

CSC Assignment 1 Documentation

Task 1

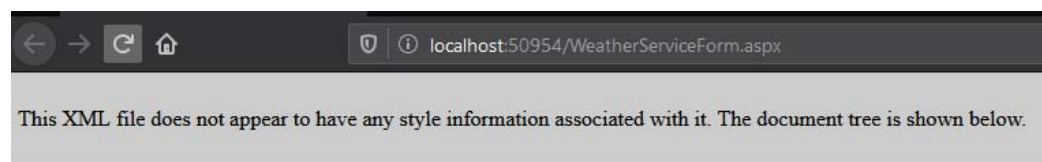
Browser (aspx)

Located in WeatherService Folder

Assign the country, number of days and insert your key in WeatherServiceForm.aspx.cs from <https://www.worldweatheronline.com/developer/>.

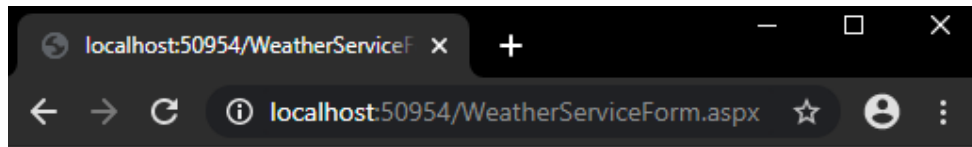
```
//Assign values for query
var country = "china";
var numOfDays = "5";
var key = "*****";
```

Debug run the application and you will see the following page on load.



```
<data>
  <request>
    <type>City</type>
    <query>Beijing, China</query>
  </request>
  <current_condition>
    <observation_time>05:03 AM</observation_time>
    <temp_C>6</temp_C>
    <temp_F>43</temp_F>
    <weatherCode>113</weatherCode>
  </current_condition>
  <weatherIconUrl>
    http://cdn.worldweatheronline.net/images/wsymbols01_png_64/wsymbol_0001_sunny.png
  </weatherIconUrl>
  <weatherDesc>Sunny</weatherDesc>
  <windspeedMiles>4</windspeedMiles>
  <windspeedKmph>7</windspeedKmph>
  <winddirDegree>320</winddirDegree>
  <winddir16Point>NW</winddir16Point>
  <precipMM>0.0</precipMM>
  <precipInches>0.0</precipInches>
  <humidity>26</humidity>
  <visibility>10</visibility>
  <visibilityMiles>6</visibilityMiles>
  <pressure>1029</pressure>
  <pressureInches>31</pressureInches>
  <cloudcover>0</cloudcover>
  <FeelsLikeC>5</FeelsLikeC>
  <FeelsLikeF>40</FeelsLikeF>
  <uvIndex>3</uvIndex>
</current_condition>
```

In the case of error, the web service will ask u to verify your API key as well as your condition inputs.



Error accessing web service

Reconnecting in 5s...



Please check the following:

- Network connection
- API key : XXXXXXXXXXXXXXXXXXXXXXXXXX
- Country selected : china
- Number of days selected : 5

Waiting for localhost...

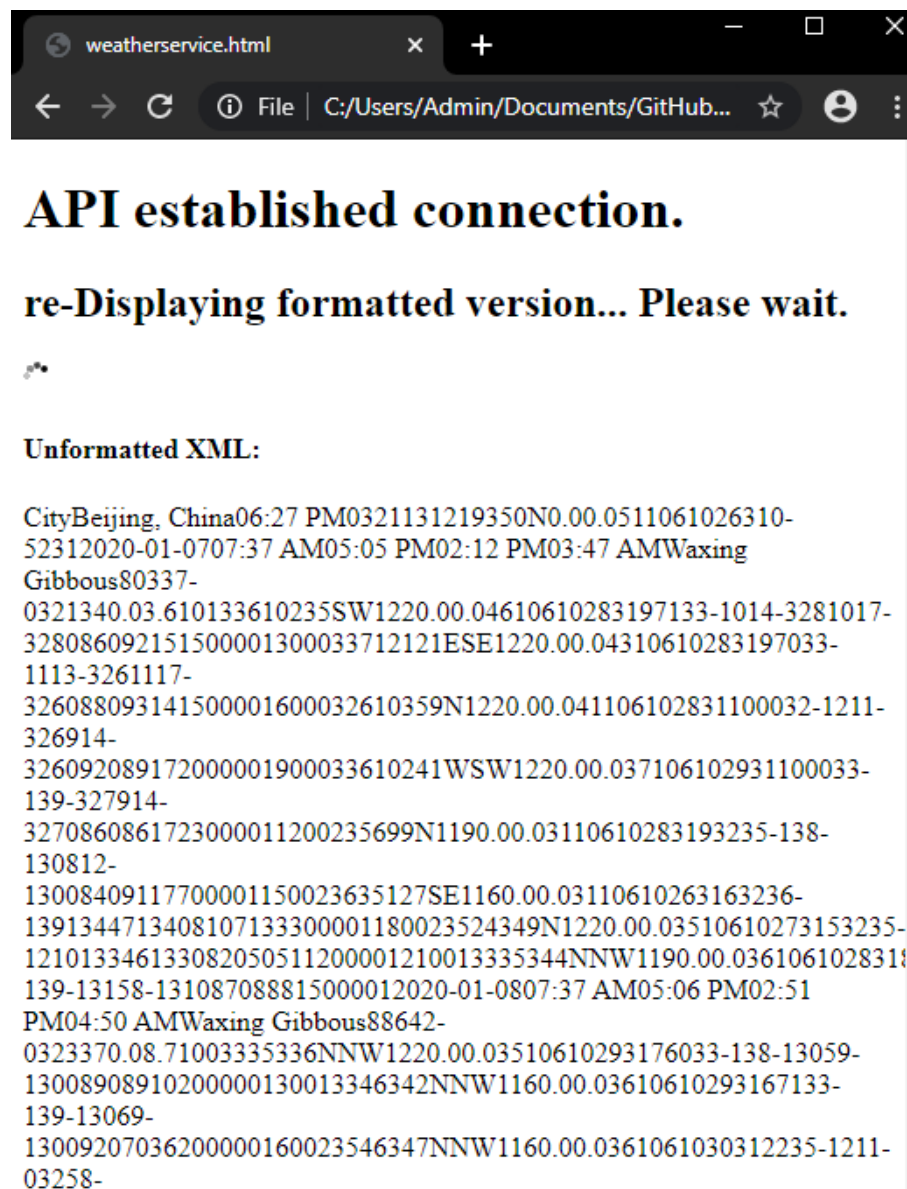
Browser (jQuery)

Located in WeatherService_js Folder

Assign the country, number of days and insert your key in script.js from <https://www.worldweatheronline.com/developer/>.

```
var country = "china";  
var numOfDays = "5";  
var key = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX";
```

Debug run the application and you will see the following page on API successful connection.



The screenshot shows a web browser window with the address bar displaying "weatherservice.html". The page content includes a large heading "API established connection." followed by the text "re-Displaying formatted version... Please wait." and a loading spinner. Below this, the text "Unformatted XML:" is displayed, followed by a large block of unformatted XML data. The XML data is a single line of text containing various weather-related information, including city names, coordinates, and timestamps.

API established connection.

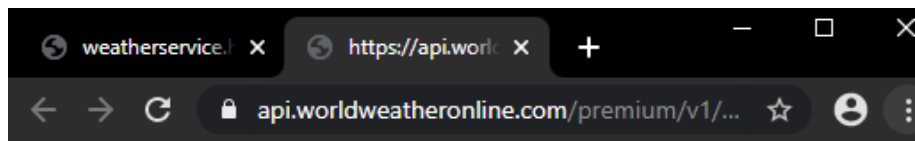
re-Displaying formatted version... Please wait.

☼

Unformatted XML:

CityBeijing, China06:27 PM0321131219350N0.00.0511061026310-52312020-01-0707:37 AM05:05 PM02:12 PM03:47 AMWaxing Gibbous80337-0321340.03.610133610235SW1220.00.04610610283197133-1014-3281017-328086092151500001300033712121ESE1220.00.04310610283197033-1113-3261117-326088093141500001600032610359N1220.00.041106102831100032-1211-326914-326092089172000001900033610241WSW1220.00.037106102931100033-139-327914-3270860861723000011200235699N1190.00.03110610283193235-138-130812-13008409117700001150023635127SE1160.00.03110610263163236-1391344713408107133300001180023524349N1220.00.03510610273153235-121013346133082050511200001210013335344NNW1190.00.036106102831139-13158-131087088815000012020-01-0807:37 AM05:06 PM02:51 PM04:50 AMWaxing Gibbous88642-0323370.08.71003335336NNW1220.00.03510610293176033-138-13059-13008908910200000130013346342NNW1160.00.03610610293167133-139-13069-13009207036200000160023546347NNW1160.00.0361061030312235-1211-03258-

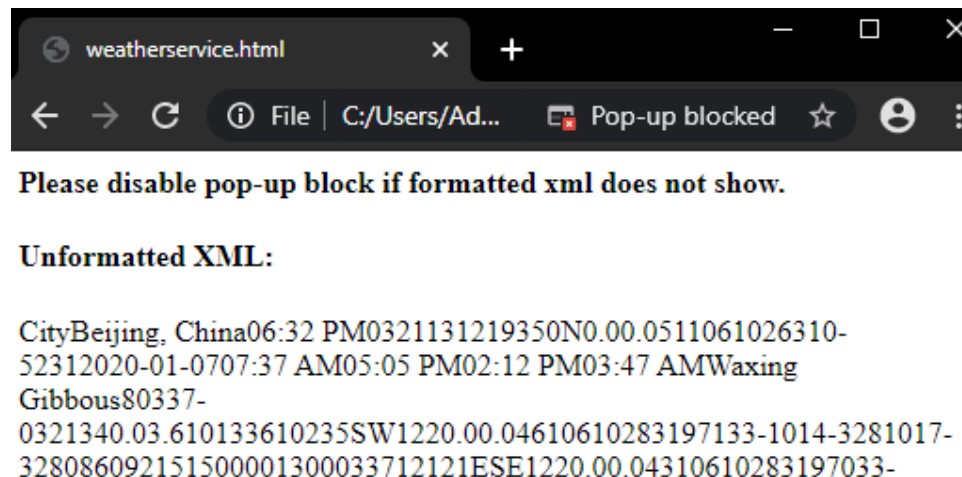
A formatted XML will be pop up in a new window. (Direct XML file from world weather)



