



Service-Oriented Computing

R . P . C . Thenuka

Table of Contents

Question 1	2
Question 2	7
Question 3	17
Create Database	17
Create restfull API by using CURD	19
Test API by suing POSTMAN TOOL	29
Question 4	41
Reference	43

Question 01

1. Read the following API and use HTML, CSS, Bootstrap, and PHP to display province names, latitudes, and longitudes in China.

URL: <https://rapidapi.com/axisbits-axisbits-default/api/covid-19-statistics>

💡 First, I will show you the source code I created for the above question. You can see it below:

```
<?php

$curl = curl_init();

curl_setopt_array($curl, [
    CURLOPT_URL => "https://covid-19-
statistics.p.rapidapi.com/provinces?iso=CHN",
    CURLOPT_RETURNTRANSFER => true,
    CURLOPT_ENCODING => "",
    CURLOPT_MAXREDIRS => 10,
    CURLOPT_TIMEOUT => 30,
    CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
    CURLOPT_CUSTOMREQUEST => "GET",
    CURLOPT_HTTPHEADER => [
        "X-RapidAPI-Host: covid-19-statistics.p.rapidapi.com",
        "X-RapidAPI-Key: 4034e2c397msh27430ace6ae51e4p17f74ejsn911be8a1ae04"
    ],
]);
]);

$response = curl_exec($curl);
$err = curl_error($curl);

curl_close($curl);

if ($err) {
    echo "cURL Error #:" . $err;
} else {
    $data = json_decode($response, true);
    $details = $data['data'];
    // var_dump($data);
}

?>
```

```
<!DOCTYPE html>
<html lang="en">
<head>

<meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Provinces of China</title>

    <!-- add Bootstrap -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOMLASjc" crossorigin="anonymous">

    <style>

        h1{
            color:rgb(234,25,100);
            font-weight: bold;
            position: fixed;
            margin-bottom: 100px;
            background-color: white;
            width: 100%;
            height: 100px;
        }

        td{
            font-weight: bold;
        }

        th{
            font-size: 1.5rem;
            position:sticky;
            top:60px;
        }

        table{
            margin-top: 50px;
        }

    </style>
</head>
<body>
```

```
<h1>Provinces names, latitudes, and longitudes in China...</h1>
<br>

<div class="container">

    <table border="1" width="100%" class="table table-striped">

        <tr class="table-dark">
            <th>province Names</th>
            <th>latitudes</th>
            <th>longitudes</th>
        </tr>

        <?php foreach($details as $detail){ ?>

            <tr>
                <td> <?php echo $detail['province']; ?> </td>
                <td> <?php echo $detail['lat']; ?> </td>
                <td> <?php echo $detail['long']; ?> </td>

            </tr>

            <?php
        }
        ?>

        </table>
    </div>
</body>
</html>
```

- Now I will show you the output of this source code. You can see it below,

Provinces names, latitudes, and longitudes in China...

province Names	latitudes	longitudes
Anhui	31.8257	117.2264
Beijing	40.1824	116.4142
Chongqing	30.0572	107.8740
Fujian	26.0789	117.9874
Gansu	36.0611	103.8343
Guangdong	23.3417	113.4244
Guangxi	23.8298	108.7881
Guizhou	26.8154	106.8748
Hainan	19.1959	109.7453
Hebei	38.0428	114.5149
Heilongjiang	47.8620	127.7615
Henan	33.8820	113.6140

Provinces names, latitudes, and longitudes in China...

province Names	latitudes	longitudes
Hong Kong	22.3000	114.2000
Hong Kong SAR	22.3	114.2
Hubei	30.9756	112.2707
Hunan	27.6104	111.7088
Inner Mongolia	44.0935	113.9448
Jiangsu	32.9711	119.4550
Jiangxi	27.6140	115.7221
Jilin	43.6661	126.1923
Liaoning	41.2956	122.6085
Macau	22.1667	113.5500
Macau SAR	22.1667	113.55
Ningxia	37.2692	106.1655
Qinghai	35.7452	95.9956

Provinces names, latitudes, and longitudes in China...

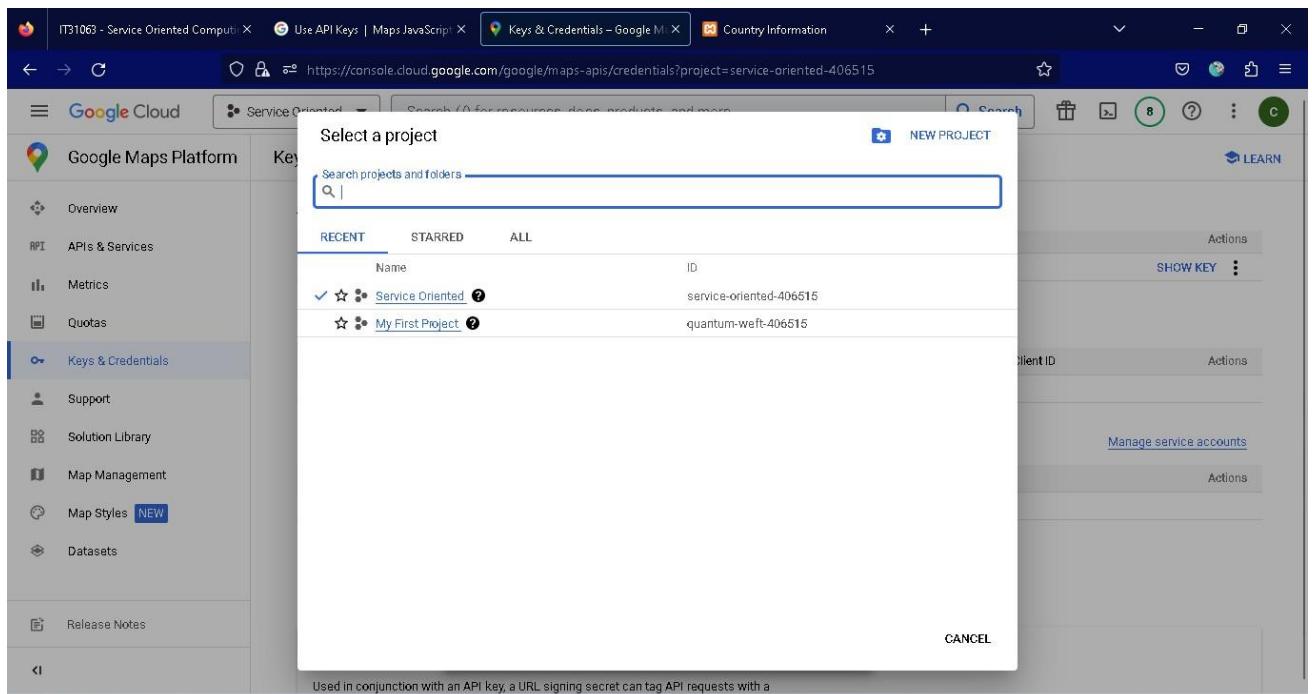
province Names	latitudes	longitudes
Qinghai	35.7452	95.9956
Shaanxi	35.1917	108.8701
Shandong	36.3427	118.1498
Shanghai	31.2020	121.4491
Shanxi	37.5777	112.2922
Sichuan	30.6171	102.7103
Tianjin	39.3054	117.3230
Tibet	31.6927	88.0924
Unknown		
Xinjiang	41.1129	85.2401
Yunnan	24.9740	101.4870
Zhejiang	29.1832	120.0934

Question 2

2. Use the REST Countries API documentation <https://restcountries.com/> to get all countries in a dropdown list, then select one and display the following information: flag, country official name, capital city, region, subregion, currencies, country code, population, area, borders, and Google Map. (Use HTML, CSS, Bootstrap, PHP, and AJAX.)

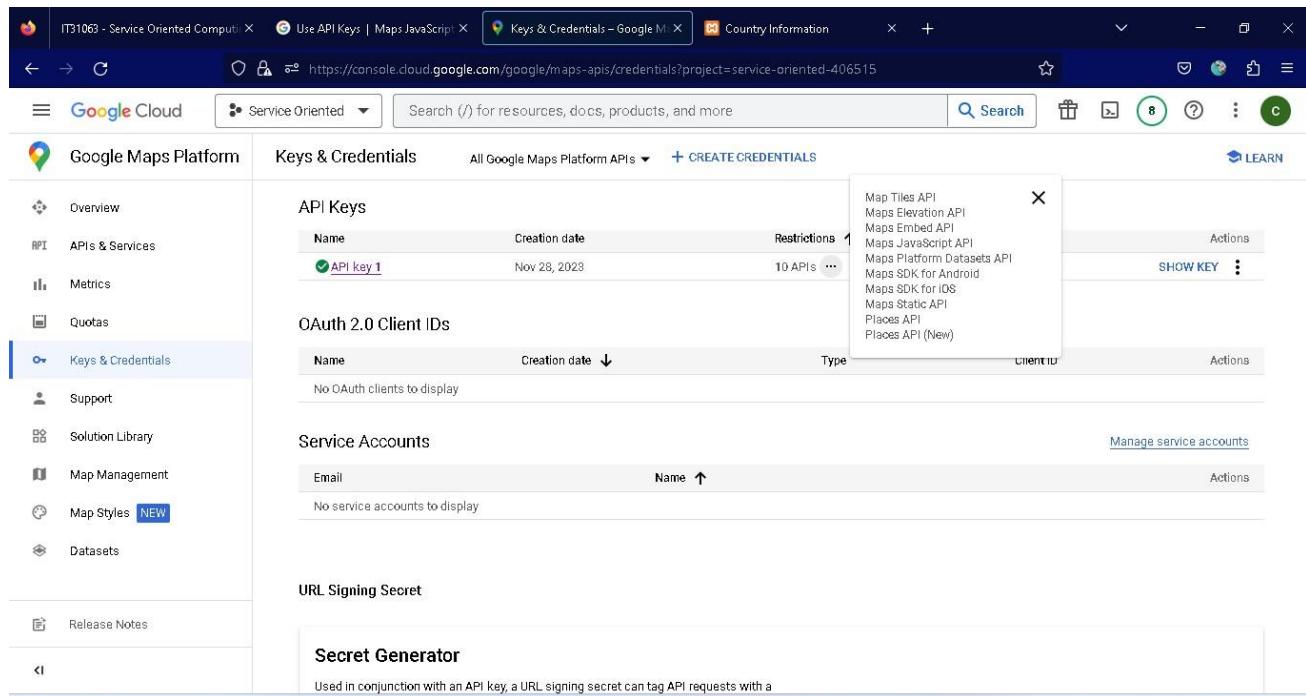
⊕ First, I created my own Google Maps API key using Google Cloud. I started by creating a new project called 'Service Oriented'. Then, I followed the necessary steps to generate my API key. I restricted the key to 10 API services, including the Maps JavaScript API. You can see my project name, API restrictions, and Google Maps API key below:

