 002
Postman 2:	
Explanation	This is handling the get method at path '/admin_infos/:admin_code' and if there's no admin code, it will show the message "please provide admin code id.". Otherwise, selecting the row of admin_info that matches with the input.

File: Backend.js [Web Service: Administrator] [Insert Admin user]	
Code	/* TASK 2.3: Part 2 INSERT ADMINISTRATOR */ router.post('/admin_info', function (req, res) { let admin_info = req.body.admin_info; console.log(admin_info); if (!admin_info) { return res.status(404).send({ error: true, message: 'please provide admin user information' }); } connection.query("INSERT INTO admin_info SET ? ", admin_info, (err, results) => { if (err) throw err; return res.send({ error: false, data: results.affectedRows, message: 'New Admin User has been created successfully.' }); }); });
Test case 1:	TEST CASE: 1 method: POST URL: http://localhost:3030/admin_info { "admin_info": { "admin_code": "222", "fname": "Junepoon", "lname": "patitin", "dob": "1999-12-25", "gender": "F", "password": "junepoonthefinal", "login_log": "2023-12-25 01:30:00", "role": "Admistrator" } }

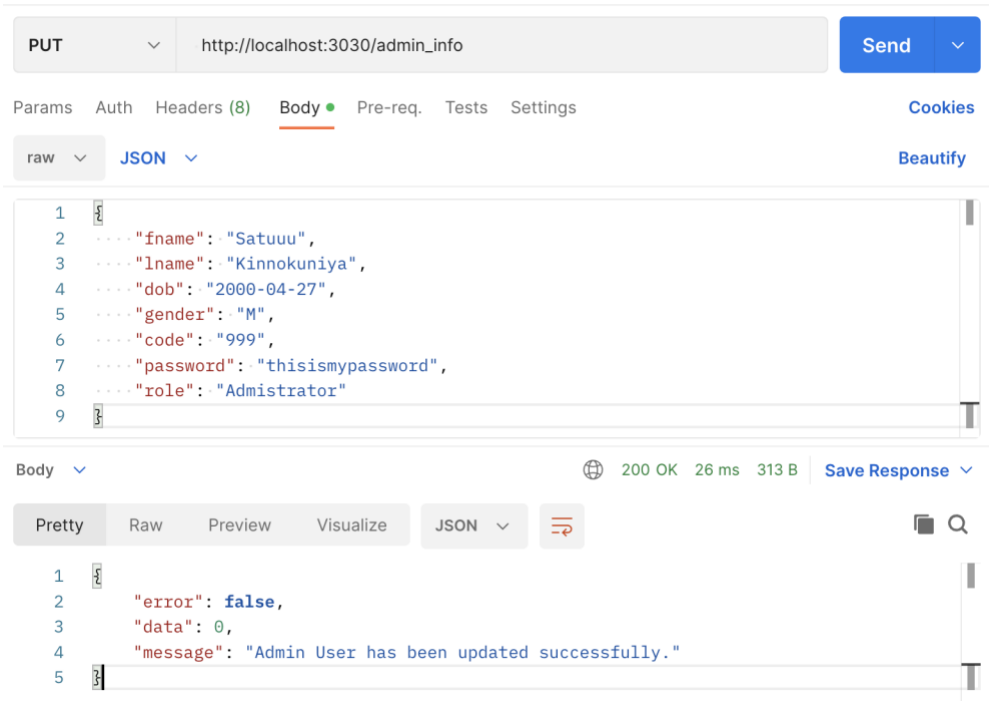
Postman 1:	
Test case 2:	TEST CASE: 2 method: POST URL: http://localhost:3030/admin_info { "admin_info": { "admin_code": "017", "fname": "Kevin", "lname": "Debruyne", "dob": "1987-03-25", "gender": "M", "password": "citycitycity", "login_log": "2023-02-19 11:22:30", "role": "Admistrator" } }

Postman 2:	
Explanation	This is handling the post method at path '/admin_info' and insert the admin user information in the table admin_info in sql. It will print out the message "New admin user has been created successfully" when admin provides the admin user information correctly. Otherwise, it will print "please provide the admin user information"

File: Backend.js [Web Service: Administrator] [Update Admin user]	
Code	/* TASK 2.3: Part 3: UPDATE USER */ router.put('/admin_info', function (req, res) { let fname = req.body.fname; let lname = req.body.lname; let dob = req.body.dob; let gen = req.body.gender; let code = req.body.code; let pass = req.body.password; let role = req.body.role; console.log(fname + " " + lname + " " + dob + " " + gen + " " + code + " " + pass + " " + role) if (!code) { return res.status(404).send({ error: code, message: 'Please provide admin user information' }); } const currentDate = new Date(); const date = currentDate.getFullYear() + "-" + (currentDate.getMonth() + 1) + "-" + currentDate.getDate(); const time = currentDate.getHours() + ":" + currentDate.getMinutes() + ":" + currentDate.getSeconds(); console.log(code) connection.query("UPDATE admin_info SET fname='" + fname + "',lname= '" + lname + "', dob='" + dob + "', gender='" + gen + "',password='" + pass + "',login_log='" + date + " " + time + "' ,role='" + role + "' WHERE admin_code ='" + code + "'", (err, results) => { if (err) throw err; return res.send({ error: false, data: results.affectedRows, message: 'Admin User has been updated successfully.' }) }); });
Test case 1:	TEST CASE: 1 method: PUT URL: http://localhost:3030/admin_info { "fname": "Satuuu", "lname": "Kinnokuniya", "dob": "2000-04-27",

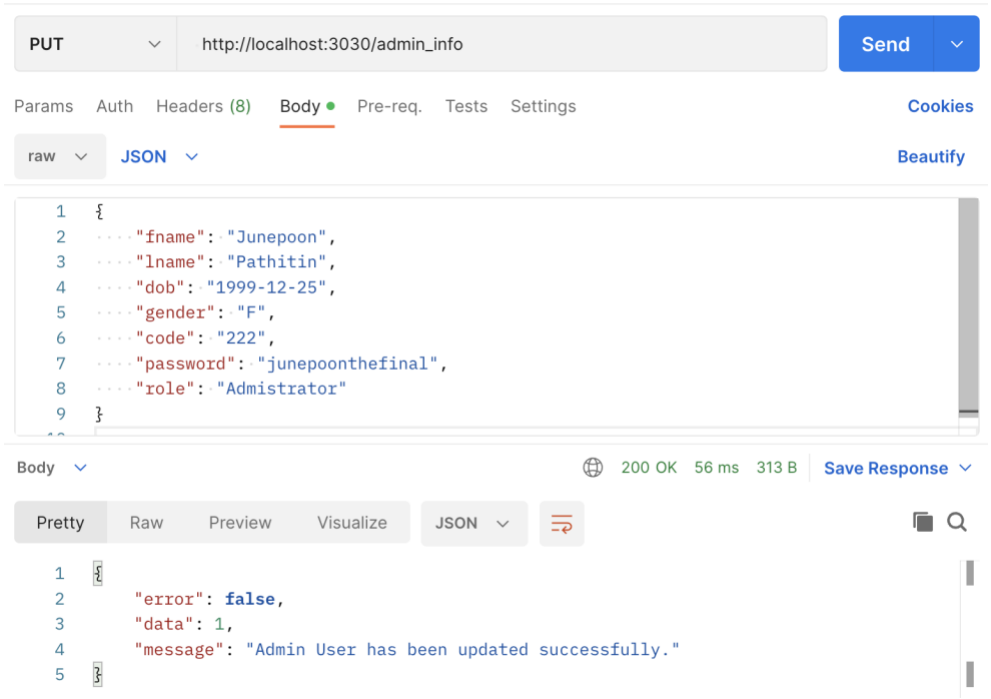
<table>
<tr><td></td><td>

```
    "gender": "M",
    "code": "999",
    "password": "thisismypassword",
    "role": "Admistrator"
}
```