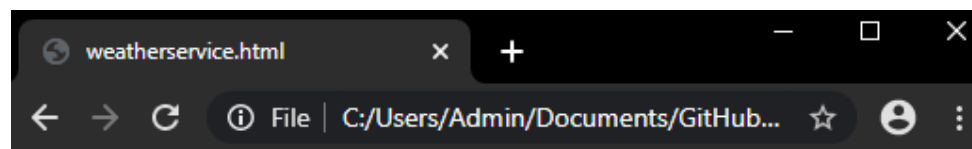
This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<data>
  <request>
    <type>City</type>
    <query>Beijing, China</query>
  </request>
  <current_condition>
    <observation_time>06:29 PM</observation_time>
    <temp_C>0</temp_C>
    <temp_F>32</temp_F>
    <weatherCode>113</weatherCode>
    <weatherIconUrl>
      <![CDATA[
        http://cdn.worldweatheronline.net/images/wsymbols01_png_64/ws
      ]]>
    </weatherIconUrl>
    <weatherDesc>
      <![CDATA[ Clear ]]>
    </weatherDesc>
    <windspeedMiles>12</windspeedMiles>
    <windspeedKmph>19</windspeedKmph>
    <winddirDegree>350</winddirDegree>
    <winddir16Point>N</winddir16Point>
    <precipMM>0.0</precipMM>
    <precipInches>0.0</precipInches>
    <humidity>51</humidity>
    <visibility>10</visibility>
    <visibilityMiles>6</visibilityMiles>
    <pressure>1026</pressure>
    <pressureInches>31</pressureInches>
    <cloudcover>0</cloudcover>
    <FeelsLikeC>-5</FeelsLikeC>
    <FeelsLikeF>23</FeelsLikeF>
    <uvIndex>0</uvIndex>
  </current_condition>
</data>
```

In the case of formatted XML not popping up, enable or allow pop-up window.



In the case of error, client will reconnect and prompt to check common user input mistakes.



API connection failed.

Reconnecting in 5s...

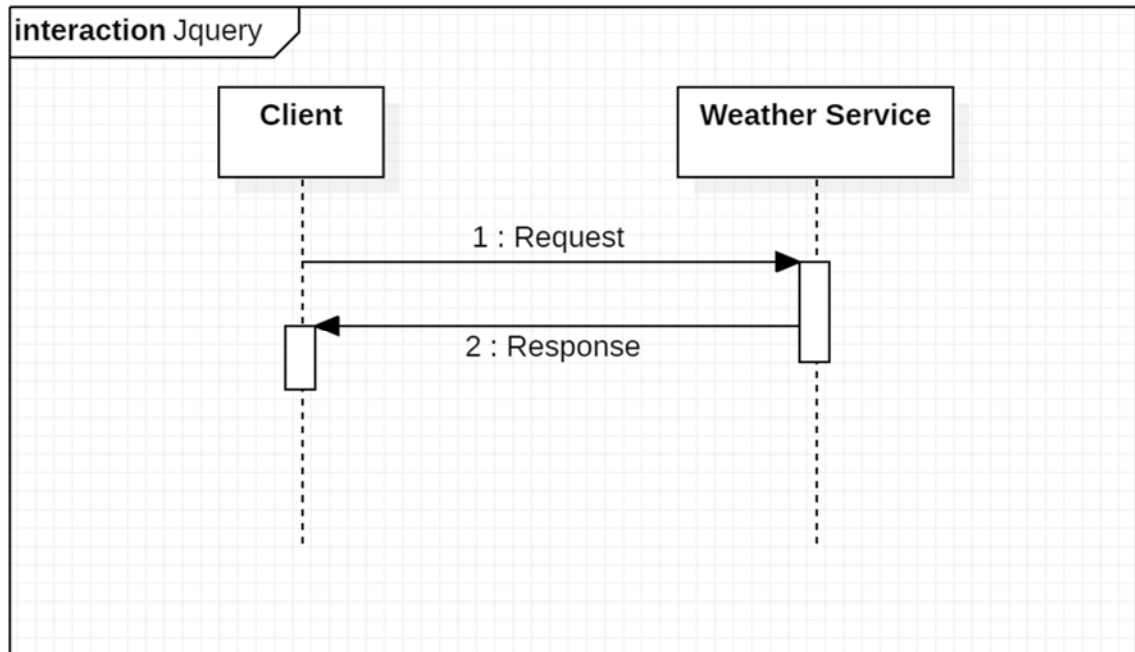


Please check the following:

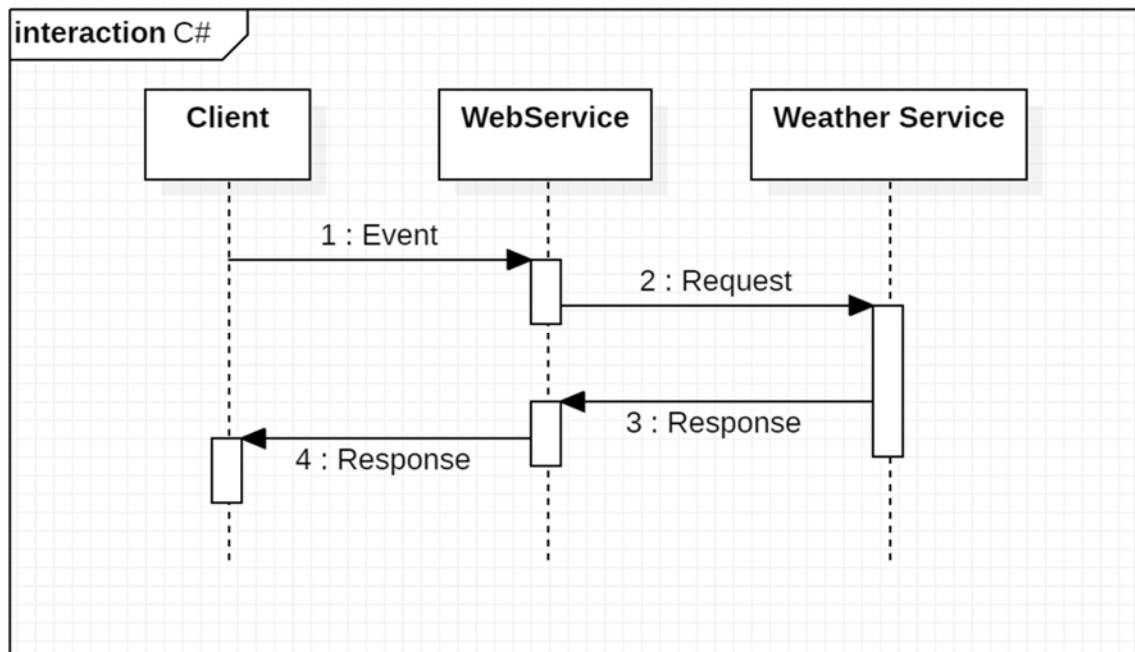
- Network connection
- API key
- Country selected
- Number of days selected

Sequence Diagrams

Jquery:



aspx:



Task 2

Browser

Debug run the application and you will see the following page on load.

(There will be a simulated loading)



All Products

- Tomato Soup: \$1
- Yo-yo: \$3.75
- Hammer: \$16.99

Search by ID

Enter a number between 1 to 3 (ID for the products).

Search by ID

Tomato Soup: \$1

In the case of error, client will reconnect and prompt to check.

All Products

❌ Fail to get products. Reloading... (Check your Uri in the project)

Search by ID

Postman

Localhost:

<http://localhost:60125/>

APIs:

Version	Type	API Route
V1	GET	api/v1/products/version
V1	GET	api/v1/products/message?name1=<name>&name2=<name>&name3=<name>
V1	GET	api/v1/products
V1	GET	api/v1/products/{id}
V2	GET	api/v2/products
V2	GET	api/v2/products/{id}
V2	GET	api/v2/products?Category=<category>
V2	POST	api/v2/products
V2	PUT	api/v2/products/{id}
V2	DELETE	api/v2/products/{id}
V3	GET	api/v3/products
V3	GET	api/v3/products/{id}
V3	GET	api/v3/products?Category=<category>
V3	POST	api/v3/products
V3	PUT	api/v3/products/{id}
V3	DELETE	api/v3/products/{id}

GET: api/v1/products/version

Link: <http://localhost:60125/api/v1/products/version>

Get API version.

GET

http://localhost:60125/api/v1/products/version

Params

Authorization

Headers (9)

Body ●

Pre-request S

Query Params

	KEY	VALUE
	Key	Value

Body

Cookies

Headers (10)

Test Results

Status: 200 OK

Pretty

Raw

Preview

Visualize BETA

JSON

1

2

3

4

5

[
 "hello",
 "version 1",
 "1"
]

GET: api/v1/products/message?name1=<name>&name2=<name>&name3=<name>

Link: <http://localhost:60125/api/v1/products/message/name1=ji&name2=jii1&name3=ji3>

Get method with user defined parameters.

GET

http://localhost:60125/api/v1/products/message?name1=ji&na...

Send

S

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Coc

Query Params

	KEY	VALUE	DESCRIPTION	...
<input checked="" type="checkbox"/>	name1	ji		
<input checked="" type="checkbox"/>	name2	jii1		
<input checked="" type="checkbox"/>	name3	ji3		
	Key	Value	Description	

Body

Cookies

Headers (10)

Test Results

Status: 200 OK

Time: 9ms

Size: 491 B

Save Re

Pretty

Raw

Preview

Visualize BETA

HTML

1

<html>

2

3

<body>Hello World name1 =ji name2= jii1 name3=ji3 is provided in path parameter</body>

4

5

</html>

GET: api/v1/products

Link: <http://localhost:60125/api/v1/products>

Get all products.

The screenshot shows a REST client interface with the following components:

- Method and URL:** GET `http://localhost:60125/api/v1/products`
- Tabs:** Params, Authorization, Headers (9), Body, Pre-request S...
- Query Params Table:**

KEY	VALUE
Key	Value
- Response Status:** Status: 200 OK
- Response Body Tabs:** Body, Cookies, Headers (10), Test Results
- Response Format:** Pretty, Raw, Preview, Visualize BETA, JSON, and a menu icon.
- JSON Response:**

```
1  [
2    {
3      "Id": 1,
4      "Name": "Tomato Soup",
5      "Category": "Groceries",
6      "Price": 1.0
7    },
8    {
9      "Id": 2,
10     "Name": "Yo-yo",
11     "Category": "Toys",
12     "Price": 3.75
13   },
14   {
15     "Id": 3,
16     "Name": "Hammer",
17     "Category": "Hardware",
18     "Price": 16.99
19   }
20 ]
```

GET: api/v1/products/{id}

Link: <http://localhost:60125/api/v1/products/1>

Get product based on id. (Available Id between 1 to 3)

GET <http://localhost:60125/api/v1/products/1>

Params Authorization Headers (9) Body ● Pre-request S

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (10) Test Results Status: 200 OK

Pretty Raw Preview Visualize BETA JSON

```
1 {
2   "Id": 1,
3   "Name": "Tomato Soup",
4   "Category": "Groceries",
5   "Price": 1.0
6 }
```

Providing invalid id will lead to 404 Not Found.

GET <http://localhost:60125/api/v1/products/5>

Params Authorization Headers (9) Body ● Pre-request S

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (9) Test Results Status: 404 Not Found

Pretty Raw Preview Visualize BETA Text

```
1
```

GET: api/v2/products

Link: <http://localhost:60125/api/v2/products>

Get all products.

The screenshot shows a REST client interface with the following components:

- Method and URL:** GET `http://localhost:60125/api/v2/products`
- Tabs:** Params, Authorization, Headers (9), Body, Pre-request S...
- Query Params Table:**

KEY	VALUE
Key	Value
- Response Status:** Status: 200 OK
- Response Format:** Pretty, Raw, Preview, Visualize BETA, JSON, and a refresh icon.
- Response Body (JSON):**

```
[
  {
    "Id": 1,
    "Name": "Tomato soup",
    "Category": "Groceries",
    "Price": 1.39
  },
  {
    "Id": 2,
    "Name": "Yo-yo",
    "Category": "Toys",
    "Price": 3.75
  },
  {
    "Id": 3,
    "Name": "Hammer",
    "Category": "Hardware",
    "Price": 16.99
  }
]
```

GET: api/v2/products/{id}

Link: <http://localhost:60125/api/V2/products>

Get product based on id. (Available Id between 1 to 3)

GET ▼ http://localhost:60125/api/v2/products/1

Params Authorization Headers (9) Body ● Pre-request

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (10) Test Results Status: 200 OK

Pretty Raw Preview Visualize BETA JSON ▼

```
1 {  
2   "Id": 1,  
3   "Name": "Tomato soup",  
4   "Category": "Groceries",  
5   "Price": 1.39  
6 }
```

Providing invalid id will lead to 400 Bad Request

GET ▼ http://localhost:60125/api/v2/products/5

Params Authorization Headers (9) Body ● Pre-request S

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (10) Test Results Status: 400 Bad Request

Pretty Raw Preview Visualize BETA Text ▼

```
1 No Product Id Found: 5
```

GET: api/v2/products?Category=<category>

Link: <http://localhost:60125/api/v2/products?Category=Toys>

Get products by Category as parameter.

The screenshot shows a REST client interface with the following components:

- Method and URL:** GET `http://localhost:60125/api/v2/products?Category=Toys`
- Params:** A table showing query parameters.
- Body:** A JSON response displayed in a pretty-printed format.

	KEY	VALUE
<input checked="" type="checkbox"/>	Category	Toys
	Key	Value

Body: `{ "Id": 2, "Name": "Yo-yo", "Category": "Toys", "Price": 3.75 }`

POST: api/v2/products

Link: <http://localhost:60125/api/v2/products>

Create product using Post.

The screenshot displays a REST client interface with the following components:

- Method and URL:** A dropdown menu shows "POST" and the URL "http://localhost:60125/api/v2/products".
- Request Body:** The "Body" tab is selected, showing a JSON object:

```
{  "Name": "Potato soup",  "Category": "Groceries",  "Price": 2.00}
```

. The "raw" radio button is selected.
- Response:** The "Body" tab is also selected for the response, showing a JSON object:

```
{  "Id": 4,  "Name": "Potato soup",  "Category": "Groceries",  "Price": 2.00}
```

. The "Pretty" radio button is selected.
- Status:** The status bar at the bottom right indicates "Status: 201 Created".

PUT: api/v2/products/{id}

Link: <http://localhost:60125/api/v2/products/2>

Change product details based on id using Put. (Available Id between 1 to 3)

The screenshot displays a REST client interface with a PUT request to `http://localhost:60125/api/v2/products/2`. The request body is a JSON object: `{ "Name": "Car", "Category": "Toy", "Price": 100.00 }`. The response status is `200 OK`, and the response body is a JSON object: `{ "Id": 2, "Name": "Car", "Category": "Toy", "Price": 100.00 }`.

PUT `http://localhost:60125/api/v2/products/2`

Params Authorization Headers (9) Body ● Pre-request!

● none ● form-data ● x-www-form-urlencoded ● raw ● t

```
1 {  
2   "Name": "Car",  
3   "Category": "Toy",  
4   "Price": 100.00  
5 }
```

Body Cookies Headers (11) Test Results Status: 200 OK

Pretty Raw Preview Visualize BETA JSON ▼ ↻

```
1 {  
2   "Id": 2,  
3   "Name": "Car",  
4   "Category": "Toy",  
5   "Price": 100.00  
6 }
```

Providing invalid id will lead to 400 Bad Request

The screenshot shows a REST client interface with the following details:

- Method:** PUT
- URL:** http://localhost:60125/api/v2/products/5
- Body:** A JSON object:

```
{  "Name": "Car",  "Category": "Toy",  "Price": 100.00}
```
- Status:** 400 Bad Request
- Response Body:** No Product Id Found: 5

DELETE: api/v2/products/{id}

Link: <http://localhost:60125/api/v2/products/2>

Remove product based on id using Delete. (Available Id between 1 to 3)

DELETE

http://localhost:60125/api/v2/products/2

Params

Authorization

Headers (9)

Body

Pre-request

Query Params

KEY	VALUE
Key	Value

Body

Cookies

Headers (10)

Test Results

Status: 200 OK

Pretty

Raw

Preview

Visualize BETA

JSON

1

[

2

{

3

"Id": 1,

4

"Name": "Tomato soup",

5

"Category": "Groceries",

6

"Price": 1.39

7

},

8

{

9

"Id": 3,

10

"Name": "Hammer",

11

"Category": "Hardware",

12

"Price": 16.99

13

}

14

]

Providing invalid id will lead to 400 Bad Request

DELETE

http://localhost:60125/api/v2/products/5

Params

Authorization

Headers (9)

Body ●

Pre-request S

Query Params

	KEY	VALUE
	Key	Value

Body

Cookies

Headers (10)

Test Results

Status: 400 Bad Request

Pretty

Raw

Preview

Visualize BETA

Text ▼

1

No Product Id Found: 5

GET: api/v3/products

Link: <http://localhost:60125/api/v3/products>

Get all products.

The screenshot shows a REST client interface with the following components:

- Method and URL:** GET `http://localhost:60125/api/v3/products`
- Tabs:** Params, Authorization, Headers (9), Body, Pre-request.
- Query Params Table:**

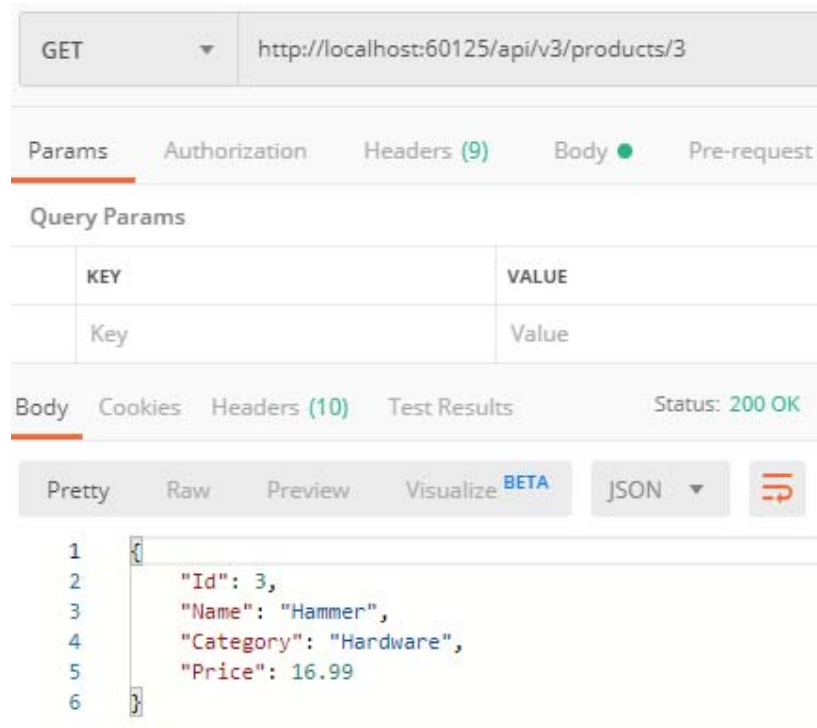
KEY	VALUE
Key	Value
- Response Status:** Status: 200 OK
- Response Format:** Pretty, Raw, Preview, Visualize BETA, JSON, and a refresh icon.
- Response Body (JSON):**

```
[
  {
    "Id": 1,
    "Name": "Tomato soup",
    "Category": "Groceries",
    "Price": 1.39
  },
  {
    "Id": 2,
    "Name": "Yo-yo",
    "Category": "Toys",
    "Price": 3.75
  },
  {
    "Id": 3,
    "Name": "Hammer",
    "Category": "Hardware",
    "Price": 16.99
  }
]
```

GET: api/v3/products/{id}

Link: <http://localhost:60125/api/v3/products/3>

Get product based on id. (Available Id between 1 to 3)




GET <http://localhost:60125/api/v3/products/3>

Params Authorization Headers (9) Body ● Pre-request

Query Params

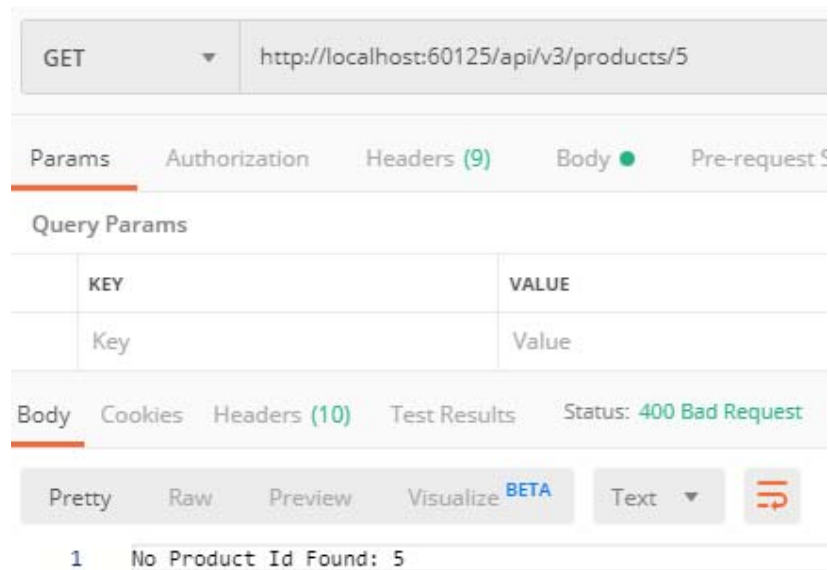
KEY	VALUE
Key	Value

Body Cookies Headers (10) Test Results Status: 200 OK

Pretty Raw Preview Visualize BETA JSON 

```
1 {  
2   "Id": 3,  
3   "Name": "Hammer",  
4   "Category": "Hardware",  
5   "Price": 16.99  
6 }
```

Providing invalid id will lead to 400 Bad Request




GET <http://localhost:60125/api/v3/products/5>

Params Authorization Headers (9) Body ● Pre-request

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (10) Test Results Status: 400 Bad Request

Pretty Raw Preview Visualize BETA Text 

```
1 No Product Id Found: 5
```

GET: api/v3/products?Category=<category>

Link: <http://localhost:60125/api/v3/products?Category=Groceries>

Get products by Category as parameter.

GET ▼ http://localhost:60125/api/v3/products?Category=Groceries

Params ● Authorization Headers (9) Body ● Pre-request Script

Query Params

	KEY	VALUE	DE
<input checked="" type="checkbox"/>	Category	Groceries	
	Key	Value	D

Body Cookies Headers (10) Test Results Status: 200 OK Time: 25ms

Pretty Raw Preview Visualize BETA JSON ▼

```
1  [
2    {
3      "Id": 1,
4      "Name": "Tomato soup",
5      "Category": "Groceries",
6      "Price": 1.39
7    }
8  ]
```

POST: api/v3/products

Link: <http://localhost:60125/api/v3/products>

Create product using Post.

The screenshot displays a REST client interface with the following details:

- Method:** POST
- URL:** http://localhost:60125/api/v3/products
- Body Type:** raw
- Request Body (JSON):**

```
{
  "Name": "ABC",
  "Category": "Toys",
  "Price": 12
}
```
- Status:** 201 Created
- Response Body (JSON):**

```
{
  "Id": 4,
  "Name": "ABC",
  "Category": "Toys",
  "Price": 12.0
}
```


“Under-posting” will lead to 400 Bad Request. (Required: Name, Price*)

**Price must be between 0 to 100*

POST http://localhost:60125/api/v3/products

Params Authorization Headers (9) **Body** Pre-request Script Test

● none ● form-data ● x-www-form-urlencoded ● **raw** ● binary ● G

```
1 {  
2   "Category": "Toys",  
3   "Price": -12  
4 }
```

Body Cookies Headers (10) Test Results Status: **400 Bad Request** Time: 14ms

Pretty Raw Preview Visualize **BETA** JSON ↺

```
1 {  
2   "Message": "The request is invalid.",  
3   "ModelState": {  
4     "item.Name": [  
5       "The Name field is required."  
6     ],  
7     "item.Price": [  
8       "The field Price must be between 0 and 100."  
9     ]  
10  }  
11 }
```

“Over-posting” will lead to 201 Created. (“Over-post” values are excluded in creation)

The screenshot displays a REST client interface. At the top, a POST request is configured for the URL `http://localhost:60125/api/v3/products`. The 'Body' tab is selected, showing a JSON payload:

```
{  "Name": "ABC",  "Category": "Toys",  "Price": 12,  "Discount": 5}
```

. Below the request, the 'Test Results' tab shows the response status as '201 Created'. The response body is displayed in 'Pretty' JSON format:

```
{  "Id": 5,  "Name": "ABC",  "Category": "Toys",  "Price": 12.0}
```

POST `http://localhost:60125/api/v3/products`

Params Authorization Headers (9) **Body** Pre-request S

● none ● form-data ● x-www-form-urlencoded ● **raw** ● b

```
1 {
2   "Name": "ABC",
3   "Category": "Toys",
4   "Price": 12,
5   "Discount": 5
6 }
```

Body Cookies Headers (11) Test Results Status: 201 Created

Pretty Raw Preview Visualize BETA JSON ↵