⊕ Create a new project!



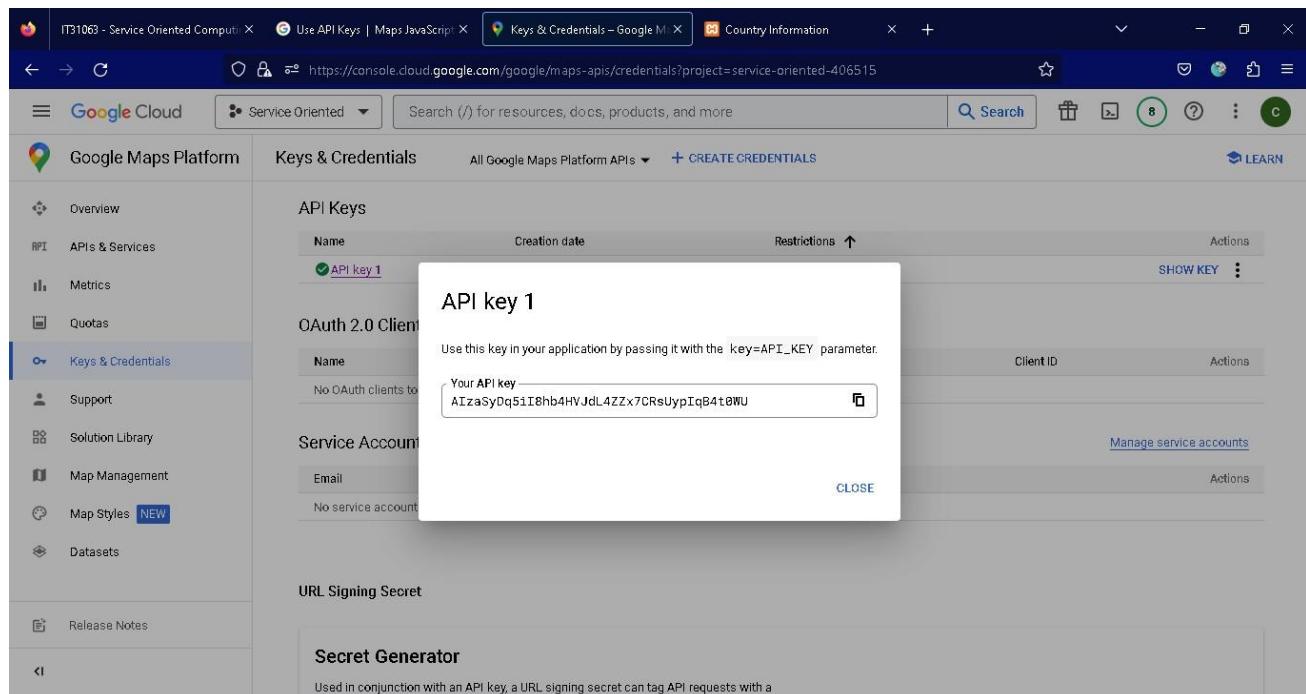
The screenshot shows a browser window with the URL <https://console.cloud.google.com/google/maps-apis/credentials?project=service-oriented-406515>. The left sidebar is for the Google Maps Platform, with 'Keys & Credentials' selected. A modal dialog titled 'Select a project' is open, showing a list of recent projects. The 'Service Oriented' project is selected, and its ID is listed as 'service-oriented-406515'. Other visible projects include 'My First Project' with ID 'quantum-weft-406515'. The dialog has 'SHOW KEY' and 'Actions' buttons at the bottom.

Added restrictions.



The screenshot shows the Google Cloud Platform API Keys page. On the left, there's a sidebar with various services like Overview, APIs & Services, Metrics, Quotas, Keys & Credentials (which is selected), Support, Solution Library, Map Management, Map Styles (NEW), and Datasets. The main area is titled "Keys & Credentials" and shows "All Google Maps Platform APIs". A table lists an "API Key" named "API key 1" created on Nov 28, 2023, with "10 APIs" under "Restrictions". A dropdown menu next to the restrictions shows a list of Google APIs: Map Tiles API, Maps Elevation API, Maps Embed API, Maps JavaScript API, Maps Platform Datasets API, Maps SDK for Android, Maps SDK for iOS, Maps Static API, Places API, and Places API (New). Below this is a section for "OAuth 2.0 Client IDs" which is currently empty. There's also a section for "Service Accounts" which is also empty. At the bottom, there's a "Secret Generator" section with a note about URL signing secrets.

Google Map API key



This screenshot is similar to the previous one but focuses on the details of the "API key 1" row. A modal window is open over the table, titled "API key 1". It contains the instruction "Use this key in your application by passing it with the `key=API_KEY` parameter." Below this is a text input field labeled "Your API key" containing the value "AIzaSyDq5iI8hb4HVJdL4ZZx7CRsUypIqB4t0WU". At the bottom right of the modal is a "CLOSE" button.

- Next, I created a file named question2.php to populate a dropdown menu with all country names. You can see the source code below

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Country Information</title>
    <link
        href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
        rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpucO0mLASjC"
        crossorigin="anonymous">

    <style>

        .container h1{
            color:rgb(234,25,100);
            font-weight: bold;
        }

        label{

            font-weight: bold;
            font-size: 1.5rem;
        }

        option{
            font-weight: bold;
        }

        option :hover{
            color: red;
        }

    </style>
```

```
select{
    width: 100%;
    color: black;
    background-color: white;
    font-weight: bold;
    border: 1px solid black;
    border-radius: 5px;
    height: 40px;
}

select:hover{
    color: red;
}

table tr{
    border: 2px solid gray;
}

table tr td{
    font-weight: bold;
}

table tr td h5{
    font-weight: bold;
}

hr{
    border: 2px solid gray;
}


```

</style>

</head>

<body>

<div class="container p-5 my-5 border">

<h1>Country Informations</h1>


```

<form>
    <div>
        <label for="countrySelect">Select a country :</label>
        <select id="countrySelect">

            <?php
                $url = "https://restcountries.com/v3.1/all";
                $data = file_get_contents($url);
                $data = json_decode($data, true);

                if(is_array($data)){
                    foreach($data as $country){
                        echo '<option value="' . $country['cca3'] . '">' .
                            $country['name']['common'] . '</option>';
                    }
                }
            ?>
        </select>
    </div>
</form>

<hr>

<div id="countryInformation"></div>

</div>

<script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>
<script>
    $('#countrySelect').change(function(){
        var SelectedcountryCode = $(this).val();
        $.ajax({
            url:'getcountry.php',
            type:'POST',
            data: {countryCode:SelectedcountryCode },
            success: function(response){
                $('#countryInformation').html(response);
            }

        });
    });
</script>

</body>
</html>

```

- After completing “question2.php” file I created “getcountry.php” file to display selected country Informations, in below you can see source code of the “getcountry.php” file.