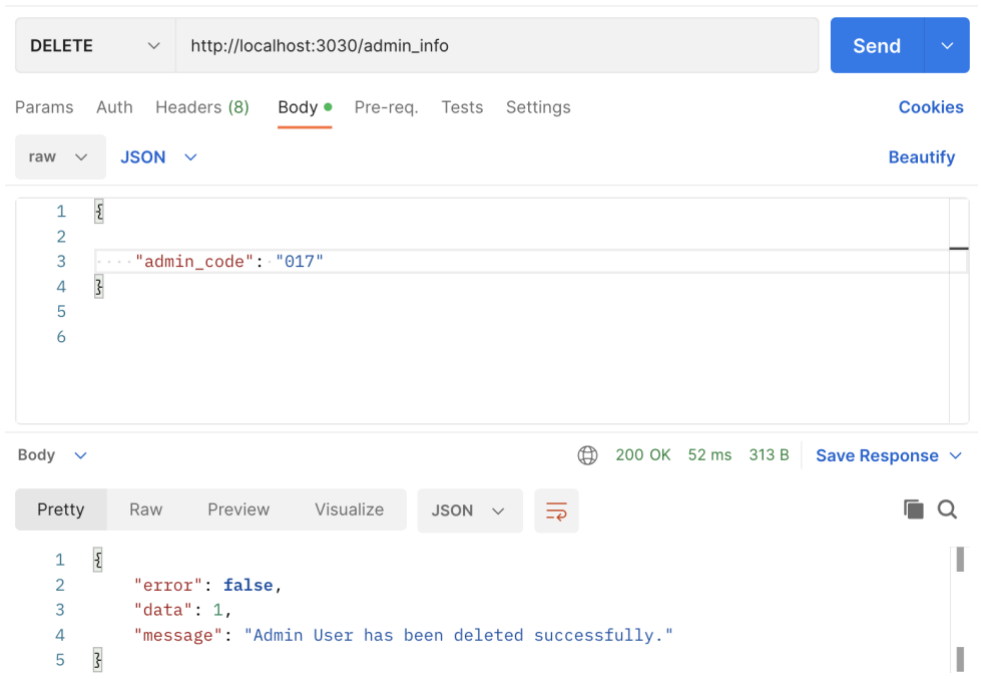
</td></tr>
<tr><td>Postman 1:</td><td> 
</td></tr>
<tr><td>Test case 2:</td><td>

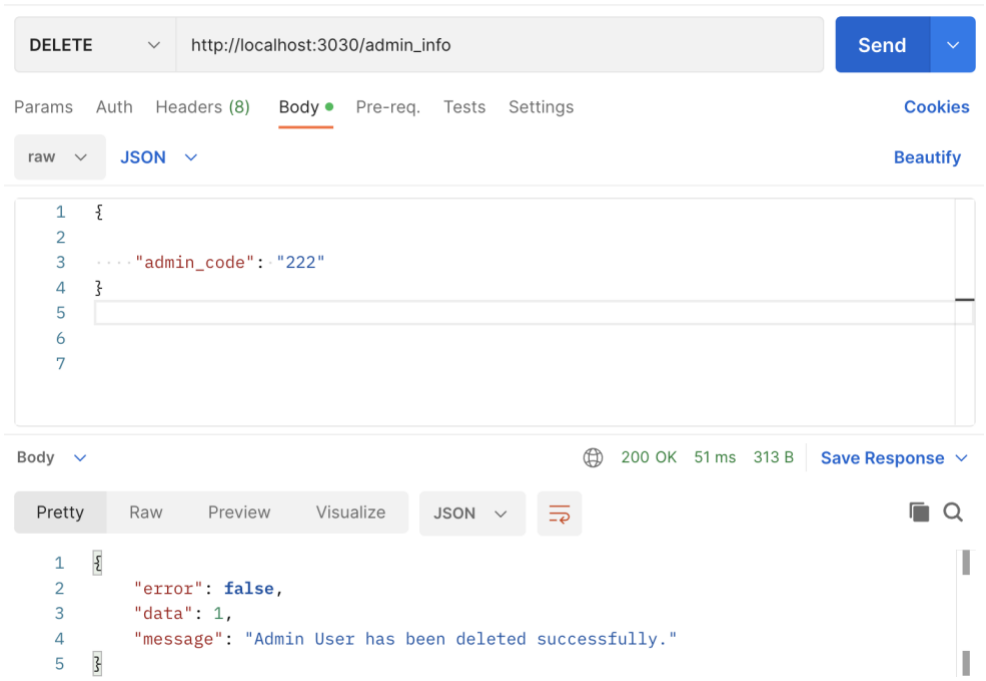
```
TEST CASE: 2
method: PUT
URL: http://localhost:3030/admin_info
{
    "fname": "Junepoon",
    "lname": "Pathitin",
    "dob": "1999-12-25",
    "gender": "F",
    "code": "222",
    "password": "junepoonthefinal",
    "role": "Admistrator"
}
```

</td></tr>
</table>

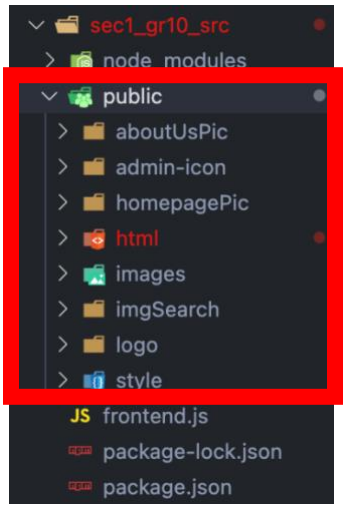
Postman 2:	
Explanation	This is handling the put method at path '/admin_info' and update the admin user information in the table admin_info in sql. It will print out the message "New admin user has been updated successfully" when admin provides the admin user information correctly. Otherwise, it will print "please provide the admin user information"

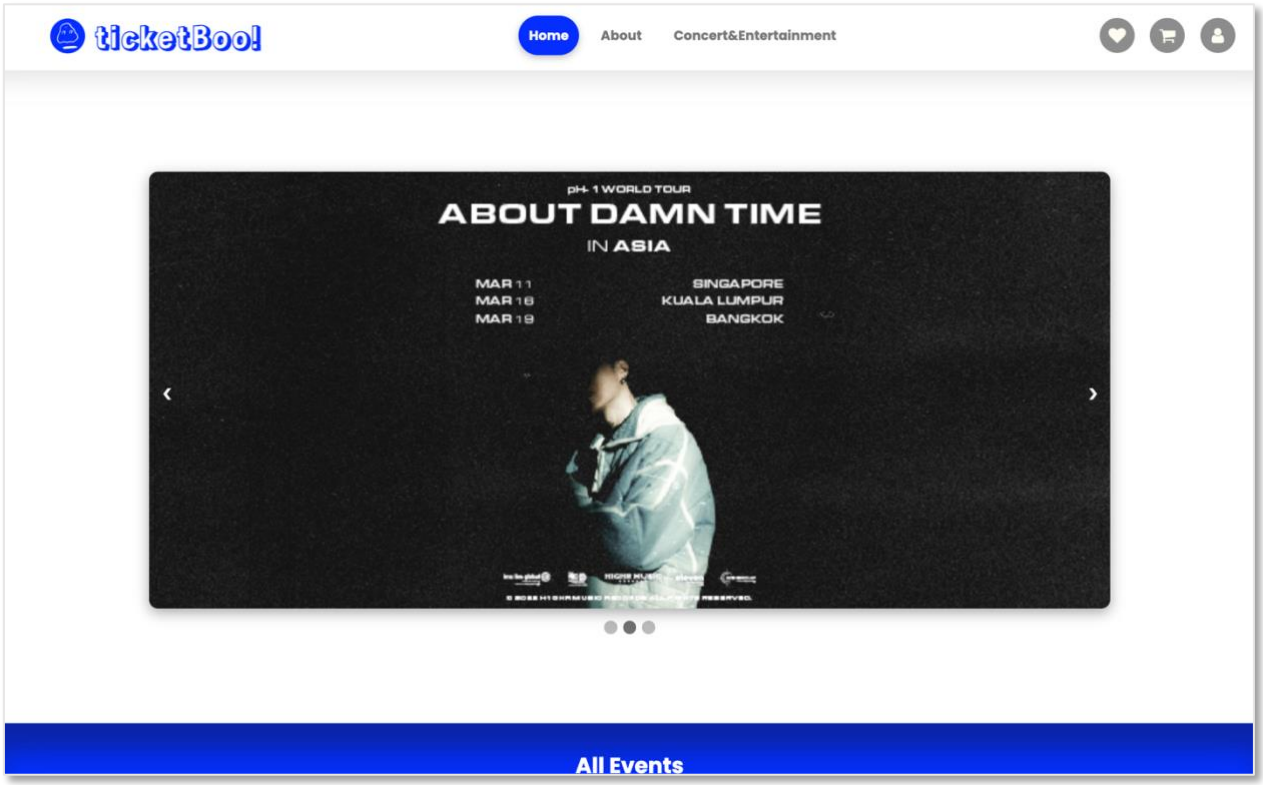
File: Backend.js [Web Service: Administrator] [Delete Admin user]	
Code	/* TASK 2.3: Part 4: DELETE USER */ router.delete('/admin_info', function (req, res) { let admin_code = req.body.admin_code; if (!admin_code) { return res.status(404).send({ error: true, message: 'Please provide admin_code' }); } connection.query('DELETE FROM admin_info WHERE admin_code = ' + admin_code, (err, results) => { if (err) throw err; return res.send({ error: false, data: results.affectedRows, message: 'Admin User has been deleted successfully.' }); }); });
Test case 1:	TEST CASE: 1 method: DELETE URL: http://localhost:3030/admin_info { "admin_code": "017" }