```
1 {
2   "Id": 5,
3   "Name": "ABC",
4   "Category": "Toys",
5   "Price": 12.0
6 }
```

^ “Discount” is excluded during creation

PUT: api/v3/products/{id}

Link: <http://localhost:60125/api/v3/products/3>

Change product details based on id using Put. (Available Id between 1 to 3)

The screenshot displays a REST client interface with a PUT request to `http://localhost:60125/api/v3/products/3`. The request body is a JSON object: `{ "Name": "Drill", "Category": "Hardware", "Price": 25.00 }`. The response status is `200 OK`, and the response body is a JSON object: `{ "Id": 3, "Name": "Drill", "Category": "Hardware", "Price": 25.00 }`.

PUT `http://localhost:60125/api/v3/products/3`

Params Authorization Headers (9) Body ● Pre-request!

● none ● form-data ● x-www-form-urlencoded ● raw ● t

```
1 {  
2   "Name": "Drill",  
3   "Category": "Hardware",  
4   "Price": 25.00  
5 }
```

Body Cookies Headers (11) Test Results Status: 200 OK

Pretty Raw Preview Visualize BETA JSON ↺

```
1 {  
2   "Id": 3,  
3   "Name": "Drill",  
4   "Category": "Hardware",  
5   "Price": 25.00  
6 }
```

“Under-posting” will lead to 400 Bad Request. (Required: Name, Price*)

**Price must be between 0 to 100*

The screenshot shows a REST client interface with the following details:

- Method:** PUT
- URL:** http://localhost:60125/api/v3/products/3
- Body Type:** JSON
- Request Body:**

```
1 {  
2   "Name": "Drill",  
3   "Category": "Hardware"  
4 }
```
- Status:** 400 Bad Request
- Time:** 11ms
- Size:** 617 B
- Response Body (JSON):**

```
1 {  
2   "Message": "The request is invalid.",  
3   "ModelState": {  
4     "product": [  
5       "Required property 'Price' not found in JSON. Path '', line 4, position 1."  
6     ],  
7     "product.Price": [  
8       "The field Price must be between 1 and 100."  
9     ]  
10  }  
11 }
```

“Over-posting” will lead to 200 OK. (“Over-post” values are excluded in creation)

The screenshot shows a REST client interface. At the top, a PUT request is configured for the URL `http://localhost:60125/api/v3/products/3`. The 'Body' tab is selected, showing a JSON payload:

```
{  "Name": "Drill",  "Category": "Hardware",  "Price": 25.00,  "Discount": 10}
```

. Below the request, the 'Test Results' tab shows a successful response with status '200 OK'. The response body is displayed in 'Pretty' JSON format:

```
{  "Id": 3,  "Name": "Drill",  "Category": "Hardware",  "Price": 25.00}
```

^ “Discount” is excluded during creation

Providing invalid id will lead to 400 Bad Request

The screenshot shows a REST client interface with the following details:

- Method:** PUT
- URL:** http://localhost:60125/api/v3/products/5
- Body:** A JSON object:

```
{  "Name": "Drill",  "Category": "Hardware",  "Price": 25.00}
```
- Status:** 400 Bad Request
- Response Body:** No Product Id Found: 5

DELETE: api/v3/products/{id}

Link: <http://localhost:60125/api/v1/products/1>

Remove product based on id using Delete. (Available Id between 1 to 3)

The screenshot shows a REST client interface with the following components:

- Method and URL:** A dropdown menu set to "DELETE" and a text field containing the URL "http://localhost:60125/api/v3/products/1".
- Tabs:** "Params", "Authorization", "Headers (9)", "Body", and "Pre-request!". The "Params" tab is currently selected.
- Query Params Table:** A table with two columns, "KEY" and "VALUE". It contains one row with the text "Key" and "Value" respectively.
- Body Tab:** The "Body" tab is selected, showing a status of "200 OK".
- Response Format:** A dropdown menu set to "JSON" and a "Pretty" button.
- JSON Response:** A pre-formatted JSON array containing two objects. The first object represents a "Yo-yo" toy, and the second represents a "Drill" hardware item.

```
1  [
2    {
3      "Id": 2,
4      "Name": "Yo-yo",
5      "Category": "Toys",
6      "Price": 3.75
7    },
8    {
9      "Id": 3,
10     "Name": "Drill",
11     "Category": "Hardware",
12     "Price": 25.00
13   }
14 ]
```

Providing invalid id will lead to 400 Bad Request

DELETE

http://localhost:60125/api/v3/products/5

Params

Authorization

Headers (9)

Body ●

Pre-request S

Query Params

KEY	VALUE
Key	Value

Body

Cookies

Headers (10)

Test Results

Status: 400 Bad Request

Pretty

Raw

Preview

Visualize BETA

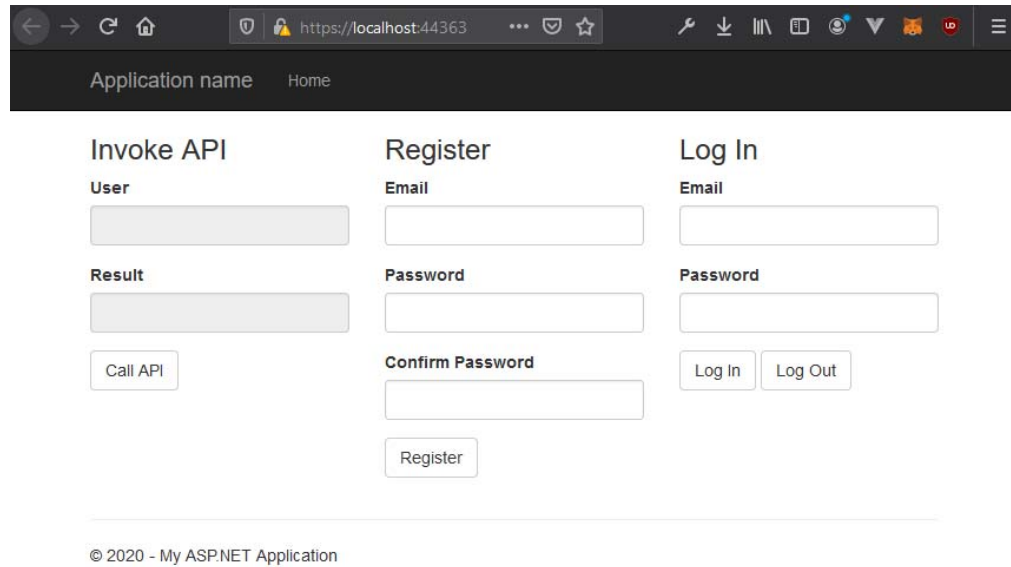
Text

1 No Product Id Found: 5

Task 3

Browser

Debug run the application and you will see the following page on load.



The screenshot shows a web browser window with the address bar displaying `https://localhost:44363`. The browser's address bar and toolbar are visible at the top. Below the browser window, the web application interface is displayed. It features a dark header bar with the text "Application name" and "Home". The main content area is divided into three columns:

- Invoke API:** Contains a "User" label above a text input field, a "Result" label above a text input field, and a "Call API" button.
- Register:** Contains an "Email" label above a text input field, a "Password" label above a text input field, a "Confirm Password" label above a text input field, and a "Register" button.
- Log In:** Contains an "Email" label above a text input field, a "Password" label above a text input field, and two buttons: "Log In" and "Log Out".

At the bottom of the page, there is a footer line that reads "© 2020 - My ASP.NET Application".

Register

Register account by entering email and password twice.

Invoke API

User

Result

Done!

Call API

Register

Email

admin@mail.com

Password

••••••••

Confirm Password

••••••••

Register

Log In

Email

Password

Log In

Log Out

Register can fail if email is already existed in the database.

Invoke API

User

Result

400: Bad Request

Call API

The request is invalid.

Name admin@mail.com is already taken.

Email 'admin@mail.com' is already taken.

Register

Email

admin@mail.com

Password

••••••••

Confirm Password

••••••~•

Register

Log In

Email

Password

Log In

Log Out

Register can fail if password fails the validation process.

Invoke API

User

Result

400: Bad Request

Call API

The request is invalid.

The Password must be at least 6 characters long.

Register

Email

admin@mail.com

Password

••••

Confirm Password

••••

Register

Log In

Email

Password

Log In

Log Out

Invoke API

User

Result

400: Bad Request

Call API

The request is invalid.

Passwords must have at least one non letter or digit character.
Passwords must have at least one lowercase ('a'-'z'). Passwords must have at least one uppercase ('A'-'Z').

Register

Email

admin@mail.com

Password

••••••

Confirm Password

••••••

Register

Log In

Email

Password

Log In

Log Out

Log in

Log in to the account by entering email and password. (Bearer token will be created)

Invoke API

User

admin@mail.com

Result

Call API

Register

Email

Password

Confirm Password

Register

Log In

Email

admin@mail.com

Password

••••••••

Log In

Log Out

Log in can fail if email does not exist or password is incorrect.

Invoke API

User

Result

400: Bad Request

Call API

invalid_grant

The user name or password is incorrect.

Register

Email

Password

Confirm Password

Register

Log In

Email

admin@mail.com

Password

••••••••••••••

Log In

Log Out

Call API

After logging in, user will be able to call API. (Authorized using bearer token)

Invoke API

User

admin@mail.com

Result

Hello, admin@mail.com.

Call API

Register

Email

Password

Confirm Password

Register

Log In

Email

admin@mail.com

Password

••••••••

Log In

Log Out

API could not be called when user is not log on. (Due to lack of bearer token)

Invoke API

User

Result

401: Unauthorized

Call API

Authorization has been denied for this request.

Register

Email

Password

Confirm Password

Register

Log In

Email

Password

Log In

Log Out

Postman

Localhost:

<https://localhost:44363/>

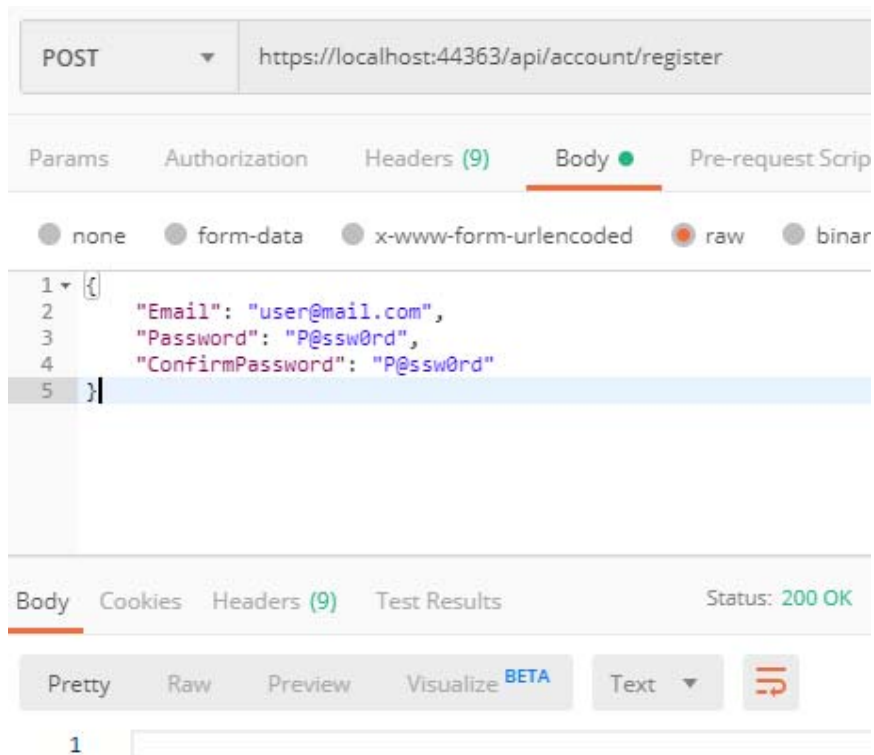
APIs:

Security Level	Type	API Route
Anonymous	POST	api/account/register
Anonymous	POST	token
Authorized	GET	api/values

POST: api/account/register

Link: <https://localhost:44363/api/account/register>

Register account by Post.



Exist email will lead to 400 Bad Request.

The screenshot shows a REST client interface with a POST request to `https://localhost:44363/api/account/register`. The request body is a JSON object with the following fields:

```
{
  "Email": "user@mail.com",
  "Password": "P@ssw0rd",
  "ConfirmPassword": "P@ssw0rd"
}
```

The response status is **400 Bad Request**. The response body is a JSON object indicating an invalid request due to an existing email:

```
{
  "Message": "The request is invalid.",
  "ModelState": {
    "": [
      "Name user@mail.com is already taken.",
      "Email 'user@mail.com' is already taken."
    ]
  }
}
```

The interface includes tabs for Params, Authorization, Headers (9), Body, and Pre-request Script. The Body tab is selected, showing the raw JSON. Below the request body, there are tabs for Body, Cookies, Headers (10), Test Results, and a status indicator. The response body is displayed in a Pretty view, showing the detailed error message.

Fail model validation will lead to 400 Bad Request.

The screenshot displays a REST client interface with the following details:

- Method:** POST
- URL:** https://localhost:44363/api/account/register
- Body Tab:** Selected, showing a JSON request body:

```
1 {  
2   "Email": "user@mail.com",  
3   "Password": "P@ssw0rd",  
4   "ConfirmPassword": "password"  
5 }
```
- Response Tab:** Selected, showing a JSON response body:

```
1 {  
2   "Message": "The request is invalid.",  
3   "ModelState": {  
4     "model.ConfirmPassword": [  
5       "The password and confirmation password do not match."  
6     ]  
7   }  
8 }
```
- Status:** 400 Bad Request
- Time:** 25ms

POST: token

Link: <https://localhost:44363/token>

Get token for API which need authorization by posting email and password. (Email is set as a ClaimType in the token)

The screenshot shows a REST client interface with a POST request to `https://localhost:44363/token`. The request body is `username=user@mail.com&password=P@ssw0rd&grant_type=password`. The response is a JSON object with the following fields:

```
1 {
2   "access_token":
3     "O7DjgAWaNG-Wi69N014dJzMKVKQtIV0aoD9EtVB1-G1HV4B8qY3ynsMpRL2Pf4J
4     JQrBr_0OGZo7b1pAB5HstF_1ykkKf8i7z5GC941XbYEvn0NC4uIYF1pJU49Xjp5y
5     PD4nabbgo1_SFf4T-tsDncYwkf2X0N1VxeFBeRqS_OUQYFtRwpfgI4VvSz5moMMa
6     awjP916qPpQaTEi1PqEj5Ac8veRoRXVvJdb9Ajq4Tq1I3sgxN34eV5xLswfUKDs9
7     hLvAIqyEhvgH70gKJxofv5k9J-fVnVLRccKZuG8WFyH9g2OoLYQk1oe5AUaZ23Sj
8     EgwGIId-DchZ52de9gOaP89AraVyFYNUc2BepfqHgU",
9   "token_type": "bearer",
10  "expires_in": 1209599,
11  "userName": "user@mail.com",
12  ".issued": "Fri, 03 Jan 2020 16:54:56 GMT",
13  ".expires": "Fri, 17 Jan 2020 16:54:56 GMT"
```

Email that does not exist or incorrect password will lead to 400 Bad Request.

POST ▼ https://localhost:44363/token

Params Authorization Headers (10) **Body** ● Pre-request Script Tests

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ Graph

1 username=user@mail.com&password=P@ssw0rd&grant_type=password

Body Cookies (1) Headers (9) Test Results Status: 400 Bad Request Time: 27ms

Pretty Raw Preview Visualize BETA JSON ▼

```
1 {
2   "error": "invalid_grant",
3   "error_description": "The user name or password is incorrect."
4 }
```

GET: api/values

Link: <https://localhost:44363/api/values>

(Requires Bearer token)

Get request with bearer token. (Email is derived from a ClaimType in the token)

GET ▼ https://localhost:44363/api/values Send ▼ Save

Params Authorization Headers (11) Body Pre-request Script Tests Settings Cookies

TYPE

Bearer Token ▼

The authorization header will be automatically generated when you send the request.
[Learn more about authorization](#)

Preview Request

Token XfixyFk3GBS6WjRWdT9ue9ls_-fD6ei0Rp-l-55Jpxi...

Body Cookies (1) Headers (10) Test Results Status: 200 OK Time: 9ms Size: 435 B Save Response

Pretty Raw Preview Visualize BETA JSON ▼ ≡

1 "Hello, user@mail.com."

Get request without bearer token will lead to 403 Unauthorized .

GET ▼ https://localhost:44363/api/values Send ▼ Save

Params **Authorization** Headers (10) Body ● Pre-request Script Tests Settings Cookies

TYPE

Inherit auth from pa... ▼

The authorization header will be automatically generated when you send the request.
[Learn more about authorization](#)

This request is not inheriting any authorization helper at the moment. Save it in a collection to use the parent's authorization helper.

Body Cookies (1) Headers (11) Test Results Status: 401 Unauthorized Time: 8ms Size: 509 B Save Response

Pretty Raw Preview Visualize **BETA** JSON ▼

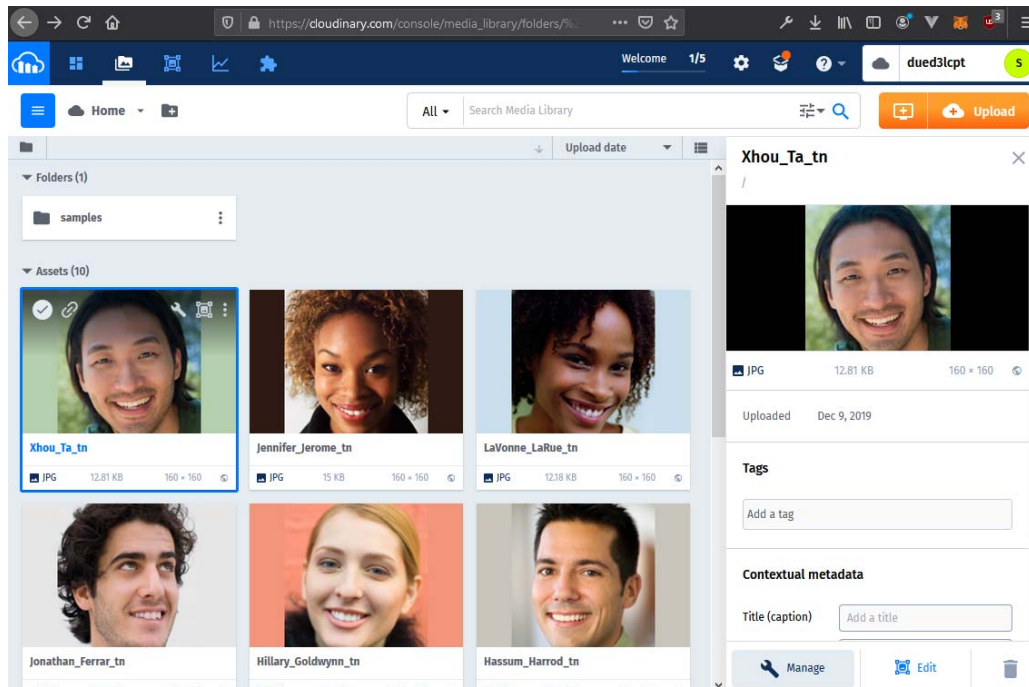
```
1 {  
2   "Message": "Authorization has been denied for this request."  
3 }
```

Task 4

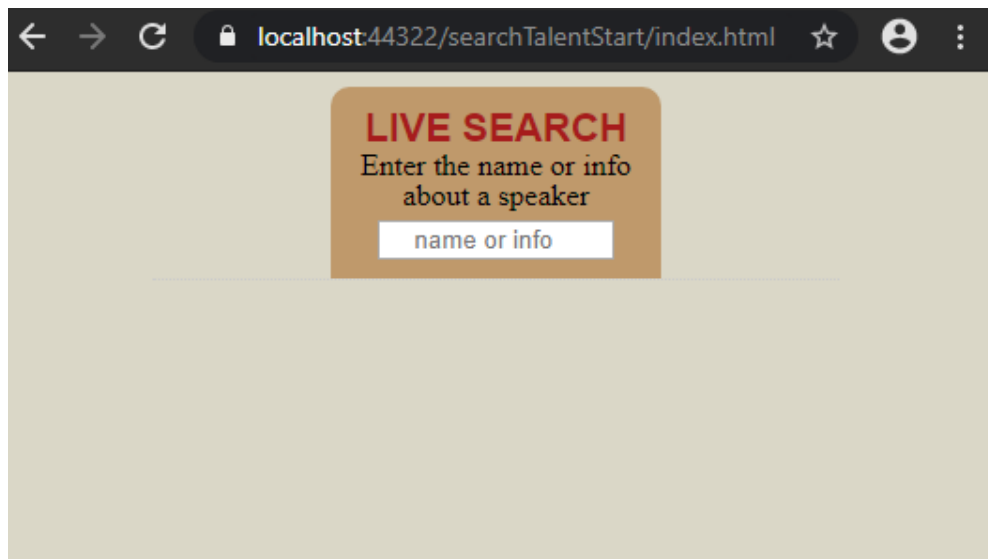
Browser

Assign your link in searchTalentStart/script.js from the Cloudinary media library
<https://cloudinary.com/> .

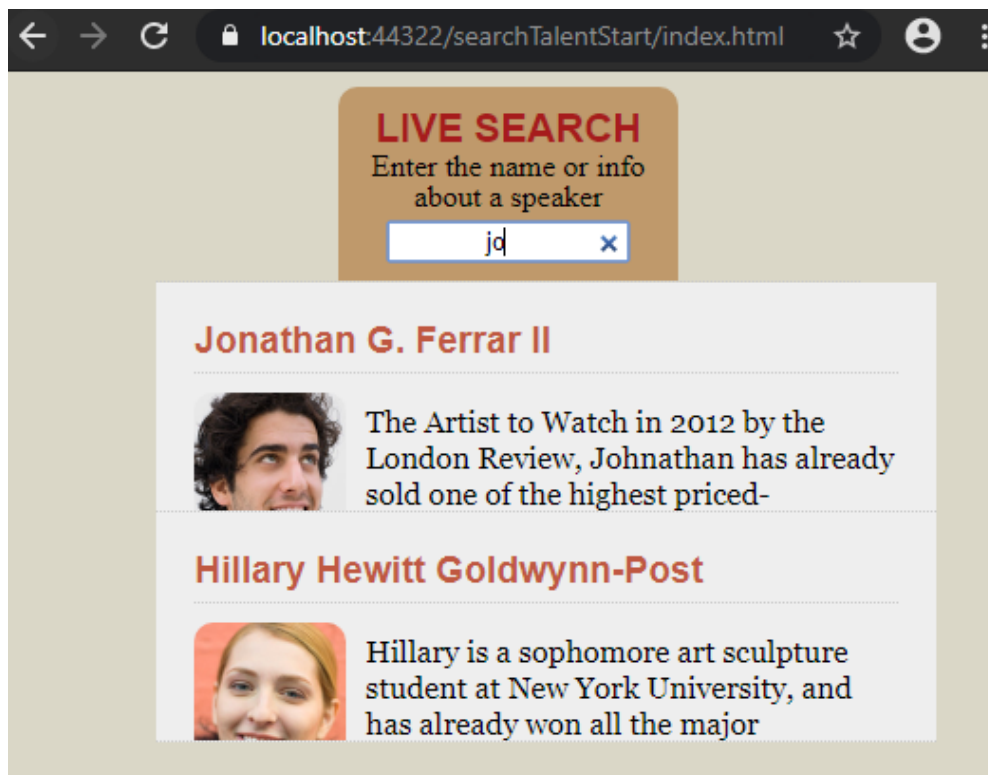
```
//Url for the Cloud image hosting  
var urlForCloudImage = "https://res.cloudinary.com/XXXXXXX/image/upload/XXXXXXX/";
```



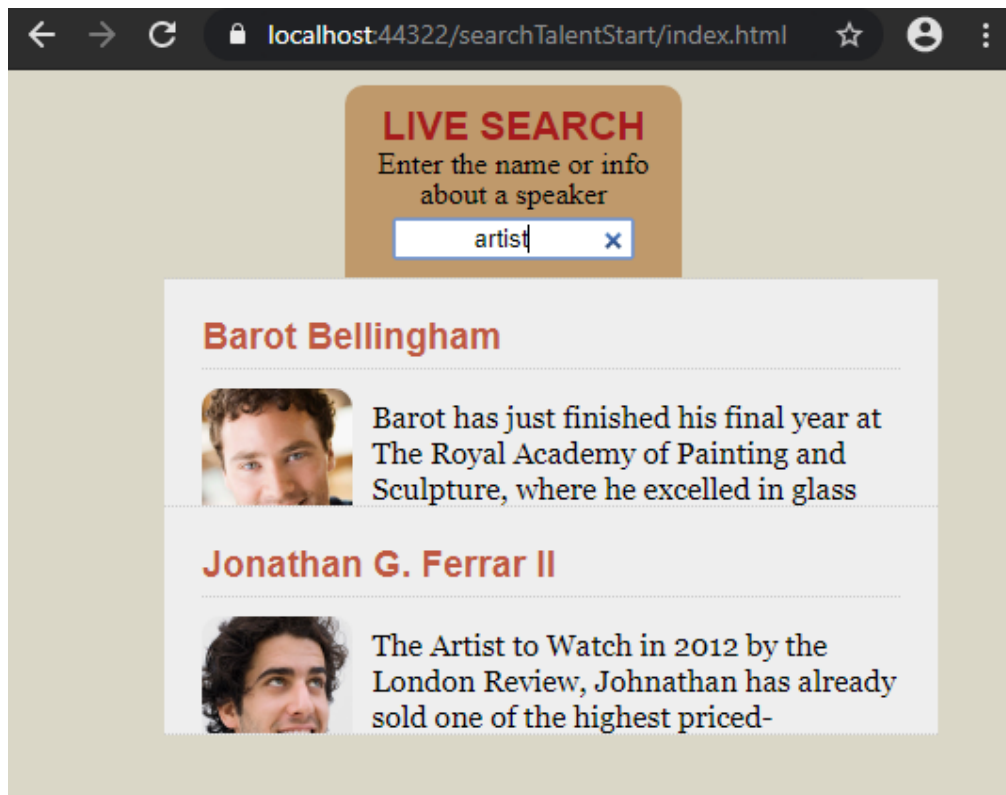
Debug run the application and you will see the following page on load.



You can search by typing the name.



You can also search by typing the info.



In the case of error, client will reconnect and prompt to check.



Postman

Localhost:

<https://localhost:44322/>

Cloud Host:

<https://productstoreweihan.azurewebsites.net>

APIs:

Type	API Route
GET	api/talents
GET	api/talents/{id}

GET: api/talents

Link: <https://localhost:44322/api/talents>

Get all talents

GET

https://localhost:44322/api/talents

Send

Save

ParamsAuthorizationHeaders (10)Body ●Pre-request ScriptTestsSettingsCookiesC

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk E
	Key	Value	Description		

BodyCookies (1)Headers (10)Test ResultsStatus: 200 OKTime: 13msSize: 2.49 KBSave Response

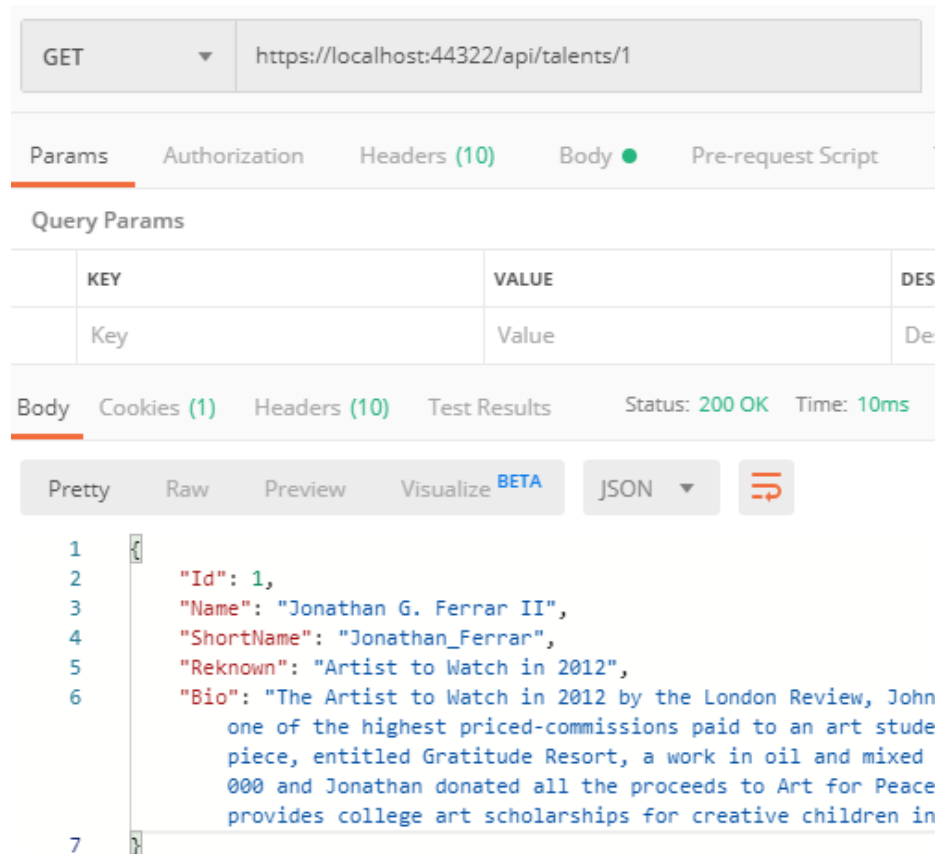
PrettyRawPreviewVisualize BETAJSON