```

<?php
if(isset($_POST[ 'countryCode'])){

    $SelectedcountryCode = $_POST[ 'countryCode'];

    $url="https://restcountries.com/v3.1/alpha/$SelectedcountryCode";
    $data = file_get_contents($url);
    $data = json_decode($data , true);

    if(is_array($data) && !empty($data)){
        $country = $data[0];

        echo '<table class="table table-bordered table-striped">

            <tr><td><h5>Flag</h5><td></td></td></tr>

            <tr><td><h5>Official
Name</h5><td>' .($country['name']['official'])?? 'Data Not
Available').'</td></td></tr>

            <tr><td><h5>Capital City</h5><td>' .($country['capital'][0])?? 'Data Not
Available').'</td></td></tr>

            <tr><td><h5>Region</h5><td>' .($country['region'])?? 'Data Not
Available').'</td></td></tr>

            <tr><td><h5>Subregion</h5><td>' .($country['subregion'] ?? 'Data
Not Available').'</td></td></tr>

            <tr><td><h5>Currency</h5><td>' .implode(', ' ,
array_column($country[ 'currencies'], 'name')).'</td></td></tr>

            <tr><td><h5>Country Code</h5><td>' .($country['tld'][0])?? 'Data
Not Available').'</td></td></tr>

            <tr><td><h5>Population</h5><td>' .(number_format($country[ 'populat
ion']))?? 'Data Not Available').'</td></td></tr>

            <tr><td><h5>Area</h5><td>' .(number_format($country[ 'area']))?? 'Data
Not Available').'</td></td></tr>

```

```
<tr><td><h5>Borders</h5><td>'.(isset($country['borders']) ?
implode(", " , $country['borders']) : 'data not availabe').'
</td></td></tr>

<tr><td><h5>Google Map</h5><td>

<iframe width="100%" height="200"
frameborder="1" style="border:1"
referrerpolicy="no-referrer-when-downgrade"
src="https://www.google.com/maps/embed/v1/place?key=AIzaSyDq5iI8hb4HVJdL4ZZx7CRsUyptqB4t0WU&q='
.($country['name'][ 'common'])?? 'Data Not Available').&zoom=7"
allowfullscreen>
</iframe>

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

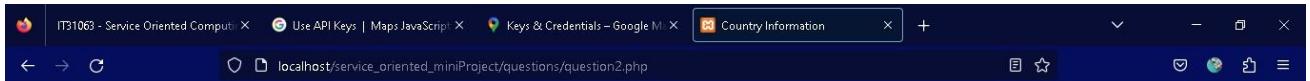
</table>';
}

else{
    echo 'not found';
}
}else{
    echo 'invalid';
}

?>
```

⊕ Now, I will show you screenshots of the outputs:

⊕ **My drop-down menu**



Country Informations

Select a country :

Sri Lanka

⊕ All country names within the dropdown menu (note: this screenshot shows only a few country names, as there are too many to capture in one screenshot):



Country Informations

Select a country :

Sri Lanka

Christmas Island
Eritrea
Samoa
North Macedonia
Djibouti
Jordan
Pakistan
French Polynesia
Ireland
Mauritania
Denmark
Namibia

 When I select the country name ‘Japan’ from the dropdown menu



Country Informations

Select a country :

Japan	
Flag	
Official Name	Japan
Capital City	Tokyo



Region	Asia
Subregion	Eastern Asia
Currency	Japanese yen
Country Code	.jp
Population	125,836,021
Area	377,930
Borders	data not available
Google Map	

 When I select another country ,

IT31063 - Service Oriented Comput X Use API Keys | Maps JavaScript X Keys & Credentials – Google M X Country Information +

localhost/service_oriented_miniProject/questions/question2.php

Country Informations

Select a country :

Poland

Flag	
Official Name	Republic of Poland
Capital City	Warsaw
Region	Europe

IT31063 - Service Oriented Comput X Use API Keys | Maps JavaScript X Keys & Credentials – Google M X Country Information +

localhost/service_oriented_miniProject/questions/question2.php

Subregion	Central Europe
Currency	Polish złoty
Country Code	.pl
Population	37,950,802
Area	312,679
Borders	BLR, CZE, DEU, LTU, RUS, SVK, UKR
Google Map	

Question 3

3. Develop RestFul API using PHP - Object Oriented Programming(OOP) and PDO – MySQL

a. Create a database and database table.

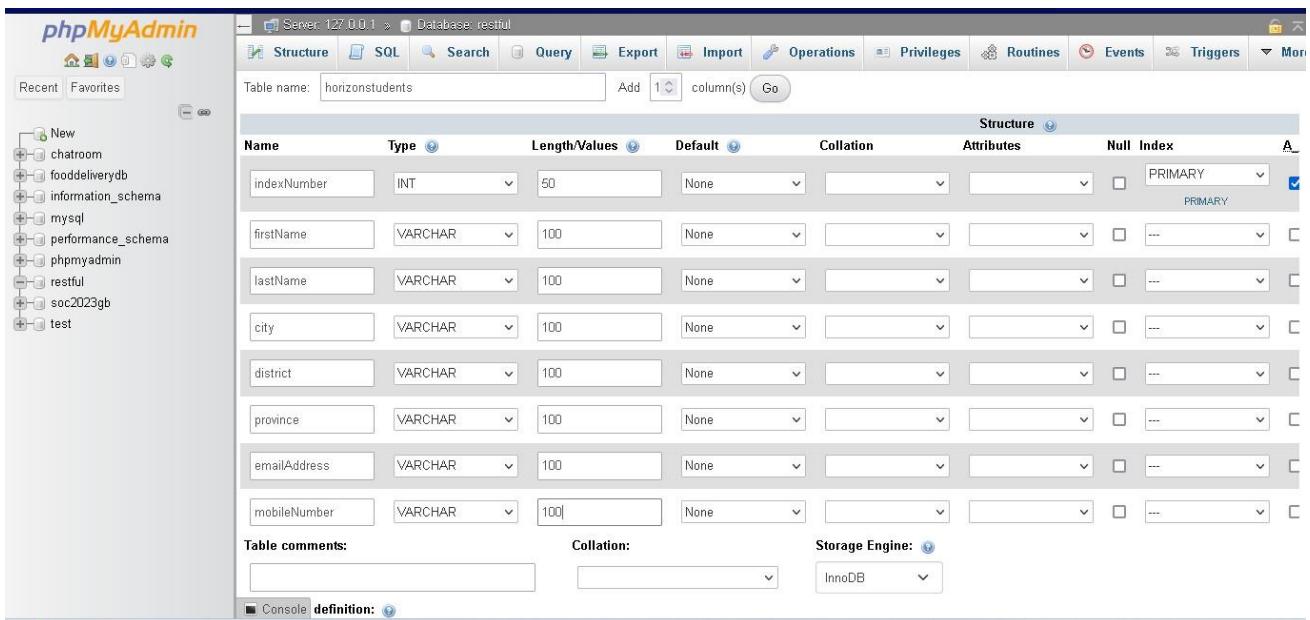
- >Create a database called "restful."

The screenshot shows the phpMyAdmin interface for MySQL. In the top navigation bar, the 'Databases' tab is selected. On the left sidebar, there is a tree view of databases: New, chatroom, fooddeliverydb, information_schema, mysql, performance_schema, phpmyadmin, soc2023gb, and test. A modal window titled 'Create database' is open in the center. It contains a text input field with 'restful' typed into it, a dropdown menu set to 'utf8mb4_general_ci', and a 'Create' button. Below the modal, a table lists existing databases with their collations and actions like 'Check privileges'. The table shows 8 total databases. At the bottom of the modal, there are 'Check all' and 'Drop' buttons, and a search bar.

- Create a table named “horizonstudents” inside the database.

The screenshot shows the phpMyAdmin interface for the 'restful' database. In the top navigation bar, the 'Structure' tab is selected. The left sidebar shows the same list of databases as the previous screenshot. A modal window titled 'Create new table' is open. It has fields for 'Table name' (set to 'horizonstudents') and 'Number of columns' (set to 8), with a 'Create' button. The main area of the screen displays the message 'No tables found in database.'

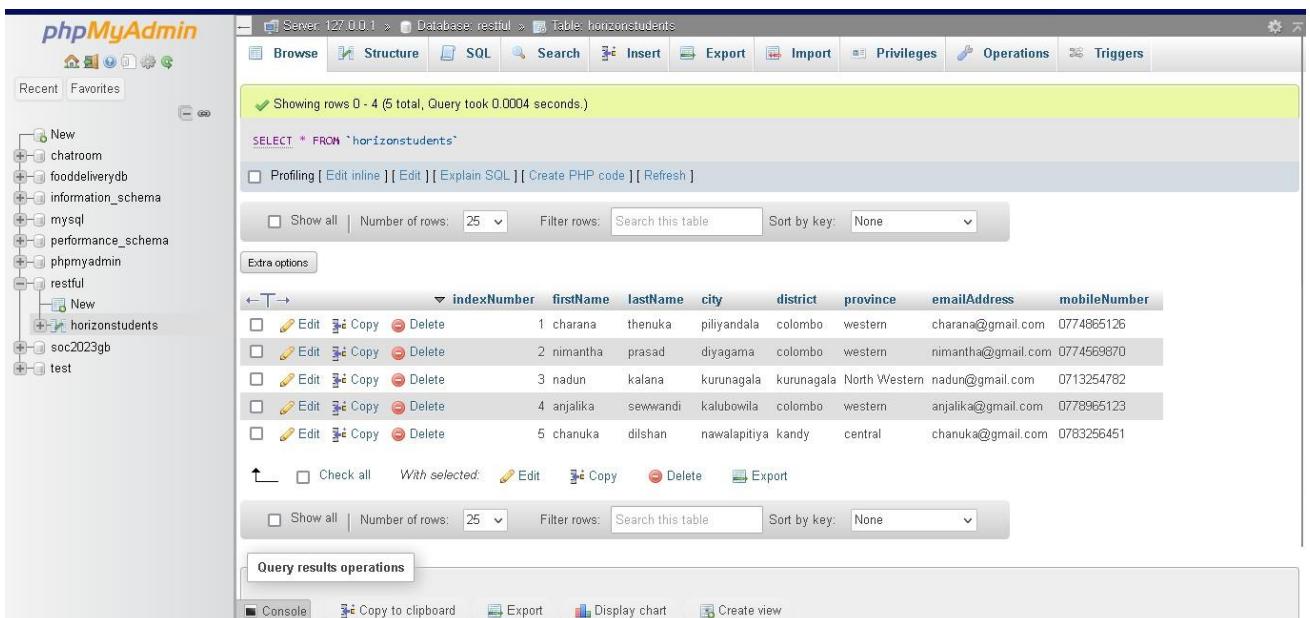
-  Add eight (8) fields named “**Index No., First Name, Last Name, City, District, Province, Email Address, and Mobile Number.**”



The screenshot shows the 'Structure' tab of the phpMyAdmin interface for creating a new table named 'horizonstudents'. The table has the following columns:

Name	Type	Length/Values	Default	Collation	Attributes	Null	Index
indexNumber	INT	50	None			✓	PRIMARY
firstName	VARCHAR	100	None			✓	...
lastName	VARCHAR	100	None			✓	...
city	VARCHAR	100	None			✓	...
district	VARCHAR	100	None			✓	...
province	VARCHAR	100	None			✓	...
emailAddress	VARCHAR	100	None			✓	...
mobileNumber	VARCHAR	100	None			✓	...

-  Insert 5 rows into the table.



The screenshot shows the 'horizonstudents' table in the phpMyAdmin 'Browse' tab. The table contains the following data:

	indexNumber	firstName	lastName	city	district	province	emailAddress	mobileNumber
1	charana	thenuka	piliyandala	colombo	western	charana@gmail.com	0774865126	
2	nimantha	prasad	diyagama	colombo	western	nimantha@gmail.com	0774569870	
3	nadun	kalana	kurunagala	kurunagala	North Western	nadun@gmail.com	0713254782	
4	arjalika	sewwandi	kalubowila	colombo	western	arjalika@gmail.com	0778965123	
5	chanuka	dilshan	nawalapitiya	kandy	central	chanuka@gmail.com	0783256451	

b. Create a RestFul web service to perform CURD operations with MySQL database.

- Write RestFul API for the popular HTTP verbs **GET, POST, PUT, and DELETE**

⊕ First, I created a folder inside the htdocs directory called restfull. Inside this folder, I created another folder called include. Within this folder, I created a connection.php file to establish the database connection. You can see the source code below.