Postman 1:	
Test case 2:	TEST CASE: 2 method: DELETE URL: http://localhost:3030/admin_info { "admin_code": "222" }

Postman 2:	
Explanation	This is handling the delete method at path '/admin_info' and delete the admin user information in the table admin_info in sql by matching the admin_code. It will print out the message "Admin user has been deleted successfully" when admin provides the admin user information correctly. Otherwise, it will print "please provide the admin code"

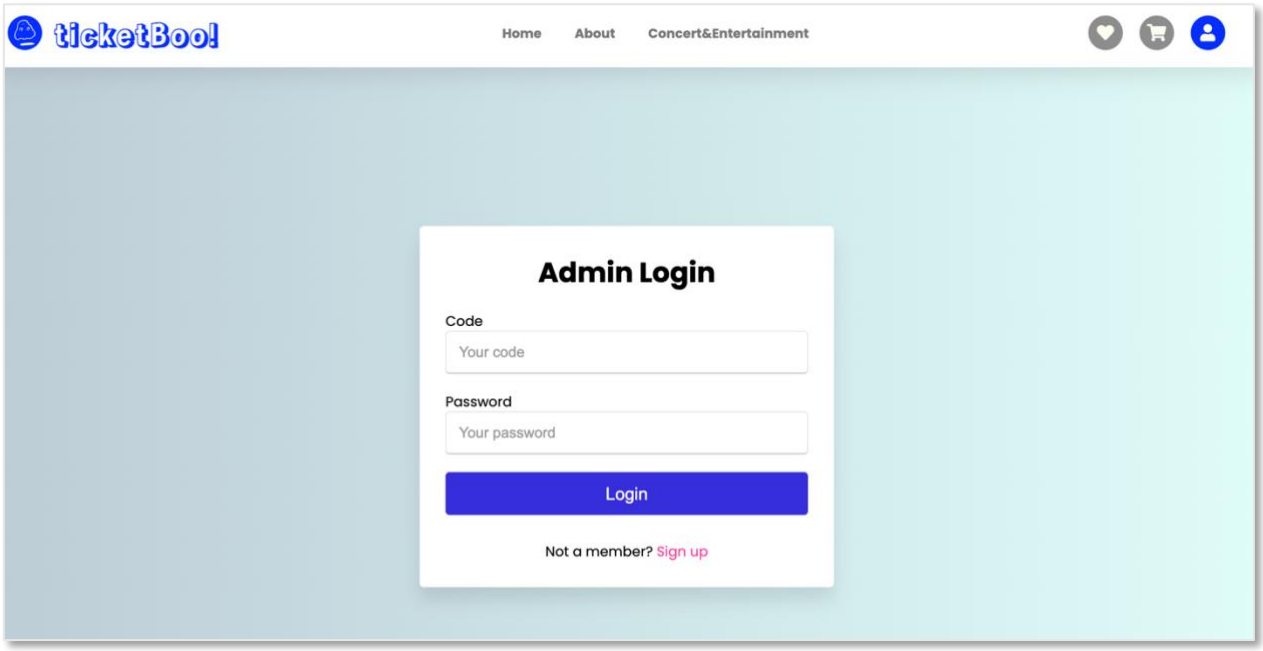
Explanation	**File: frontend.js** **[To connect the frontend.js and the backend.js together by fetch]**
This code sets up the basic structure	`const express = require('express');` `const path = require('path');` `const port = 3000` `const app = express();`

of an **Express application** and router that can be used to **handle incoming requests and return responses.**	`const router = express.Router();`
	`app.use(router)` `app.use(express.static(path.join(__dirname, 'public')));` `//For this line tells the application to serve static files from the 'public' directory` `//This part is for POST Method` `router.use(express.json());` `router.use(express.urlencoded({ extended: true }));` 
	`/* path to HomePage.html */` `router.get('/', (req, res) => {` `  console.log('Request at /');` `  res.sendFile(path.join(__dirname, '/public/html/HomePage.html'))` `})`



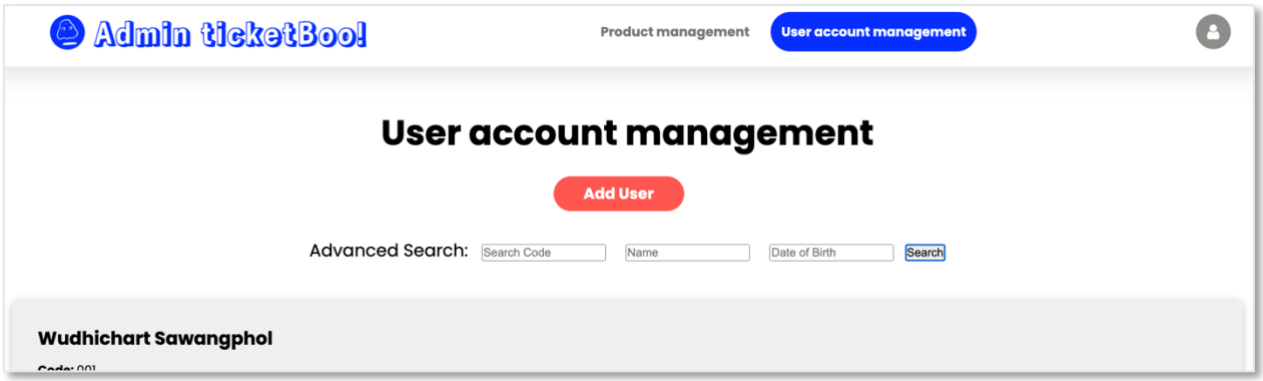
‘/public/html/HomePage.html’

```
/* path to Userlogin.html */
router.get('/login', (req, res) => {
    console.log('Request at /login');
    res.sendFile(path.join(__dirname, '/public/html/Userlogin.html'))
})
```