```
1  [
2    {
3      "Id": 0,
4      "Name": "Barot Bellingham",
5      "ShortName": "Barot_Bellingham",
6      "Reknown": "Royal Academy of Painting and Sculpture",
7      "Bio": "Barot has just finished his final year at The Royal Academy of Painting and
8             Sculpture, where he excelled in glass etching paintings and portraiture. Hailed
9             as one of the most diverse artists of his generation, Barot is equally as
10            skilled with watercolors as he is with oils, and is just as well-balanced in
11            different subject areas. Barot's collection entitled \"The Un-Collection\" will
12            adorn the walls of Gilbert Hall, depicting his range of skills and
13            sensibilities - all of them, uniquely Barot, yet undeniably different"
14    },
15    {
16      "Id": 1,
17      "Name": "Jonathan G. Ferrar II",
18      "ShortName": "Jonathan_Ferrar",
19      "Reknown": "Artist to Watch in 2012",
20      "Bio": "The Artist to Watch in 2012 by the London Review, Johnathan has already
21            sold one of the highest priced-commissions paid to an art student, ever on
22            record. The piece, entitled Gratitude Resort, a work in oil and mixed media,
23            was sold for $750,000 and Jonathan donated all the proceeds to Art for Peace,
24            an organization that provides college art scholarships for creative children in
25            developing nations"
26    },
27    {
28      "Id": 2,
29      "Name": "Hillary Hewitt Goldwynn-Post",
30      "ShortName": "Hillary_Goldwynn",
31      "Reknown": "New York University",
32      "Bio": "Hillary is a sophomore art sculpture student at New York University, and
33            has already won all the major international prizes for new sculptors, including
34            the Divinity Circle, the International Sculptor's Medal, and the Academy of
35            Paris Award. Hillary's CAC exhibit features 25 abstract watercolor paintings
36            that contain only water images including waves, deep sea, and river."
37    },
38    {
39      "Id": 3,
40      "Name": "Hassum Harrod",
41      "ShortName": "Hassum_Harrod",
42      "Reknown": "Art College in New Dehli",
43      "Bio": "The Art College in New Dehli has sponsored Hassum on scholarship for his
44            entire undergraduate career at the university, seeing great promise in his
45            contemporary paintings of landscapes - that use equal parts muted and vibrant
46            tones, and are almost a contradiction in art. Hassum will be speaking on \"The
47            use and absence of color in modern art\" during Thursday's agenda."
48    }
49  ]
```

GET: api/talents/{id}

Link: <https://localhost:44322/api/talents/1>

Get talent based on Id. (Available Id between 0 to 3)




GET ▼ https://localhost:44322/api/talents/1

Params Authorization Headers (10) Body ● Pre-request Script

Query Params

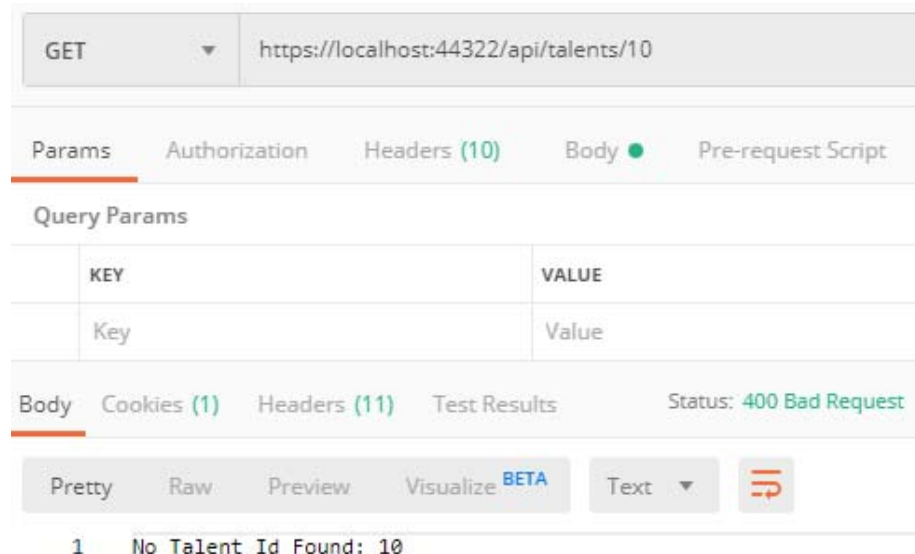
KEY	VALUE	DES
Key	Value	De

Body Cookies (1) Headers (10) Test Results Status: 200 OK Time: 10ms

Pretty Raw Preview Visualize BETA JSON ▼ 

```
1 {  
2   "Id": 1,  
3   "Name": "Jonathan G. Ferrar II",  
4   "ShortName": "Jonathan_Ferrar",  
5   "Reknown": "Artist to Watch in 2012",  
6   "Bio": "The Artist to Watch in 2012 by the London Review, John  
       one of the highest priced-commissions paid to an art stude  
       piece, entitled Gratitude Resort, a work in oil and mixed  
       000 and Jonathan donated all the proceeds to Art for Peace  
       provides college art scholarships for creative children in  
7 }
```

Providing invalid id will lead to 400 Bad Request




GET ▼ https://localhost:44322/api/talents/10

Params Authorization Headers (10) Body ● Pre-request Script

Query Params

KEY	VALUE
Key	Value

Body Cookies (1) Headers (11) Test Results Status: 400 Bad Request

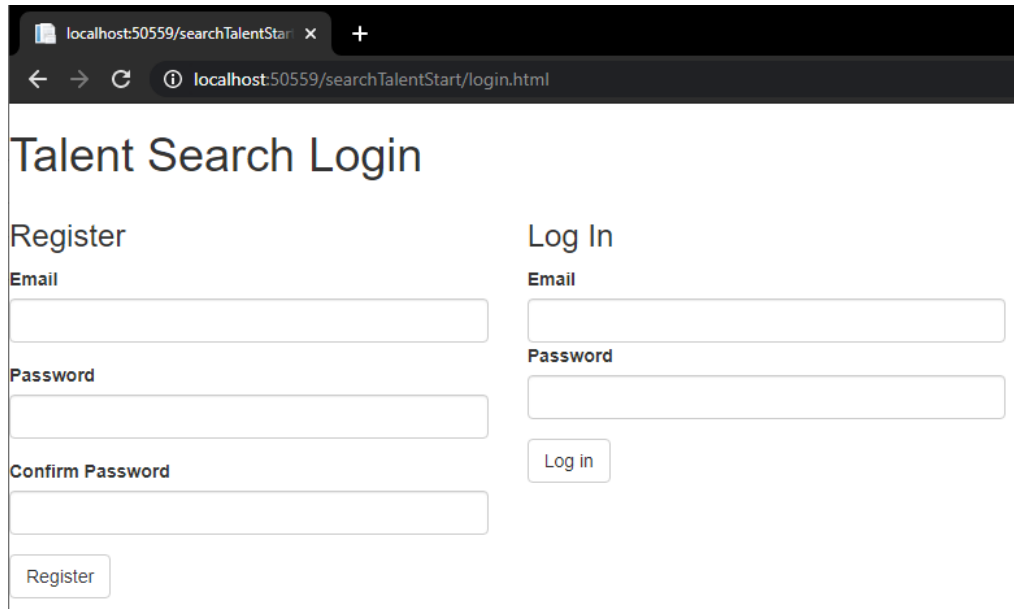
Pretty Raw Preview Visualize BETA Text ▼ 