```
<?php

$host = "localhost";
$username = "root";
$password = "";
$dbname = "restful";

$connection = mysqli_connect($host , $username , $password , $dbname);

if(!$connection){
    die("connection Failed " . mysqli_connect_error());
}

?>
```

- Then I created “**horizonstudents.php**” file inside the ”**restfull**” folder. I will show to you source code of the “**horizonstudents.php**” file in below,

```
<?php

error_reporting(0); // to use hide the unnesserory errors

header('Access-Control-Allow-Origin: *');
header('Content-Type: application/json');

include('function.php');

$requestMethod = $_SERVER["REQUEST_METHOD"];

switch ($requestMethod) {
    case 'GET':
        $studentList = getStudentlist();
        echo $studentList;
        break;

    case 'POST':
        $inputData = file_get_contents("php://input");
        $inputData = json_decode($inputData, true);
        $insertStudent = insertStudent($inputData);
        echo $insertStudent;
        break;

    case 'PUT':
        $inputData = file_get_contents("php://input");
        $inputData = json_decode($inputData, true);
        $updateStudent = updateStudent($inputData, $_GET);
        echo $updateStudent;
        break;

    case 'DELETE':
        $deleteStudent = deleteStudent($_GET);
        echo $deleteStudent;
        break;
}
```

```

default:
    $data = [
        'status' => 405,
        'message' => $requestMethod . " Method Not Allowed",
    ];

    header("HTTP/1.0 405 Method Not Allowed");
    echo json_encode($data);
    break;
}
?>

```

- After that, I created a .htaccess file to remove the .php extension from the horizonstudents.php file, as the API URL should not include the extension. I will show you the source code of this .htaccess file below

```

<IfModule mod_rewrite.c>
    RewriteEngine On
    RewriteCond %{REQUEST_FILENAME} !-d
    RewriteCond %{REQUEST_FILENAME} !-f
    RewriteRule ^([^\.]+)$ $1.php [NC,L]
</IfModule>

```

- Finally, I created the function.php file. I used functions for each HTTP verb (GET, POST, PUT, DELETE). This file includes all database queries, response messages (error messages), and validation parts for when users request services. I will show you the function.php file below.

```
<?php

require './include/connection.php';

function getStudentlist(){

    global $connection;

    $query = "SELECT * FROM horizonstudents";
    $query_run = mysqli_query($connection , $query);

    if($query_run){

        if(mysqli_num_rows($query_run) > 0){

            $res = mysqli_fetch_all($query_run , MYSQLI_ASSOC);

            $data = [
                'status' =>200,
                'message' => "Successfully fetch students" ,
                'data' => $res
            ];
            header("HTTP/1.0 200 OK ");
            return json_encode($data);
        }
    }else{
        $data = [
            'status' =>404,
            'message' => "No student Found" ,
        ];
        header("HTTP/1.0 404 No student Found");
        return json_encode($data);
    }
}
```

```

}else{

    $data = [
        'status' =>500,
        'message' => "Internal Server Error" ,

    ];
    header("HTTP/1.0 500 Internal Server Error");
    return json_encode($data);
}

function error422($message){

    $data = [
        'status' =>422,
        'message' => $message,

    ];
    header("HTTP/1.0 422 Unprocessable entity");
    echo json_encode($data);
    exit();
}

function insertStudent($studentInput){

    global $connection;

    $firstName = mysqli_real_escape_string($connection ,
$studentInput['firstName']);
    $lastName = mysqli_real_escape_string($connection ,
$studentInput['lastName']);
    $city = mysqli_real_escape_string($connection ,
$studentInput['city']);
    $district = mysqli_real_escape_string($connection ,
$studentInput['district']);
    $province = mysqli_real_escape_string($connection ,
$studentInput['province']);
    $emailAddress = mysqli_real_escape_string($connection ,
$studentInput['emailAddress']);
    $mobileNumber = mysqli_real_escape_string($connection ,
$studentInput['mobileNumber']);
}

```

```

if(empty(trim($firstName))){
    return error422('enter your first Name');
}elseif(empty(trim($lastName))){
    return error422('enter your last name');

}elseif(empty(trim($city))){
    return error422('enter your city');

}elseif(empty(trim($district))){
    return error422('enter your district');

}elseif(empty(trim($province))){
    return error422('enter your province');

}elseif(empty(trim($emailAddress))){
    return error422('enter your email address');

}elseif(empty(trim($mobileNumber))){
    return error422('enter your mobile number');

}else{
    $query = "INSERT INTO horizonstudents (firstName , lastName ,
city , district , province , emailAddress , mobileNumber)
VALUES ('$firstName' , '$lastName' , '$city' , '$district' ,
'$province' , '$emailAddress' , '$mobileNumber')";

    $result = mysqli_query($connection , $query);

    if($result){
        $data = [
            'status' =>201,
            'message' => "student Insert Sucessfully" ,

        ];
        header("HTTP/1.0 201 Created");
        return json_encode($data);
    }
}

```

```

}else{
    $data = [
        'status' =>500,
        'message' => "Internal Server Error" ,
    ];
    header("HTTP/1.0 500 Internal Server Error");
    return json_encode($data);
}

}

function updateStudent($studentInput , $studentParams){
    global $connection;

    if(!isset($studentParams['indexNumber'])){
        return error422("student Index Number Not found in URL");
    }elseif($studentParams['indexNumber'] == null){
        return error422("Enter the Student Index Number");
    }

    $IndexNumber = mysqli_real_escape_string($connection ,
$studentParams['indexNumber']);

    $firstName = mysqli_real_escape_string($connection ,
$studentInput['firstName']);
    $lastName = mysqli_real_escape_string($connection ,
$studentInput['lastName']);
    $city = mysqli_real_escape_string($connection ,
$studentInput['city']);
    $district = mysqli_real_escape_string($connection ,
$studentInput['district']);
    $province = mysqli_real_escape_string($connection ,
$studentInput['province']);
    $emailAddress = mysqli_real_escape_string($connection ,
$studentInput['emailAddress']);
    $mobileNumber = mysqli_real_escape_string($connection ,
$studentInput['mobileNumber']);
}

```

```

if(empty(trim($firstName))){
    return error422('enter your first Name');
}elseif(empty(trim($lastName))){
    return error422('enter your last name');

}elseif(empty(trim($city))){
    return error422('enter your city');

}elseif(empty(trim($district))){
    return error422('enter your district');

}elseif(empty(trim($province))){
    return error422('enter your province');

}elseif(empty(trim($emailAddress))){
    return error422('enter your email address');

}elseif(empty(trim($mobileNumber))){
    return error422('enter your mobile number');

}else{
    $query = "UPDATE horizonstudents SET firstName='$firstName' ,
lastName='$lastName' , city='$city' ,
district='$district' , province='$province' ,
emailAddress='$emailAddress' , mobileNumber='$mobileNumber'
WHERE indexNumber='$IndexNumber' LIMIT 1" ;

$result = mysqli_query($connection , $query);

if($result){
    $data = [
        'status' =>200,
        'message' => "student Updated Sucessfully" ,
    ];
    header("HTTP/1.0 200 success");
    return json_encode($data);
}

```