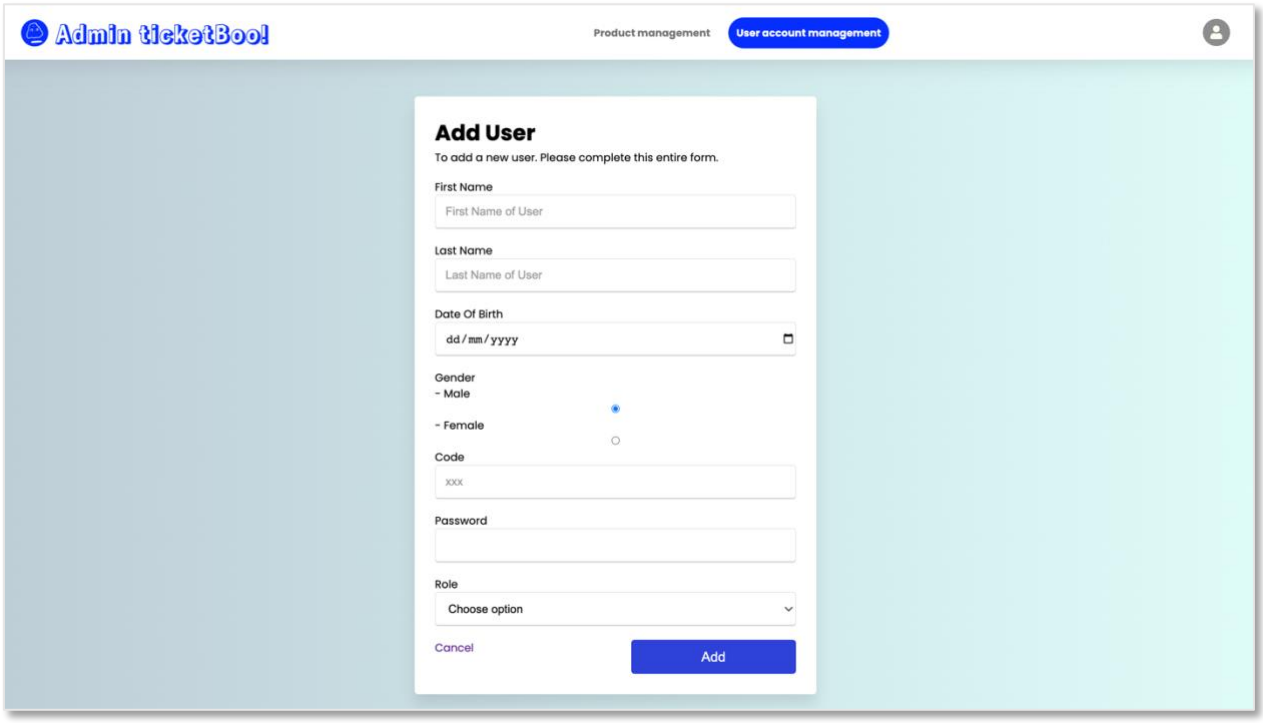
‘/public/html/Userlogin.html’

```
/* path to User-account-management.html */
router.get('/useracct', (req, res) => {
    console.log('Request at /userAcctManage');
    res.sendFile(path.join(__dirname, '/public/html/User-account-management.html'))
})
```



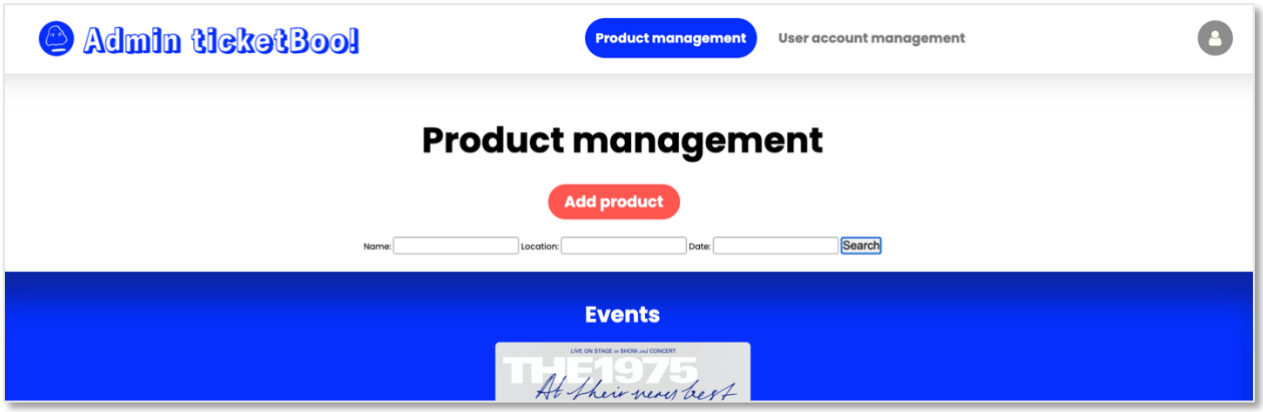
‘/public/html/User-account-management.html’

```
/* path to user.html */
router.get('/user', (req, res) => {
    console.log('Request at /user');
    res.sendFile(path.join(__dirname, '/public/html/user.html'))
})
```



‘/public/html/user.html’

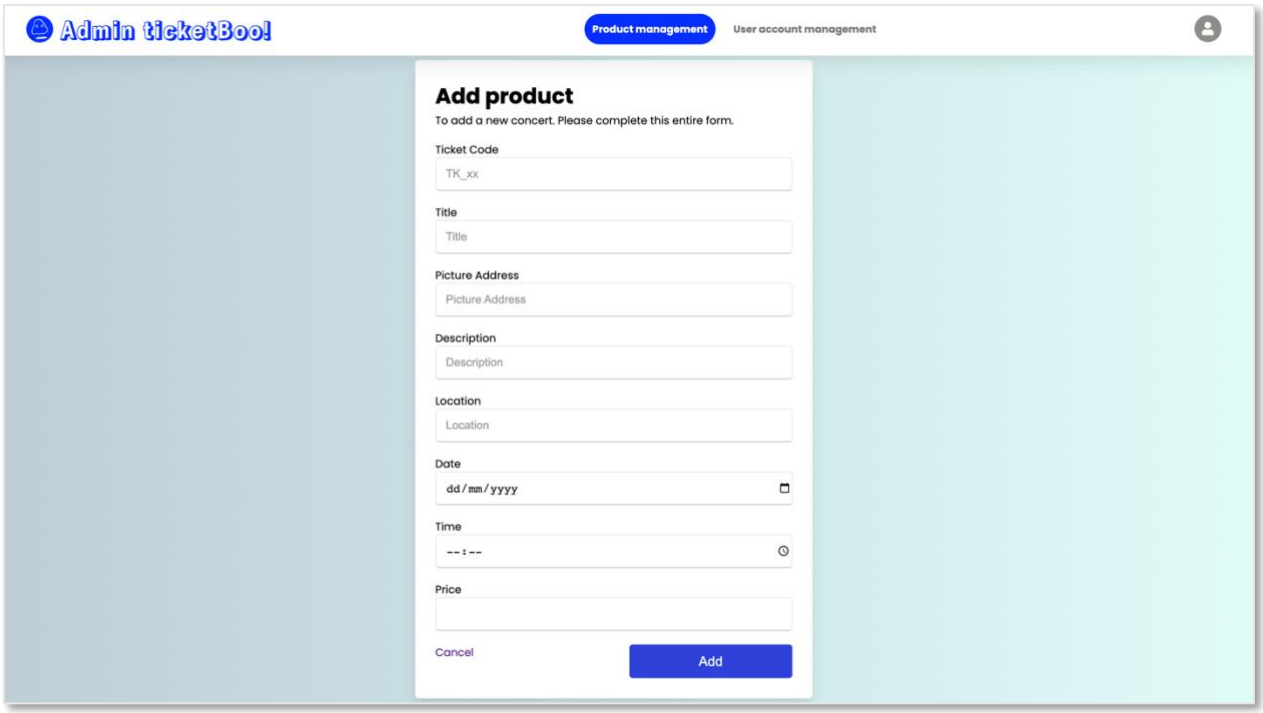
```
/* path to Product-management.html */
router.get('/manage', (req, res) => {
    console.log('Request at /manage');
    res.sendFile(path.join(__dirname, '/public/html/Product-management.html'))
})
```