```
1 No Talent Id Found: 10
```

Task 5

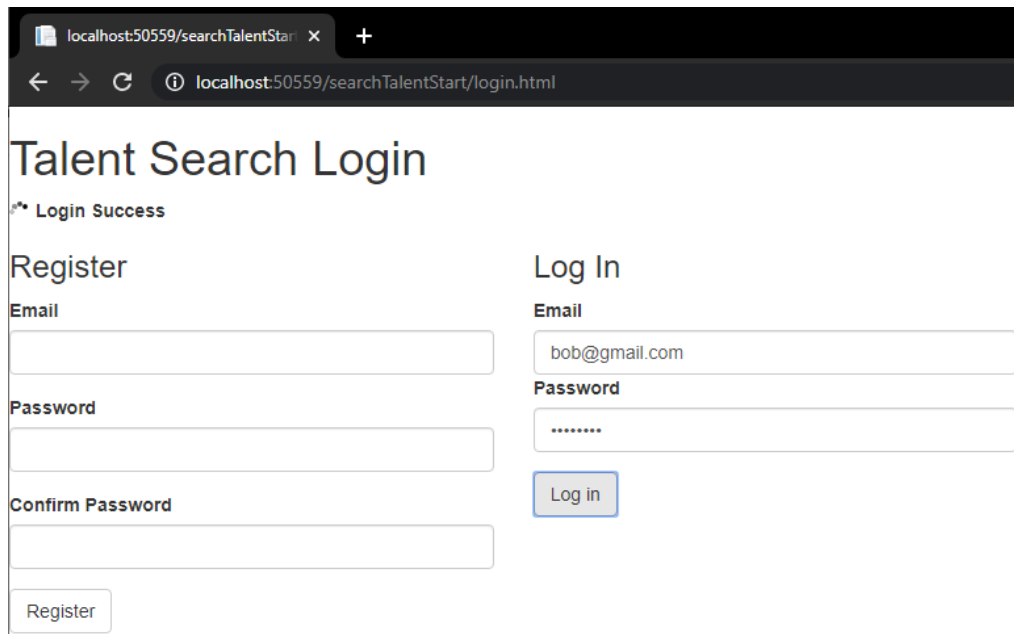
Browser

Debug run the application and you will see the following page on load.



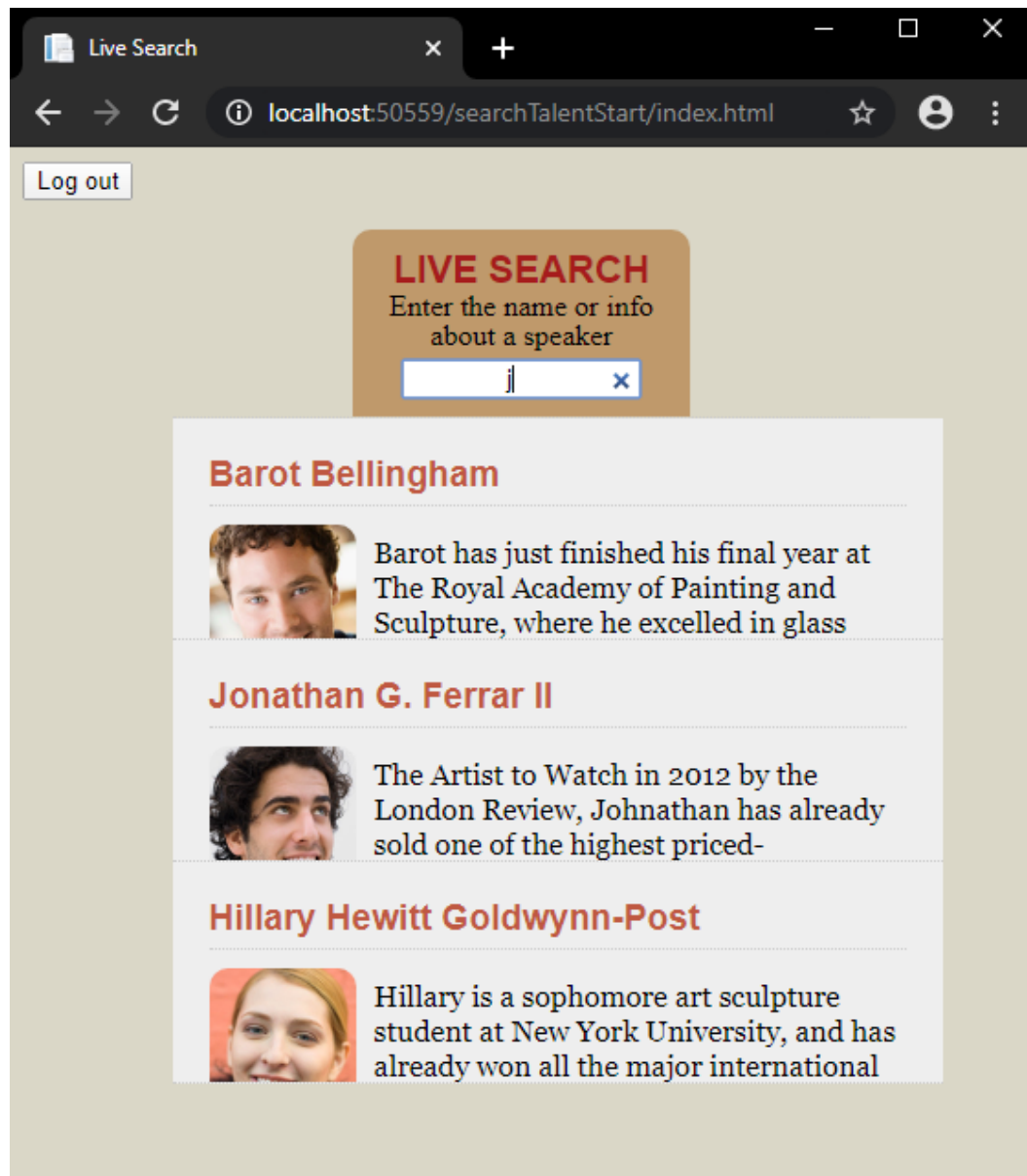
The screenshot shows a web browser window with the address bar displaying 'localhost:50559/searchTalentStart/login.html'. The page title is 'Talent Search Login'. It features two main sections: 'Register' and 'Log In'. The 'Register' section has three input fields labeled 'Email', 'Password', and 'Confirm Password', followed by a 'Register' button. The 'Log In' section has two input fields labeled 'Email' and 'Password', followed by a 'Log in' button.

Create an account or login. (Email: bob@gmail.com, Pwd: P@ssw0rd)



The screenshot shows the same web browser window, but now with a 'Login Success' message at the top left. The 'Log In' section's 'Email' field is populated with 'bob@gmail.com' and the 'Password' field is filled with dots. The 'Log in' button is highlighted in blue. The 'Register' section remains unchanged with empty input fields and the 'Register' button.

Success login will redirect to talent search page. (Token will be stored in localStorage)



Using the browser console, we can see that https and Token are used for data in transit.

Status	Method	Domain	File	Cause	Type	Transferred	Size
302	GET	localhost:50559	talents	xhr	json	2.53 KB	2.09 KB
200	OPTIONS	localhost:44398	talents	xhr	plain	476 B	0 B
200	GET	localhost:44398	talents	xhr	json	2.54 KB	2.09 KB

3 requests

4.17 KB / 5.53 KB transferred

Finish: 75 ms

Headers

Request URL: http://localhost:50559/api/talents

Request Method: GET

Remote Address: [::1]:50559

Status Code: 302 Found

Version: HTTP/1.1

Referrer Policy: no-referrer-when-downgrade

Edit and Resend

Status	Method	Domain	File	Cause	Type	Transferred	Size
302	GET	localhost:50559	talents	xhr	json	2.53 KB	2.09 KB
200	OPTIONS	localhost:44398	talents	xhr	plain	476 B	0 B
200	GET	localhost:44398	talents	xhr	json	2.54 KB	2.09 KB

3 requests

4.17 KB / 5.53 KB transferred

Finish: 75 ms

Headers

Request URL: https://localhost:44398/api/talents

Request Method: GET

Remote Address: [::1]:44398

Status Code: 200 OK

Version: HTTP/2

Referrer Policy: no-referrer-when-downgrade

Edit and Resend

Filter Headers

Response Headers (464 B)

Request Headers (901 B)

Accept: */*

Accept-Encoding: gzip, deflate, br

Accept-Language: en-US,en;q=0.5

Authorization: Bearer EqoXE_7wqqYQqRCVSj-Fs0N...GIkj1bIIS575Z4uYPk1I3TsP9z9aE

Connection: keep-alive

DNT: 1

Host: localhost:44398

Origin: http://localhost:50559

Referer: http://localhost:50559/searchTalentStart/index.html

TE: Trailers

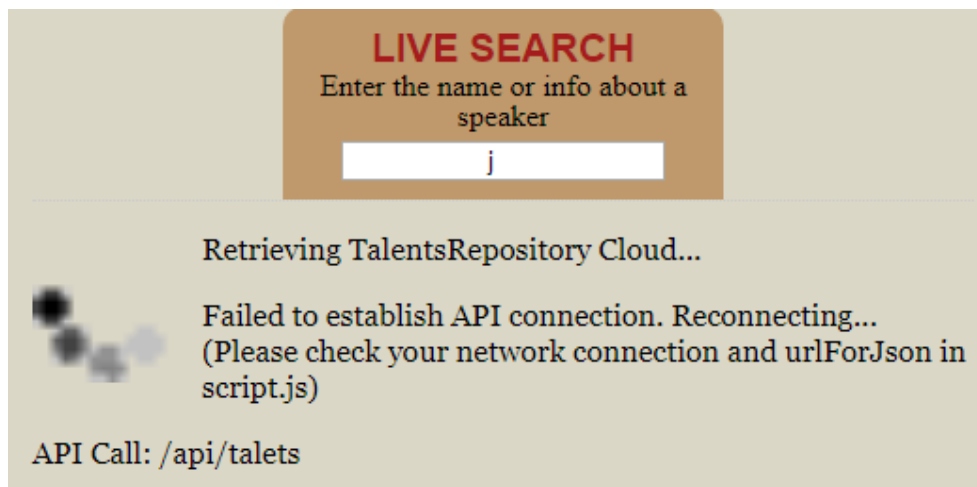
User-Agent: Mozilla/5.0 (Windows NT 10.0; ...) Gecko/20100101 Firefox/73.0

X-Requested-With: XMLHttpRequest

Without proper login, search will be unauthorized.



In the case of error, client will reconnect and prompt to check.



Insomnia (Postman)

Localhost:

<http://localhost:50559/>

<https://localhost:44398/>


APIs:


Type	API Route
GET	api/talents
GET	api/talents/{id}


GET: api/talents

Link: <http://localhost:50559/api/talents>

Get all talents

GET  http://localhost:50559/api/talents Send

Body 


Bearer 


Query


Header


Docs

TOKEN SNqKO2C_qUQck-y3JVPlsszkcCIHy218dYhno0yN7CW481i8CWkl87K

PREFIX 

ENABLED 

200 OK 15 ms 2.1 KB 2 Minutes Ago 

Preview 

Header 10 Cookie Timeline

```
1 [
2   {
3     "Id": 0,
4     "Name": "Barot Bellingham",
5     "ShortName": "Barot_Bellingham",
6     "Reknown": "Royal Academy of Painting and Sculpture",
7     "Bio": "Barot has just finished his final year at The Royal Academy
of Painting and Sculpture, where he excelled in glass etching paintings
and portraiture. Hailed as one of the most diverse artists of his
generation, Barot is equally as skilled with watercolors as he is with
oils, and is just as well-balanced in different subject areas. Barot's
collection entitled \"The Un-Collection\" will adorn the walls of Gilbert
Hall, depicting his range of skills and sensibilities - all of them,
uniquely Barot, yet undeniably different"
8   },
```

- The screenshot displays the Insomnia REST client interface. At the top, the request method is GET and the URL is http://localhost:50559/api/talents. The authentication is set to Bearer with the token SNqKO2C_qUQCk-y3JVP1sszkcCIHy218dYhno0yN7CW481i8CWki87K. The request is enabled. The response status is 200 OK, with a response time of 15 ms and a size of 2.1 KB. The response was received 4 minutes ago. The 'Preview' tab is active, showing the response body as HTML. The response starts with < HTTP/1.1 302 Found, indicating a redirect. The 'Location' header points to https://localhost:44398/api/talents. The response is served by Microsoft-IIS/10.0 on an ASP.NET 4.0.30319 server.