```

}else{
    $data = [
        'status' =>500,
        'message' => "Internal Server Error" ,
    ];
    header("HTTP/1.0 500 Internal Server Error");
    return json_encode($data);
}

}

function deleteStudent($studentParams){
    global $connection;

    if(!isset($studentParams['indexNumber'])){
        return error422("student Index Number Not found in URL");
    }elseif($studentParams['indexNumber'] == null){
        return error422("Enter the Student Index Number");
    }

    $IndexNumber = mysqli_real_escape_string($connection ,
$studentParams['indexNumber']);

    $query = "DELETE FROM horizonstudents WHERE
indexNumber='$IndexNumber' LIMIT 1" ;
    $result = mysqli_query($connection , $query);

    if($result){

        $data = [
            'status' =>200,
            'message' => "student delete successfully" ,
        ];
        header("HTTP/1.0 200 OK");
        return json_encode($data);
    }
}

```

```

}else{

    $data = [
        'status' =>404,
        'message' => "student not found" ,

    ];
    header("HTTP/1.0 404 not found");
    return json_encode($data);
}

?>

```

💡 Now I show my API without the file extension(.php). You can see below,

```

{
  "status": 200,
  "message": "Successfully fetch students",
  "data": [
    {
      "indexNumber": "1",
      "firstName": "charana",
      "lastName": "thenuka",
      "city": "piliyandala",
      "district": "colombo",
      "province": "western",
      "emailAddress": "charana@gmail.com",
      "mobileNumber": "0774865126"
    },
    {
      "indexNumber": "2",
      "firstName": "nimantha",
      "lastName": "prasad",
      "city": "diyagama",
      "district": "colombo",
      "province": "western",
      "emailAddress": "nimantha@gmail.com",
      "mobileNumber": "0774569870"
    },
    {
      "indexNumber": "3",
      "firstName": "nadun",
      "lastName": "kalana",
      "city": "kurunapala",
      "district": "kurunapala",
      "province": "North Western",
      "emailAddress": "nadun@gmail.com"
    }
  ]
}

```

The screenshot shows the Postman interface with a JSON response. The URL in the address bar is `localhost/restfull/horizonstudents`. The response is a list of student objects, each containing properties like indexNumber, firstName, lastName, city, district, province, emailAddress, and mobileNumber. There are four student objects in the list.

```

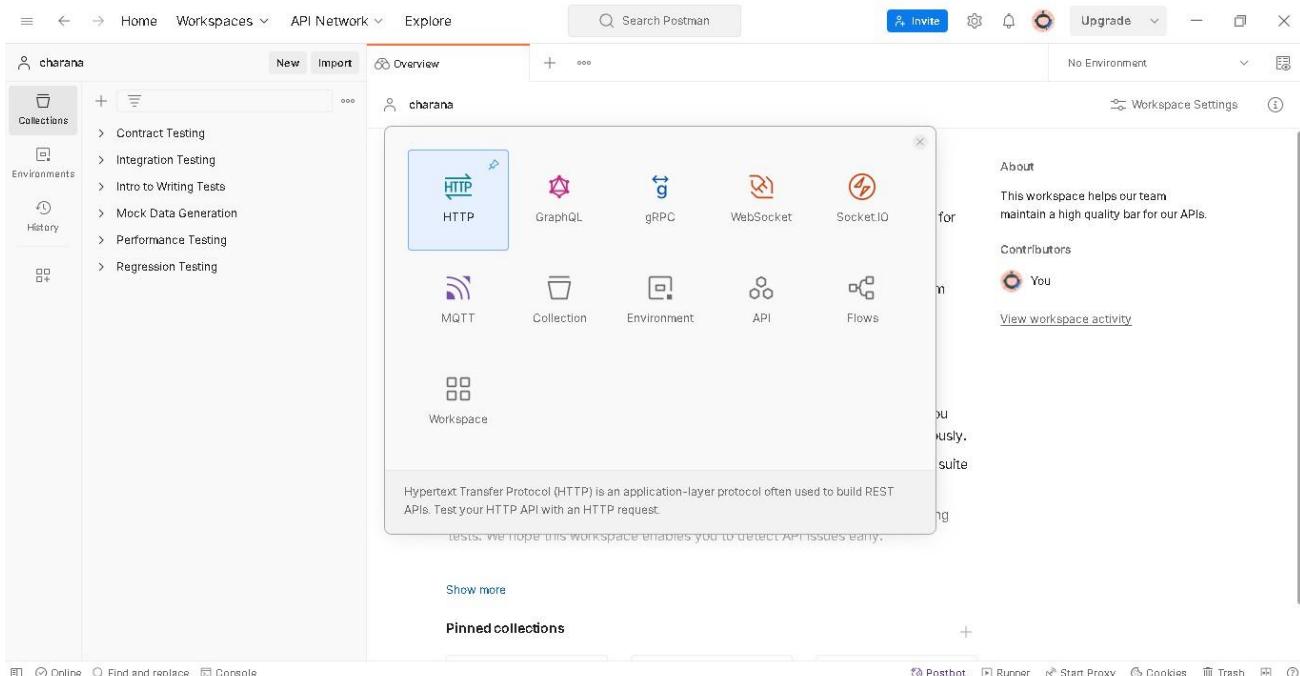
[{"indexNumber": "3", "firstName": "nadin", "lastName": "kalana", "city": "kurunagala", "district": "kurunagala", "province": "North Western", "emailAddress": "nadin@gmail.com", "mobileNumber": "0774569870"}, {"indexNumber": "4", "firstName": "anjaliaka", "lastName": "sewandu", "city": "kalubowila", "district": "colombo", "province": "western", "emailAddress": "anjaliaka@gmail.com", "mobileNumber": "0778965123"}, {"indexNumber": "5", "firstName": "chanuka", "lastName": "dilshan", "city": "nawalapitiya", "district": "kandy", "province": "central", "emailAddress": "chanukap@gmail.com", "mobileNumber": "07693256452"}]

```

- My API url is <http://localhost/restfull/horizonstudent> by using this url only postman tool response the users requests.

c. Test I created API using the Postman tool.

- ✚ To test my API, first I opened the Postman software tool and copied the URL from my browser's address bar. Then, I pasted it into Postman and selected the GET method to retrieve all details about the students from the database. You can see it below:



The screenshot shows the Postman application interface. On the left, there's a sidebar with 'charana' selected under 'Collections', and a list of testing categories: Contract Testing, Integration Testing, Intro to Writing Tests, Mock Data Generation, Performance Testing, and Regression Testing. The main area is titled 'Overview' and shows a 'GET' request to 'http://localhost/restful/horizonstudents'. The 'Params' tab is active, showing a single 'Key' entry. Below it, the 'Response' section displays a cartoon character holding a megaphone with the text 'Click Send to get a response'. At the bottom, there are various status indicators and links like 'Postbot', 'Runner', 'Start Proxy', 'Cookies', and 'Trash'.

Now I send the request. If the request is sent successfully, the tool will display the status as '200 OK' and the message as 'Successfully fetch students'. After that, the tool will display the results. I will show you below.

This screenshot shows the same Postman interface after the request has been sent. The status bar at the bottom now indicates 'Status: 200 OK' and 'Time: 230 ms'. The 'Body' tab in the response panel is selected, displaying the JSON response:

```

1  {
2   "status": 200,
3   "message": "Successfully fetch students",
4   "data": [
5     {
6       "indexNumber": "1",
7       "firstName": "charana",
8       "lastName": "thenuka",
9       "city": "piliyandala",
10      "district": "colombo",
11      "province": "western",
12      "emailAddress": "charana@gmail.com",
13      "mobileNumber": "0774865126"
14    }
15  }

```

The screenshot shows the Postman application interface. On the left, there's a sidebar with sections for Collections, Environments, and History. The main area displays a collection named 'charana'. A specific API endpoint is selected: `GET http://localhost/restfull/horizonstudents`. The response status is 200 OK, and the response body is a JSON array of student objects:

```
14, 15, { 16, "indexNumber": "2", 17, "firstName": "nimantha", 18, "lastName": "prasad", 19, "city": "diyagama", 20, "district": "colombo", 21, "province": "western", 22, "emailAddress": "nimantha@gmail.com", 23, "mobileNumber": "0774569870" 24, }, 25, { 26, "indexNumber": "3", 27, "firstName": "nadun", 28, "lastName": "kalana", 29, "city": "kurunegala", 30, "district": "kurunegala", 31, "province": "North Western", 32, "emailAddress": "nadun@gmail.com", 33, "mobileNumber": "0713264782" 34, },
```

This screenshot continues the JSON response from the previous one. It shows the remaining part of the array of student objects:

```
34, }, 35, { 36, "indexNumber": "4", 37, "firstName": "anjaliika", 38, "lastName": "sewandhi", 39, "city": "kalubowila", 40, "district": "colombo", 41, "province": "western", 42, "emailAddress": "anjaliika@gmail.com", 43, "mobileNumber": "0778965123" 44, }, 45, { 46, "indexNumber": "5", 47, "firstName": "chanuka", 48, "lastName": "dilshan", 49, "city": "nawalapitiya", 50, "district": "kandy", 51, "province": "central", 52, "emailAddress": "chanuka@gmail.com", 53, "mobileNumber": "0783256451" 54, },
```

- After completing the GET method, I selected POST as the method and sent the request. Then, the tool displayed the message 'Enter your first name'. I will show you below,

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'Collections' (Contract Testing, Integration Testing, etc.), 'Environments', and 'History'. The main area shows a 'POST' request to 'http://localhost/restfull/horizonstudents'. The 'Body' tab is selected, showing a JSON object with 'status': 422 and 'message': 'enter your first Name'. The 'Pretty' view shows the same JSON. Below the request, the response status is listed as 'Status: 422 Unprocessable entity'.

- Now I enter the details of a new student as raw data and select the text as JSON, because I am entering the data in JSON format.

This screenshot shows the same Postman interface as above, but the 'Body' tab is now set to 'raw' and contains raw JSON data for a student: {"firstName": "kasun", "lastName": "kumara", "city": "abilipitiya", "district": "rathnapura", "province": "sabaragamuwa", "emailAddress": "sabaragamuwa@gmail.com", "mobileNumber": "0773895451"}. The response status remains 'Status: 422 Unprocessable entity'.

- After completing the add details as JSON format I sent the request, now the system displays message as “**student Insert Successfully**”, you can see below,

The screenshot shows the Postman interface with a collection named "charana". A POST request is made to `http://localhost/restful/horizonstudents`. The request body is a JSON object:

```

1  {
2   "firstName": "kasun",
3   "lastName": "kumara",
4   "city": "abilipitiya",
5   "district": "rathnapura",
6   "province": "sabaragamuwa",
7   "emailAddress": "sabaragamuwa@gmail.com",
8   "mobileNumber": "0773895451"
9 }

```

The response status is 201 Created, and the message is "student Insert Sucessfully".

- Now I show you a newly created student. By using Postman tool and database. The newly created student's index number is 6.

The screenshot shows the Postman interface with a collection named "charana". A GET request is made to `http://localhost/restful/horizonstudents?=`. The response body is a JSON object:

```

54  },
55  {
56   "indexNumber": "6",
57   "firstName": "kasun",
58   "lastName": "kumara",
59   "city": "abilipitiya",
60   "district": "rathnapura",
61   "province": "sabaragamuwa",
62   "emailAddress": "sabaragamuwa@gmail.com",
63   "mobileNumber": "0773895451"
64 }
65 ]
66

```

Showing rows 0 - 5 (6 total, Query took 0.0084 seconds)

	indexNumber	firstName	lastName	city	district	province	emailAddress	mobileNumber
<input type="checkbox"/>	1	charana	thenuka	piliyandala	colombo	western	charana@gmail.com	0774865126
<input type="checkbox"/>	2	nimantha	prasad	diyagama	colombo	western	nimantha@gmail.com	0774569870
<input type="checkbox"/>	3	nadun	kalana	kurunagala	kurunagala	North Western	nadun@gmail.com	0713254782
<input type="checkbox"/>	4	anjalika	seewandi	kalubowila	colombo	western	anjalika@gmail.com	0778965123
<input type="checkbox"/>	5	chanuka	dilshan	awalapitiya	kandy	central	chanuka@gmail.com	0783256451
<input type="checkbox"/>	6	kasun	kumara	abilipitiya	rathnapura	sabaragamuwa	sabaragamuwa@gmail.com	0773895451

it also inserts in API,

```

{
  "data": [
    {
      "indexNumber": "1",
      "firstName": "charana",
      "lastName": "thenuka",
      "city": "piliyandala",
      "district": "colombo",
      "province": "western",
      "emailAddress": "charana@gmail.com",
      "mobileNumber": "0774865126"
    },
    {
      "indexNumber": "2",
      "firstName": "nimantha",
      "lastName": "prasad",
      "city": "diyagama",
      "district": "colombo",
      "province": "western",
      "emailAddress": "nimantha@gmail.com",
      "mobileNumber": "0774569870"
    },
    {
      "indexNumber": "3",
      "firstName": "nadun",
      "lastName": "kalana",
      "city": "kurunagala",
      "district": "kurunagala",
      "province": "North Western",
      "emailAddress": "nadun@gmail.com",
      "mobileNumber": "0713254782"
    },
    {
      "indexNumber": "4",
      "firstName": "anjalika",
      "lastName": "seewandi",
      "city": "kalubowila",
      "district": "colombo",
      "province": "western",
      "emailAddress": "anjalika@gmail.com",
      "mobileNumber": "0778965123"
    },
    {
      "indexNumber": "5",
      "firstName": "chanuka",
      "lastName": "dilshan",
      "city": "awalapitiya",
      "district": "kandy",
      "province": "central",
      "emailAddress": "chanuka@gmail.com",
      "mobileNumber": "0783256451"
    },
    {
      "indexNumber": "6",
      "firstName": "kasun",
      "lastName": "kumara",
      "city": "abilipitiya",
      "district": "rathnapura",
      "province": "sabaragamuwa",
      "emailAddress": "sabaragamuwa@gmail.com",
      "mobileNumber": "0773895451"
    }
  ]
}

```

- Now I select **PUT** as the method and send the request. The Postman tool displays the message 'Student Index Number Not found in URL'. The PUT method is used to update the selected field, so I need to provide a student index number.

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'charana' selected under 'Collections', followed by 'Environments' and 'History'. The main workspace shows a 'Overview' tab for 'http://localhost/restfull/horizonstudents'. A 'PUT' request is selected with the URL 'http://localhost/restfull/horizonstudents'. The 'Params' tab is active, showing a table with one row: 'Key' and 'Value' both are empty. Below the table, the 'Body' tab is selected, showing a JSON response with status 422 and message 'student Index Number Not found in URL'. At the bottom, the status bar indicates 'Status: 422 Unprocessable entity Time: 7 ms Size: 323 B'.

- When I provide a student index number as 6 (newly created student), then the Postman tool display the message “enter your first Name”. you can see below,

This screenshot shows the same Postman interface as the previous one, but with a successful response. The 'Params' tab now has a checked checkbox next to 'indexNumber', with the value '6' entered. The 'Body' tab shows a JSON response with status 422 and message 'enter your first Name'. The status bar at the bottom shows 'Status: 422 Unprocessable entity Time: 98 ms Size: 307 B'.

Now I enter the new details for student 6 , you can see in below,

The screenshot shows the Postman interface. In the left sidebar, there's a collection named 'charana' containing various testing categories. The main workspace displays a PUT request to 'http://localhost:restfull/horizonstudents?indexNumber=6'. The 'Body' tab is selected, showing a JSON payload:

```

1 {
2   "firstName": "lalani",
3   "lastName": "kumari",
4   "city": "homagama",
5   "district": "colombo",
6   "province": "western",
7   "emailAddress": "lalani@gmail.com",
8   "mobileNumber": "07778904451"
9 }

```

Below the request, the response status is shown as 422 Unprocessable entity with a message: "enter your first Name".

Now I send the request then postman tool display the message “student Updated Sucessfully”. Now I check by using the GET method, number 6 student update or not. You can see in below,

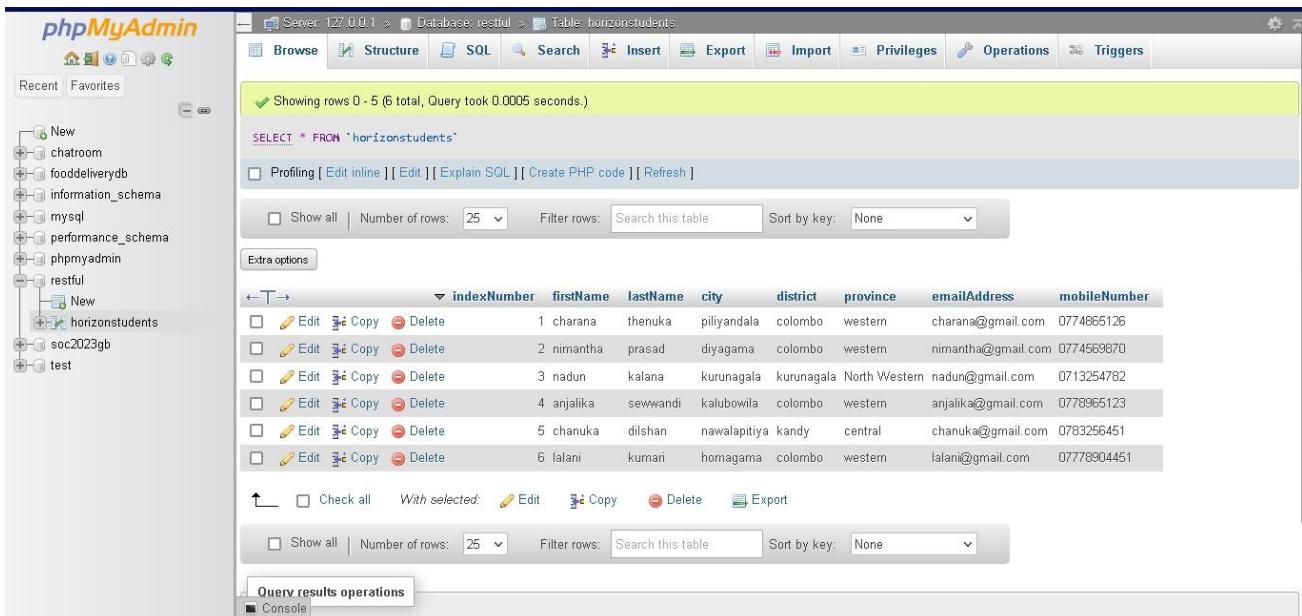
The screenshot shows the Postman interface with a GET request to 'http://localhost:restfull/horizonstudents?='. The 'Headers' tab is selected, showing 'Content-Type: application/json'. Below the request, the response status is 200 OK, indicating success. The response body contains the updated student details:

```

54 },
55 {
56   "indexNumber": "6",
57   "firstName": "lalani",
58   "lastName": "kumari",
59   "city": "homagama",
60   "district": "colombo",
61   "province": "western",
62   "emailAddress": "lalani@gmail.com",
63   "mobileNumber": "07778904451"
64 }
65 ]
66

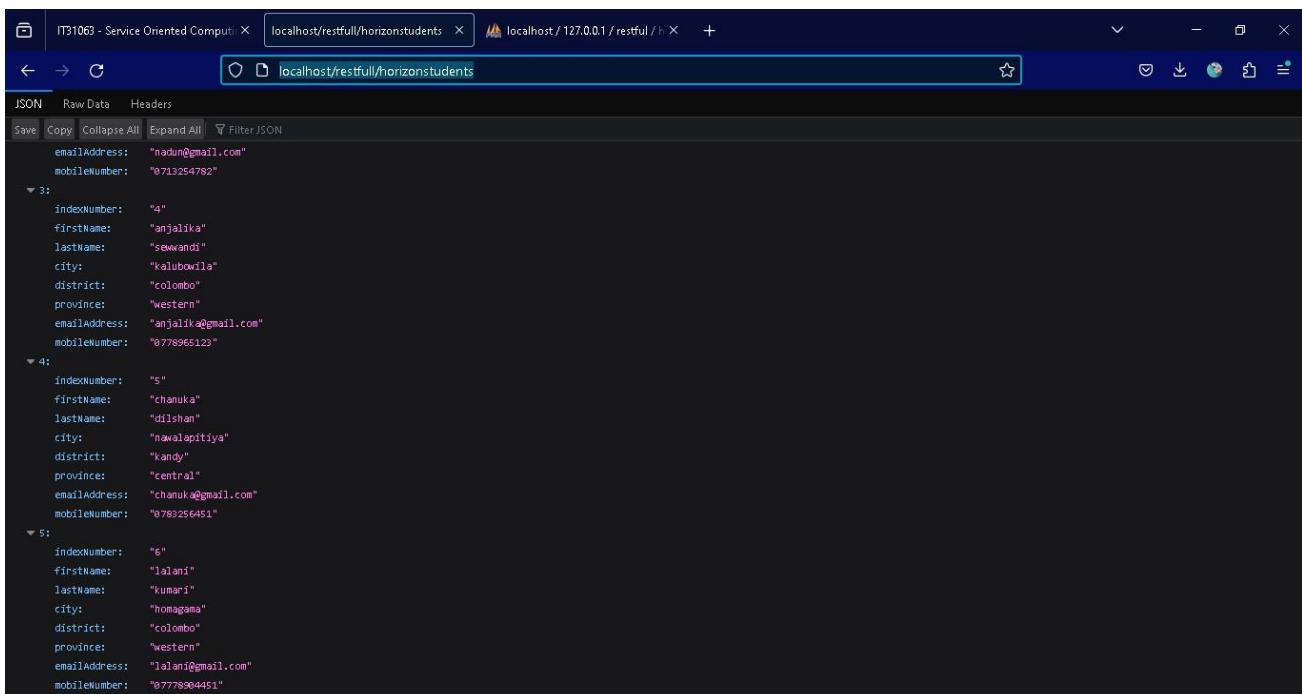
```

 You can see the number 6 student's details changed, now I show this by using database and API in like below screenshots



The screenshot shows the phpMyAdmin interface for the 'horizonstudents' table. The table has columns: indexNumber, firstName, lastName, city, district, province, emailAddress, and mobileNumber. The data is as follows:

	indexNumber	firstName	lastName	city	district	province	emailAddress	mobileNumber
1	1	charana	thenuka	piiyandala	colombo	western	charana@gmail.com	0774865126
2	2	nimantha	prasad	diyagama	colombo	western	nimantha@gmail.com	0774569870
3	3	nadun	kalana	kurunagala	kurunagala	North Western	nadun@gmail.com	0713254782
4	4	anjalika	seewandi	kalubowila	colombo	western	anjalika@gmail.com	0778965123
5	5	chanuka	dilshan	nawalapitiya	kandy	central	chanuka@gmail.com	0783256451
6	6	lalani	kumari	homagama	colombo	western	lalani@gmail.com	07778904451



The screenshot shows a browser window displaying the JSON response of the 'horizonstudents' API. The response is a list of 6 students, each with fields: indexNumber, firstName, lastName, city, district, province, emailAddress, and mobileNumber. The data matches the table above.

```

[{"indexNumber": "1", "firstName": "charana", "lastName": "thenuka", "city": "piiyandala", "district": "colombo", "province": "western", "emailAddress": "charana@gmail.com", "mobileNumber": "0774865126"}, {"indexNumber": "2", "firstName": "nimantha", "lastName": "prasad", "city": "diyagama", "district": "colombo", "province": "western", "emailAddress": "nimantha@gmail.com", "mobileNumber": "0774569870"}, {"indexNumber": "3", "firstName": "nadun", "lastName": "kalana", "city": "kurunagala", "district": "kurunagala", "province": "North Western", "emailAddress": "nadun@gmail.com", "mobileNumber": "0713254782"}, {"indexNumber": "4", "firstName": "anjalika", "lastName": "seewandi", "city": "kalubowila", "district": "colombo", "province": "western", "emailAddress": "anjalika@gmail.com", "mobileNumber": "0778965123"}, {"indexNumber": "5", "firstName": "chanuka", "lastName": "dilshan", "city": "nawalapitiya", "district": "kandy", "province": "central", "emailAddress": "chanuka@gmail.com", "mobileNumber": "0783256451"}, {"indexNumber": "6", "firstName": "lalani", "lastName": "kumari", "city": "homagama", "district": "colombo", "province": "western", "emailAddress": "lalani@gmail.com", "mobileNumber": "07778904451"}]

```

- Finally I changed the method to **DELETE**. When I send the request, the Postman tool displays the message as “**student Index Number Not found in URL**”. This means that the index number of the student to be deleted has not been added to the URL. Therefore I add the newly created student index number which is 6.

The screenshot shows the Postman interface with a collection named 'charana'. A new request is being prepared with the URL `http://localhost:restful/horizonstudents?indexNumber=6`. The method is set to **DELETE**. In the 'Query Params' section, there is a single entry for 'indexNumber' with a value of '6'. The 'Body' tab is selected, showing a JSON response with a status of 422 and a message: "status": 422, "message": "student Index Number Not found in URL".

- now I send the request and the Postman tool displays the message as “**student Delete Successfully**”. you can see in the below screenshot.

The screenshot shows the Postman interface with the same setup as the previous one, but the response has changed. The status is now 200 OK, and the message is "status": 200, "message": "student delete successfully".

- ✍ I show my database to you database and the API student 6 details are deleted or not.

The screenshot shows the phpMyAdmin interface for the 'horizonstudents' table. The table has columns: indexNumber, firstName, lastName, city, district, province, emailAddress, and mobileNumber. The data is as follows:

	indexNumber	firstName	lastName	city	district	province	emailAddress	mobileNumber
1	charana	thenuka	piliyandala	colombo	western		charana@gmail.com	0774865126
2	nimantha	prasad	diyagama	colombo	western		nimantha@gmail.com	0774569870
3	nadun	kalana	kurunagala	kurunagala	North Western		nadun@gmail.com	0713254782
4	anjalika	seewandi	kalubowila	colombo	western		anjalika@gmail.com	0778965123
5	chanuka	dilshan	newalapitiya	kandy	central		chanuka@gmail.com	0783256451

The screenshot shows the Postman interface displaying the JSON response for the 'horizonstudents' collection. The data is identical to the one shown in the phpMyAdmin screenshot, with 5 entries. The JSON structure is as follows:

```

{
  "0": {
    "indexNumber": "1",
    "firstName": "charana",
    "lastName": "thenuka",
    "city": "piliyandala",
    "district": "colombo",
    "province": "western",
    "emailAddress": "charana@gmail.com",
    "mobileNumber": "0774865126"
  },
  "1": {
    "indexNumber": "2",
    "firstName": "nimantha",
    "lastName": "prasad",
    "city": "diyagama",
    "district": "colombo",
    "province": "western",
    "emailAddress": "nimantha@gmail.com",
    "mobileNumber": "0774569870"
  },
  "2": {
    "indexNumber": "3",
    "firstName": "nadun",
    "lastName": "kalana",
    "city": "kurunagala",
    "district": "kurunagala",
    "province": "North Western",
    "emailAddress": "nadun@gmail.com",
    "mobileNumber": "0713254782"
  },
  "3": {
    "indexNumber": "4",
    "firstName": "anjalika",
    "lastName": "seewandi",
    "city": "kalubowila",
    "district": "colombo",
    "province": "western",
    "emailAddress": "anjalika@gmail.com",
    "mobileNumber": "0778965123"
  },
  "4": {
    "indexNumber": "5",
    "firstName": "chanuka",
    "lastName": "dilshan",
    "city": "newalapitiya",
    "district": "kandy",
    "province": "central",
    "emailAddress": "chanuka@gmail.com",
    "mobileNumber": "0783256451"
  }
}

```

- likewise, you can see all GET, POST, PUT, and DELETE methods checked by using the Postman tool. All the methods are working properly.

- ⊕ Now I used the method as **PATCH**, then the Postman tool display the message as “PATCH Method Not Allowed” because I wrote code only **GET**, **POST**, **PUT**, and **DELETE**. You can see below screenshot.

The screenshot shows the Postman interface with the following details:

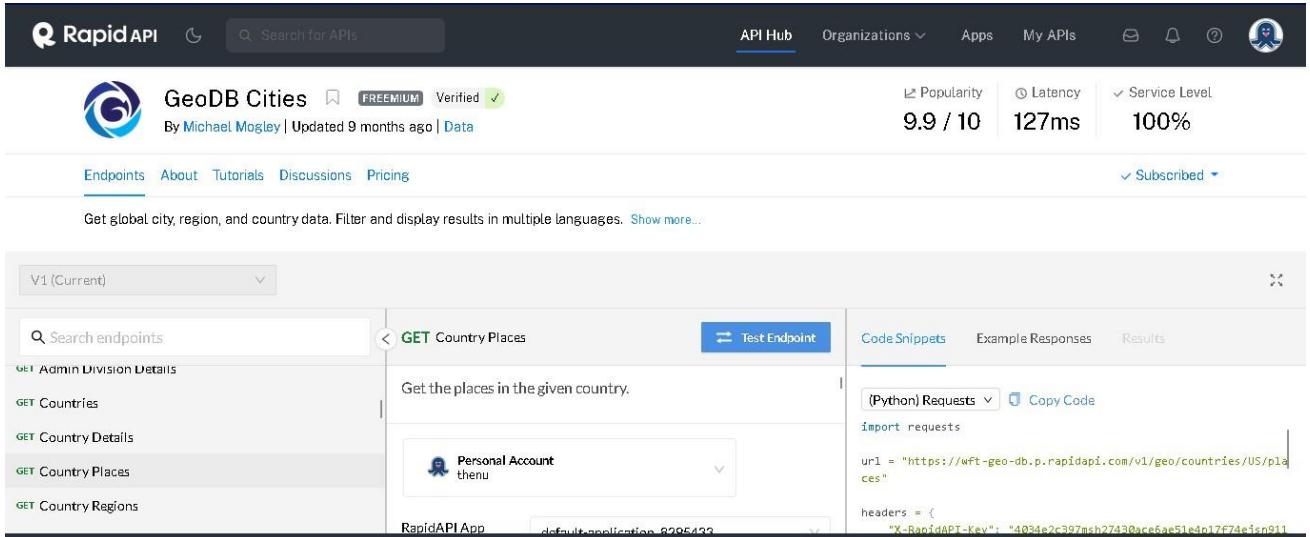
- Left Sidebar:** Collections (Contract Testing, Integration Testing, Intro to Writing Tests, Mock Data Generation, Performance Testing, Regression Testing), Environments, History.
- Top Bar:** Home, Workspaces, API Network, Explore, Search Postman, Invite, Upgrade, No Environment.
- Request URL:** PATCH http://localhost/restfull/horizonstudents
- Method:** PATCH
- Params:** Key, Value, Description
- Body:** Status: 405 Method Not Allowed, Time: 51 ms, Size: 307 B
- Response Body (Pretty JSON):**

```
1 {  
2   "status": 405,  
3   "message": "PATCH Method Not Allowed"  
4 }
```
- Bottom Tools:** Online, Find and replace, Console, Postbot, Runner, Start Proxy, Cookies, Trash.

Question 4

4. Access <https://rapidapi.com/wirefreethought/api/geodb-cities> Rest API and get country places in New Zealand using Python and display “place id”, “place name”, “place type”, and “region code” by using the panda library.

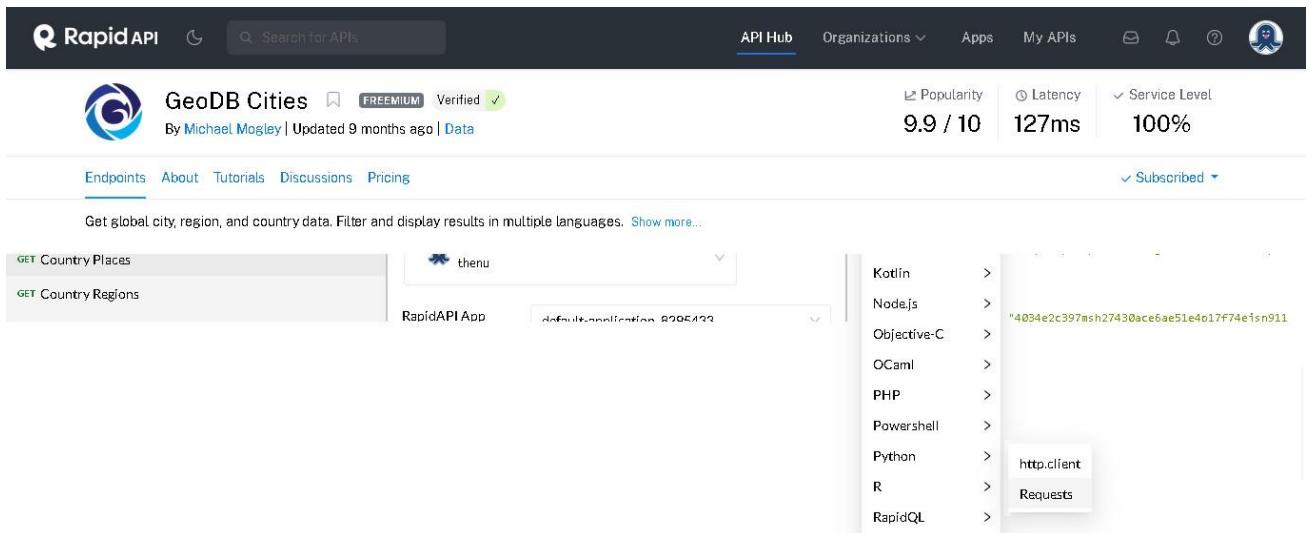
➤ First I go to the provided link and then I choose “get country places”, like below



The screenshot shows the RapidAPI interface for the GeoDB Cities API. The endpoint selected is 'GET Country Places'. The description is 'Get the places in the given country.' Below the description is a dropdown menu labeled 'Personal Account' with 'thenu' selected. To the right, there are tabs for 'Code Snippets', 'Example Responses', and 'Results'. Under 'Code Snippets', the language is set to '(Python) Requests' and the code provided is:

```
url = "https://wft-geo-db.p.rapidapi.com/v1/geo/countries/US/places"
headers = {
    "X-RapidAPI-Key": "4034e2c397msh27430ace6ae51e4b17f74eisn911"
}
```

➤ Thereafter I choose code Snippets as (python) Request, like below.



The screenshot shows the same RapidAPI interface for the GeoDB Cities API, but the 'Code Snippets' tab is now active. It lists various programming languages under the '(Python) Requests' section. The 'Python' section is expanded, showing the same code as the previous screenshot:

```
url = "https://wft-geo-db.p.rapidapi.com/v1/geo/countries/US/places"
headers = {
    "X-RapidAPI-Key": "4034e2c397msh27430ace6ae51e4b17f74eisn911"
}
```

- Now I copy that code and paste then I saw this provided url gives only United States (US) details, like below

```
url = "https://wft-geo-db.p.rapidapi.com/v1/geo/countries/US/places"
```

- Therefor I changed this country code (US) to nz , because New Zealand's country code is nz. Now you can see My modified code. I display only “place id”, “place name”, “place type”, and “region code”, You can see my code and output below,

```
import requests

url = "https://wft-geo-db.p.rapidapi.com/v1/geo/countries/nz/places"

headers = {
    "X-RapidAPI-Key": "4034e2c397msh27430ace6ae51e4p17f74ejsn911be8a1ae04",
    "X-RapidAPI-Host": "wft-geo-db.p.rapidapi.com"
}

response = requests.get(url, headers=headers)

data=(response.json())
data=data['data']
import pandas as pd
df=pd.DataFrame(data)
df[['id' , 'name' , 'type' , 'regionCode']]
```

	id	name	type	regionCode	
0	3458993	Abbotsford	CITY	OTA	
1	3401902	Acacia Bay	CITY	WKO	
2	3529320	Adair	CITY	CAN	
3	3617971	Adams Island	ISLAND	NaN	
4	3357962	Addington	CITY	CAN	

References

1. Anon., n.d. [Online]
Available at:
https://colab.research.google.com/github/Rblivingstone/iMaterialist_Kaggle_competition/blob/master/Untitled0.ipynb?pli=1#scrollTo=NOhr7LV7aN7u
2. Anon., n.d. [Online]
Available at: <https://getbootstrap.com/docs/5.0/getting-started/introduction/>
3. Anon., n.d. *Find Color On Image, Match PMS Colors.* [Online]
Available at: https://www.ginifab.com/feeds/pms/pms_color_in_image.php#google_vignette
4. Anon., n.d. *W3 school.* [Online]
Available at: https://www.w3schools.com/bootstrap5/bootstrap_get_started.php