‘/public/html/Product-management.html’

```
/* path to product.html */
```

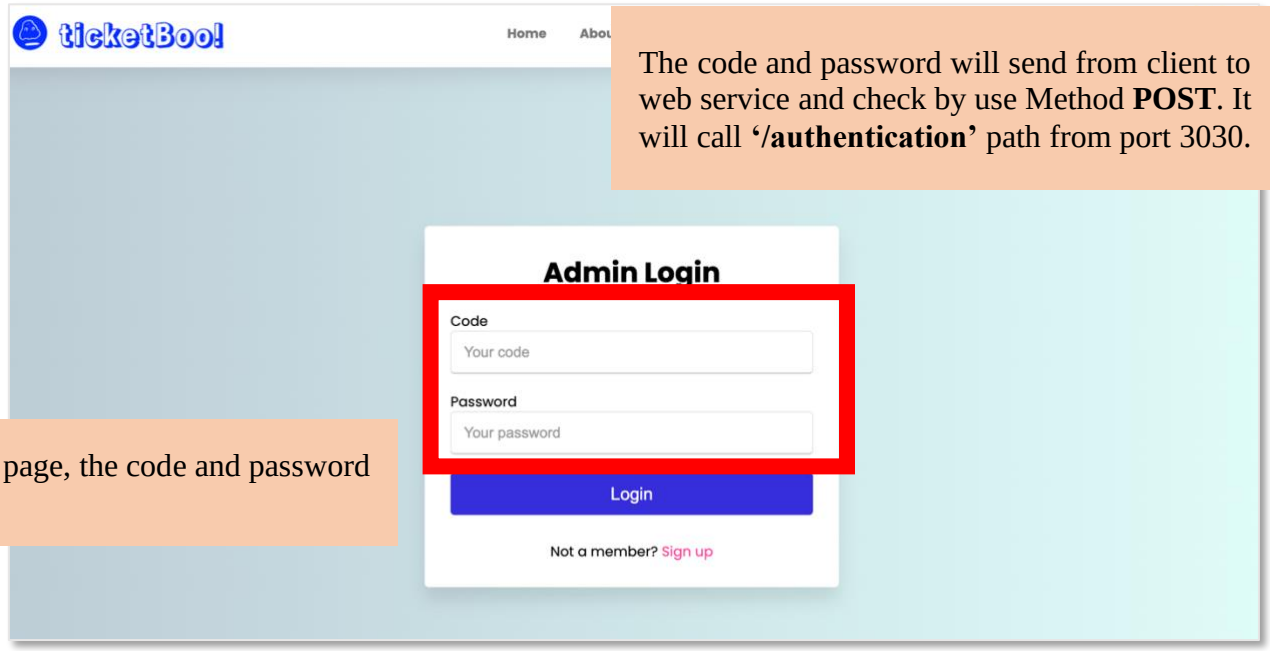
```
router.get('/product', (req, res) => {
    console.log('Request at /product');
    res.sendFile(path.join(__dirname, '/public/html/product.html'))
})
```



‘/public/html/product.html’

```
/* Form in Userlogin.html page will link to this
  If user submit the form in login page it will direct to this code.
  The value will send and in login in form will pass the code and password.*/
router.post('/login-submit', (req, res) => {
    console.log('Request at /login-submit');
    console.log("Login by " + req.body.code);//
    let code = req.body.code;
    let pass = req.body.password;

    /* we use fetch function with a method of POST to direct to
      server at port 3030 with path /authentication. And we will
      put the req.body of code and password in the JSON format*/
    fetch("http://localhost:3030/authentication", {
        method: 'POST',
        headers: {
```

```
            'Content-Type': 'application/json'
        },
        body: JSON.stringify({
            code: code,
            password: pass
        })
    })
        .then(response => response.json()) //after send the result, we wait for the response
        .then(data => {
            console.log(data)
            /* data that send back has 'exist' to indicate whether
               this account is exist or not. If it true it will go the
               manage page. It's mean if user can login, that mean that user
               can access to manage the product as they are admin. If it not
               it will go to path /login again to let user put the account again*/
            if (data.exist === "true") {
                res.redirect('/manage');
            } else {
                res.redirect('/login');
            }
        })
        .catch(error => console.error(error));
})
```

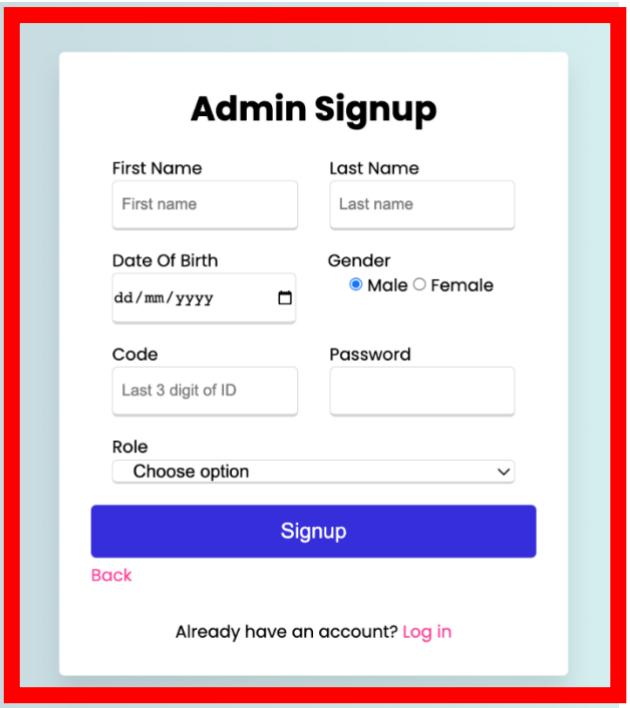


The code and password will send from client to web service and check by use Method **POST**. It will call **'/authentication'** path from port 3030.

To login in this page, the code and password must **MATCH**!

```
/* This will happen when user click submit to signup form which is
  in the SignuoAdmin.html*/
router.post('/signup-submit', (req, res) => {
    console.log('Request at /signin-submit');
    console.log("Form submitted by " + req.body.code);

    /* fetch to the server with port of 3030 with path newuser by method
      of 'POST'. We keep all data that user input in the form and send
      the data to this port */
    fetch("http://localhost:3030/newuser", {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify({
            fname: req.body.fname,
            lname: req.body.lname,
            dob: req.body.dob,
            gender: req.body.gender,
            code: req.body.code,
            password: req.body.password,
```

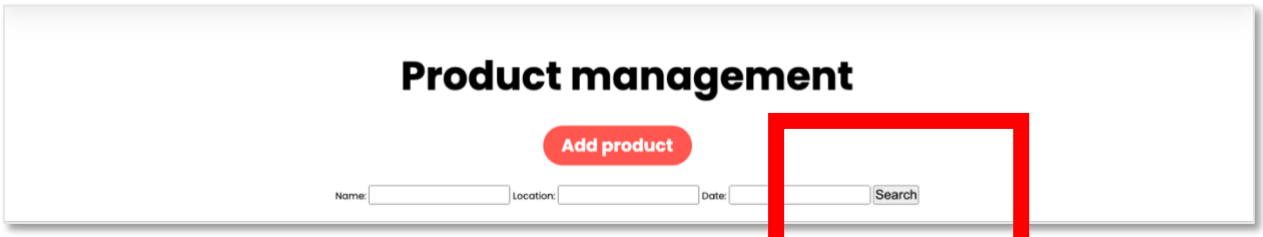
```
            role: req.body.role
        })
    })
        .then(response => response.json()) //after send the result, we wait for the response
        .then(data => {
            /* After signup, it will go to login page*/
            res.redirect('/login');
        })
        .catch(error => console.error(error));


})
```



If you don't have an account yet, you can fill it out and the information will be sent from the client to the web service of the web and collect data. Then you will return to the login page again to sign in. It will call **'/newuser' path** and **Method POST.**

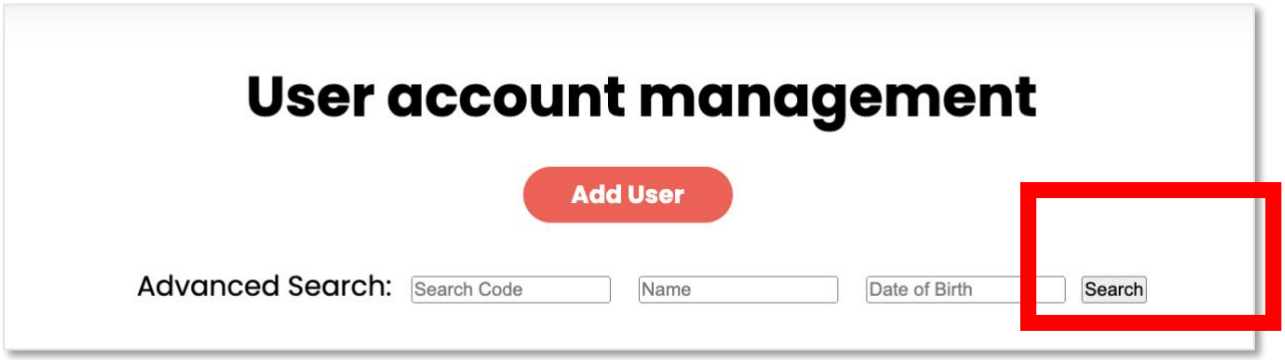
```
/* This use when user click the search button in page of Product-management.html
  it will collect the query that user want to find which can find by name, location, and date*/
router.post('/search-product-submit', (req, res) => {
    console.log("Submit")
```

```
    let name = req.body.name;
    let location = req.body.location;
    let date = req.body.date;
    console.log(name)

    /* we fetch to this server to send data in the format of JSON*/
    fetch('http://localhost:3030/searchticket', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify({
            name: name,
            location: location,
            date: date
        })
    })
        .then(response => response.json()) //after send the result, we wait for the response
        .then(data => {
            console.log(data);
            res.send(data); //send the result back to client
        })
        .catch(error => console.error(error));


});
```



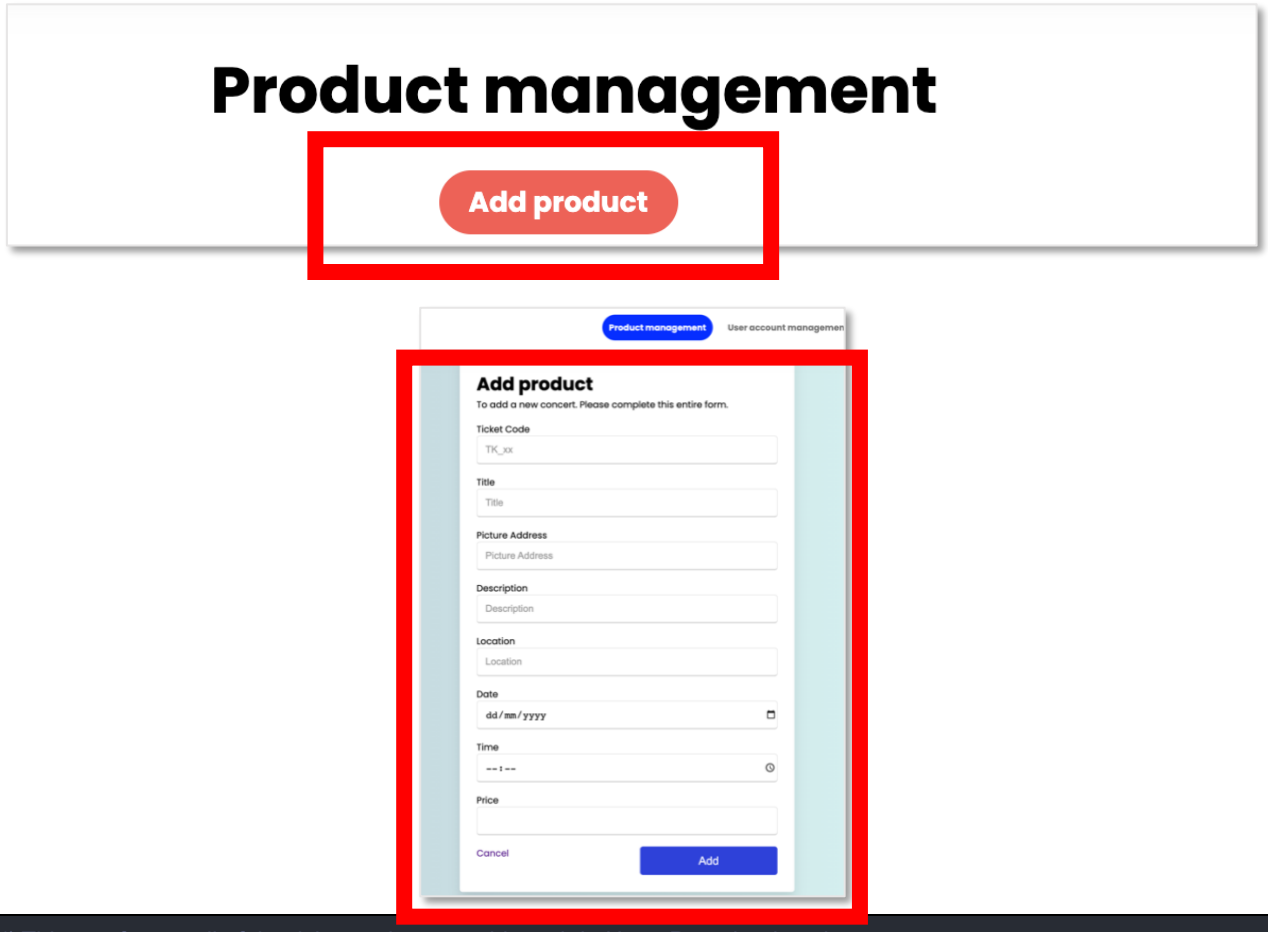

```
/* This will happen when user submit the search button of admin from the page of
  User-account-management.html by send the query key of code, name, and date. */
router.post('/search-user-submit', (req, res) => {
    console.log("Submit")
```

```
    let code = req.body.code;
    let name = req.body.name;
    let date = req.body.date;
    fetch('http://localhost:3030/searchadmin', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify({
            code: code,
            name: name,
            date: date
        })
    })
        .then(response => response.json()) //after send the result, we wait for the response
        .then(data => {
            console.log(data);
            res.send(data); //send the result back to client
        })
        .catch(error => console.error(error));
});
```



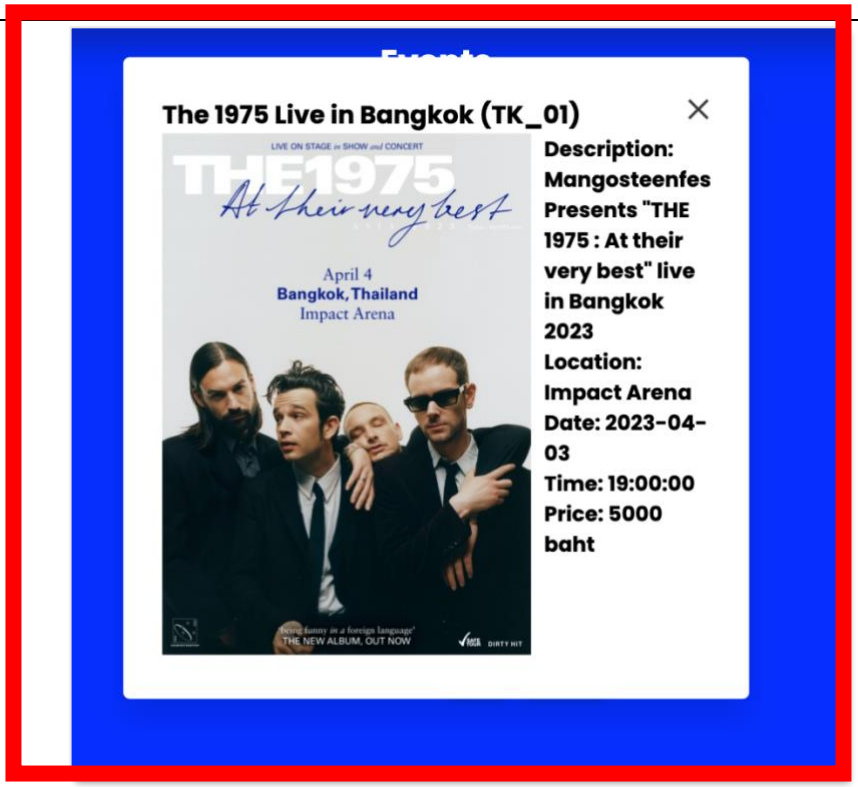

```
/* If user click add product which is the ticket in Product-management.html
  To add the ticket  user has to fill in all of these data. The data will send
  to this router*/
router.post('/addProduct', (req, res) => {
    let t_code = req.body.t_code;
    let t_title = req.body.t_title;
```

```
    let t_img = req.body.t_img;
    let t_desc = req.body.t_desc;
    let t_loca = req.body.t_loca;
    let t_date = req.body.t_date;
    let t_time = req.body.t_time;
    let t_price = req.body.t_price;
    console.log('Request at /addProduct');

    /* fetch data from this port and send the data as a JSON format */
    fetch('http://localhost:3030/insertticket', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify({
            t_code: t_code,
            t_title: t_title,
            t_img: t_img,
            t_desc: t_desc,
            t_loca: t_loca,
            t_date: t_date,
            t_time: t_time,
            t_price: t_price
        })
    })
        .then(response => response.json()) //after send the result, we wait for the response
        .then(data => {
            console.log(data);
            if (data.finish === "true") { //id finish add the ticket it will go the manage page
                return res.redirect('/manage');
            }
        })
        .catch(error => console.error(error));
});
```



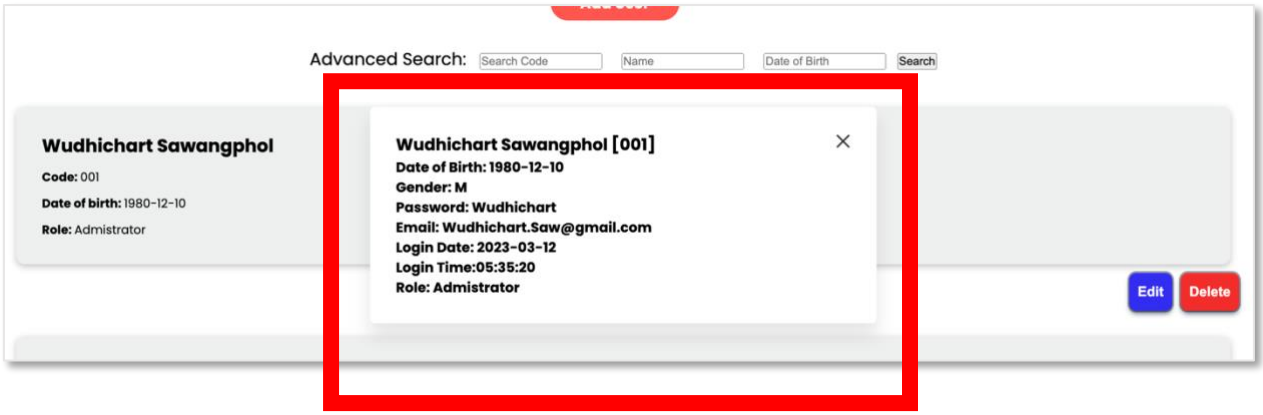

```
/* This use for get all of the ticket and we use this path in HomePage.html to show
  all of the ticket that we have.*/
router.get('/gettickets', (req, res) => {
    console.log('Request at /gettickets');

    /* We fetch the data from this port that will send all ticket by using GET method.
      And we don't send any query to this  */
    fetch('http://localhost:3030/tickets', {
        method: 'GET',
        headers: {
            'Content-Type': 'application/json'
        },
    })
        .then(response => response.json())
        .then(data => {
            console.log(data);
            res.send(data) /**send the data to the HomePage.html that fetch port3000 with
```

```
                    path of /gettickets */
        })
        .catch(error => console.error(error));
});
/* This will happen when user click the card to get only that information. This receive the ticketcode
  from the HomePage.html file*/
router.get('/getticket/:ticketCode', (req, res) => {
    const { ticketCode } = req.params;
    console.log(`Request at /getticket/${ticketCode}`);

    /* fetch the data with GET method by this ticketCode*/
    fetch(`http:///localhost:3030/ticket/${ticketCode}`, {
        method: 'GET',
        headers: {
            'Content-Type': 'application/json'
        },
    })
        .then(response => response.json())
        .then(data => {
            console.log(data);
            res.send(data);
        })
        .catch(error => console.error(error));
});
```

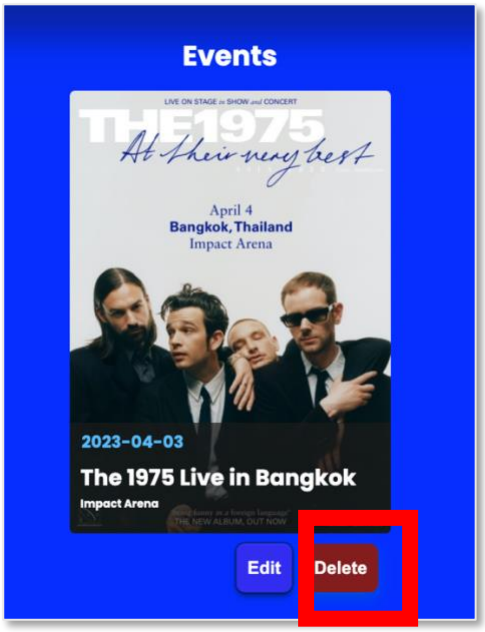



```
/* This use for get all of the admins we use this path in AboutUs.html to show
 all of the admin that we have.*/
router.get('/getadmins', (req, res) => {
    console.log('Request at /getadmins');

    /* We fetch the data from this port that will send all ticket by using GET method.
       And we don't send any query to this  */
    fetch('http://localhost:3030/admin_infos', {
        method: 'GET',
        headers: {
            'Content-Type': 'application/json'
        },
    })
        .then(response => response.json())
        .then(data => {
            console.log(data);
            res.send(data.data) /** send the data to the AboutUs.html that fetch port3000 with
                        path of /admin_infos */
```

```
        })
        .catch(error => console.error(error));
});
/** Same as ticket, when user click the card, this will get the
 * information from the port of 3030 and passing the adminCode to
 * find the information and send the response back to user.
 */
router.get('/admin_infos/:adminCode', (req, res) => {
    const { adminCode } = req.params;
    console.log(`Request at /admin_infos/${adminCode}`);
    fetch(`http://localhost:3030/admin_infos/${adminCode}`, {
        method: 'GET',
        headers: {
            'Content-Type': 'application/json'
        },
    })
        .then(response => response.json())
        .then(data => {
            console.log(data);
            res.send(data);
        })
        .catch(error => console.error(error));
});
```

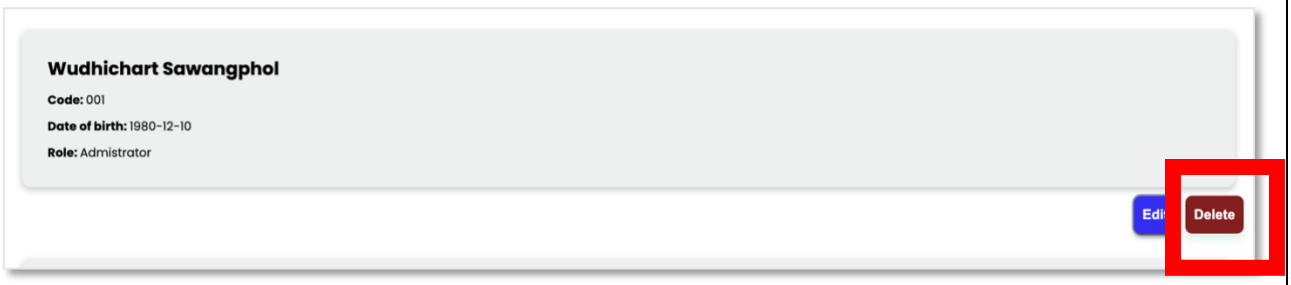


```
/** This will happen when user click delete product. It will delete by uesing
  * thie ticket code
*/
router.delete('/deleteProduct', (req, res) => {
    let t_code = req.body.t_code;
    console.log(req.body)
    console.log('Request at /deleteProduct');
    fetch('http://localhost:3030/deleteticket', {  // fetch this port with this path in the method of delete
        method: 'DELETE',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify({
            t_code: t_code,              // send this t_code to port of 3030
        })
    })
        .then(response => response.json())
        .then(data => {
            if (data.error === false) {  /* It has to make sure that there is no problem of DELETE
                                        and after that it will reload the page*/
                res.send(data);
                //return res.redirect('/manage');
            }
        })
        .catch(error => console.error(error));
});
```



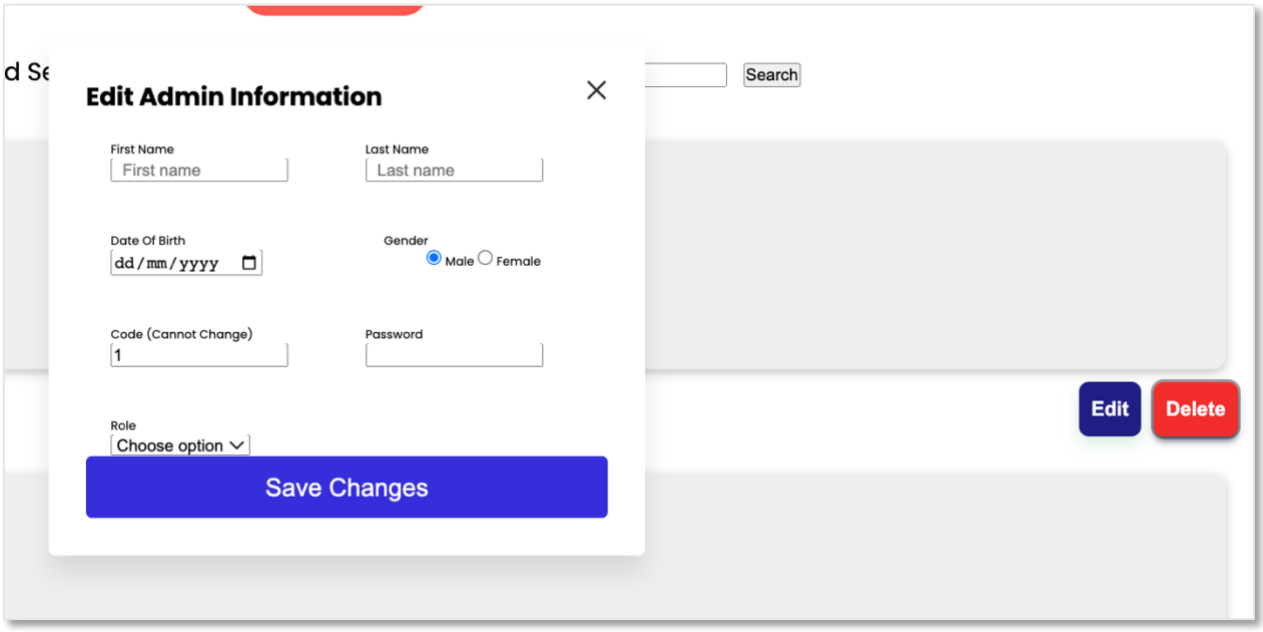

```
/** same as delete ticlet we use this to delete by useing admin_code that we get from html form. */
router.delete('/deleteAdmin', (req, res) => {
    let admin_code = req.body.admin_code;
    console.log('Request at /deleteAdmin');
    fetch('http://localhost:3030/admin_info', {
        method: 'DELETE',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify({
            admin_code: admin_code,
        })
    })
        .then(response => {
            console.log(response);
            return response.json();
        })
        .then(data => {
            console.log(data);
            if (data.error === false) {

                //res.redirect('/useracct');
                res.send(data.error);
```

```
        }
    })
    .catch(error => console.error(error));
});
```



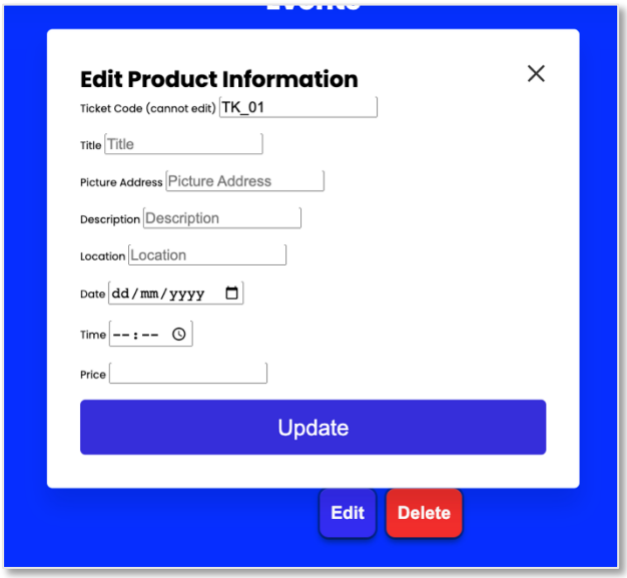

```
/** To edituser, It will receive data by using POST method and fetch to the server with PUT methid
 * to update the data.
 */
router.post('/edituser', (req, res) => {
  /**/
  fetch("http://localhost:3030/gettickets", {
    method: 'PUT',
    headers: {
      'Content-Type': 'application/json'
    },
    body: JSON.stringify({
      fname: req.body.fname,
      lname: req.body.lname,
      dob: req.body.dob,
      gender: req.body.gender,
      code: req.body.code,
      password: req.body.password,
      role: req.body.role
    })

  })
    .then(response => {
      console.log(response);
      return response.json();
```

```
    })
    .then(data => {
        console.log(data);
        if (data.error === false) {
            return res.redirect('/useracct');
        }
    })
    .catch(error => console.error(error));
});
```




```
/* Same as the ticket, we get the value */
router.post('/editproduct', (req, res) => {
    /* It will fetch with the method of 'PUT' to update the ticket*/
    fetch("http://localhost:3030/updateticket", {
        method: 'PUT',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify({          // we send all of these by convert to JSON format
            ticket_code: req.body.t_code,
            ticket_title: req.body.t_title,
            ticket_img: req.body.t_img,
```

```
            ticket_desc: req.body.t_desc,
            ticket_loca: req.body.t_loca,
            ticket_date: req.body.t_date,
            ticket_time: req.body.t_time,
            ticket_price: req.body.t_price,
        })

    })
        .then(response => {
            console.log(response);
            return response.json();
        })
        .then(data => {
            console.log(data);
            if (data.error === false) {  /** If it pass, it will go to path of manage */
                return res.redirect('/manage');
            }
        })
        .catch(error => console.error(error));
});
```




```
app.listen(port, function () {
    console.log("Server listening at Port " + port);
});
```

	[redacted]

Reference

https://youtu.be/MJUssi2C6Ls

https://youtu.be/t5zFfDdvApE
https://youtu.be/Ei_zX44ItEU
https://youtu.be/bW8X-tt5AZQ
https://youtu.be/oLgtucwjVII
https://youtube.com/watch?v=kRs3aTi3pzU&si=EnSIkaIECMiOmarE
https://www.youtube.com/watch?v=RctaFustg5w
https://youtu.be/H-DvSPRnKWQ
https://youtube.com/watch?v=kNO2WkKJzu0&si=EnSIkaIECMiOmarE
https://www.ticketmelon.com
https://youtu.be/V8PU_geaCCU