```
< HTTP/1.1 302 Found
< Cache-Control: no-cache
< Pragma: no-cache
< Content-Length: 32
< Content-Type: text/html; charset=utf-8
< Expires: -1
< Location: https://localhost:44398/api/talents
< Server: Microsoft-IIS/10.0
< X-AspNet-Version: 4.0.30319
< X-SourceFiles: =?UTF-8?B?
QzpcVXNlcnNcQWRtaW5cRG9jdW11bnRzXEdpdEh1Y1x0U0NfQXNzawdubWVudF8xUUYXNjVxQm9k
dWw0U3RvcmlvZjJcUHJvZHVjdFNoZjJlXGFwaVx0YXN1bnRz?=.
< X-Powered-By: ASP.NET
< Date: Wed, 08 Jan 2020 05:51:37 GMT
```


2. Transported to <https://localhost:44398> with response 200 OK.

The screenshot displays the Swagger UI interface for a REST API. At the top, the method is **GET** and the URL is `http://localhost:50559/api/talents`. The **Body** tab is selected, showing a **Bearer** token: `SNqKO2C_qUQck-y3JVP1sszkcCIHy218dYhno0yN7CW481i8CWki87K`. The **PREFIX** is empty, and **ENABLED** is checked. Below this, the response status is **200 OK** with a response time of **15 ms** and a size of **2.1 KB**, dated **6 Minutes Ago**. The **Preview** tab is active, showing the raw response text. The request log indicates a successful **GET /api/talents HTTP/1.1** with a status of **200 OK**. The response headers include `Cache-Control: no-cache`, `Pragma: no-cache`, `Content-Type: application/json; charset=utf-8`, `Expires: -1`, `Server: Microsoft-IIS/10.0`, `X-AspNet-Version: 4.0.30319`, `X-SourceFiles: =?UTF-8?B?QZpcVXNlcnNcQWRtaW5rRG9jdW11bnRzXEdpdEh1Y1x0U0NfQXNzaWduYVdFbXVhYXNlbnRzXQcm9kdWNoU3RvcmlvVjJcUHJvZHVjdFNoZ3JlXGFwaVx0YWx1bnRz?=>`, `X-Powered-By: ASP.NET`, `Date: Wed, 08 Jan 2020 05:51:37 GMT`, and `Content-Length: 2136`.

Without proper token, request will be 401 Unauthorized.

GET ▼ http://localhost:50559/api/talents Send

Body ▼

Bearer ▼

Query

Header

Docs

TOKEN

PREFIX ⓘ

ENABLED ☒

401 Unauthorized 31 ms 61 B Just Now ▼

Preview ▼

Header 11

Cookie

Timeline

```
1 {
2   "Message": "Authorization has been denied for this request."
3 }
```

GET: api/talents/{id}

Link: <http://localhost:50559/api/talents/2>

Get talent using http will result a redirect to https. (Available Id between 0 to 3)

GET

http://localhost:50559/api/talents/2

Send

Body

Bearer

Query

Header

Docs

TOKEN

SNqKO2C_qUQck-y3JVPlsszkcCIHy218dYhno0yN7CW481i8CWkl87K

PREFIX

ENABLED

☒

200 OK

0 ms

492 B

Just Now

Preview

Header

Cookie

Timeline

1 {

2 "Id": 2,

3 "Name": "Hillary Hewitt Goldwynn-Post",


4 "ShortName": "Hillary_Goldwynn",


5 "Reknown": "New York University",


6 "Bio": "Hillary is a sophomore art sculpture student at New York University, and has already won all the major international prizes for new sculptors, including the Divinity Circle, the International Sculptor's Medal, and the Academy of Paris Award. Hillary's CAC exhibit features 25 abstract watercolor paintings that contain only water images including waves, deep sea, and river."

7 }

1. Header shows transport to <https://localhost:44398/> with response 302 Found

GET  http://localhost:50559/api/talents/2 Send

Body 

Bearer 


Query

Header

Docs

TOKEN

SNqKO2C_qUQck-y3JVPlsszkCIHy218dYhno0yN7CW481i8CWki87K

PREFIX 


ENABLED


☒

200 OK

0 ms

492 B

Just Now 

Preview 

Header 10

Cookie

Timeline


> GET /api/talents/2 HTTP/1.1

> Host: localhost:50559

> User-Agent: insomnia/7.0.6

> Authorization: Bearer SNqKO2C_qUQck-y3JVPlsszkCIHy218dYhno0yN7CW481i8CWki87K
iabqjEqGt14eKm37jwOhNKqs6qNU1I7LMY8LcRza8W7iL1pid4x08PSAse-T80WyyDgcIDZQK3vkMbUCndGuyKo588cfE7eMqhw10zxXP1PGQDdBEZIXgSaR37i8QpXLhumR6j9CIPoR2CVR10RKe7ImPf4K-XniRJm6rJN-Qrra6VaxrWKDUNTK38Vocw1T7_sruUdjLCLA2xHJf3hdvTY08GMS1st_8bZDeD48PtCizfleaqyLueWfQ4aA1dNn5gX81MpfMpDZ2k-cVhhKMeP1B3pKLNutMF7Ux8f6NSloLUzPotaglv080AY_bJU2srB7i_Zkms1M31o51ROhRlQhFwym6KLI9fjEo3MZlIpSSczFzGTVP353rxAJqvHCNw6_wy0UFFu0bxC08YXcxcmh-0mBNPGETPjK0TPkiU8K7iUTNDY

> Accept: */*

< HTTP/1.1 302 Found 


< Cache-Control: no-cache

< Pragma: no-cache

< Content-Length: 32

< Content-Type: text/html; charset=utf-8

< Expires: -1

< Location: <https://localhost:44398/api/talents/2> 

< Server: Microsoft-IIS/10.0

< X-AspNet-Version: 4.0.30319

< X-SourceFiles: =?UTF-8?B?QZpcVXNlcnNcQWRtaW5cRG9jdW1lbnRzXEdpdEh1Y1x0U0NfQXNzaWdubWVudFhUYXNrNVxQcm9kdWN0U3RvcnVfVjJcUHJvZHVjdFN0b3JlXGFwaVx0YXxlb3RzXDI=?=

< X-Powered-By: ASP.NET

< Date: Wed, 08 Jan 2020 06:23:41 GMT

- A screenshot of a REST client interface. At the top, the method is 'GET' and the URL is 'http://localhost:50559/api/talents/2'. Below this, there are tabs for 'Body', 'Bearer', 'Query', 'Header', and 'Docs'. The 'Bearer' tab is selected, showing a 'TOKEN' field with the value 'SNqKO2C_qUQck-y3JVPlsszkcCIHy218dYhno0yN7CW481i8CWki87K'. Below the token, there is a 'PREFIX' field with a question mark icon and an 'ENABLED' checkbox which is checked. At the bottom, there is a status bar showing '200 OK', '0 ms', and '492 B'. On the far right, it says 'A Minute Ago' with a dropdown arrow.

```

> GET /api/talents/2 HTTP/1.1
> Host: localhost:44398
> User-Agent: insomnia/7.0.6
> Authorization: Bearer SNqK02C_qUQcK-
y3JVP1sszkcCIHy218dyhno0yN7CW481i8CwkI87KiabqjEqGt14eKm37jWOhNKqs6qNU1I7LMY8
LcRzaBW7iL1pid4x08PsAse-
T80myyDgcIDZQK3vkMBUCndGUyKo588cfE7eMqhw10zxXP1PGQDdBZIXg5aR37i8QpXLhumR6j9
CIPoR2CVR10RKe7Impf4K-XniRJM6rJN-
Qrra6VaxrWKDUNTK38V0cw1T7_sruUdjLCLA2xHJf3hdvTY08BGMS1st_8bZDeD48PtCizfleaayL
ueWfq4aA1dNn5gX81MpfMpDZ2k-
cVhhkMep1B3pKLNuTMF7Ux8f6NSloLUzPotaglv080AY_bJU2srB7i_ZkmslMB3Io51RoHRlqhFWy
m6KlIf9fjEo3MZlIpSSczFzGtVP353rxAJqvHcNw6_wy0UFFu0bxC08YXcxcmh-
0mBNPGETPjKOTPKiU8K7iUTNDY
> Accept: */*

< HTTP/1.1 200 OK
< Cache-Control: no-cache
< Pragma: no-cache
< Content-Length: 492
< Content-Type: application/json; charset=utf-8
< Expires: -1
< Server: Microsoft-IIS/10.0
< X-AspNet-Version: 4.0.30319
< X-SourceFiles: =?UTF-8?B?
QzpcVXNlcnNcQWRtaW5CRG9jdWllbnRzXEEdpdEh1Y1x0U0NfQXNzaWduYVudFhUyXNlbnVxQcm9k
dWV0U3RvcmlvVjJcUHJvZHVjZDfN0b3JlXGFwaVx0YXlbnRzXDI=?=
< X-Powered-By: ASP.NET
< Date: Wed, 08 Jan 2020 06:23:41 GMT

```

Without proper token, request will be 401 Unauthorized.

GET ▼ http://localhost:50559/api/talents/2 Send

Body ▼	Bearer ▼	Query	Header	Docs
<div>TOKEN</div> <div>PREFIX ⓘ</div> <div>ENABLED <input checked="" type="checkbox"/></div>				

401 Unauthorized 15 ms 61 B Just Now ▼

Preview ▼ Header 11 Cookie Timeline

```
1 {  
2   "Message": "Authorization has been denied for this request."  
3 }
```

Invalid Talent id will still redirect to https but with 400 Bad Request

GET ▼ http://localhost:50559/api/talents/10 Send

Body ▼	Bearer ▼	Query	Header	Docs
<div>TOKEN ICNw6_wy0UFFu0bxC08YXcxcmh-0mBNPGETPjKOTPkIU8K7iUTNDY</div> <div>PREFIX ⓘ</div> <div>ENABLED <input checked="" type="checkbox"/></div>				

400 Bad Request 31 ms 22 B Just Now ▼

Preview ▼ Header 10 Cookie Timeline

```
1 No Talent Id Found: 10
```

Task 6

Browser

Insert your publishable key and secret key inside StripeChargeModel.cs from <https://dashboard.stripe.com/test/dashboard> .

▼ Get your test API keys

Publishable key pk_test_XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

Secret key sk_test_XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```
public class StripeChargeModel
{
    //API Testing Keys
    //PKey = publishable key, SKey = secret key
    1 reference | 0 changes | 0 authors, 0 changes
    public string PKey { get; } = "pk_test_XXX";
    0 references | 0 changes | 0 authors, 0 changes
    public static string SKey { get; } = "sk_test_XXX";
}
```

Debug run the application and you will see the following page on load.

← → ↻ ⓘ localhost:7574/Home/Charge

Stripe Project

☰

Stripe Charge Example with Stripe.js

Stripe Charge Example

Card Number *

Cvc *

Expiry Month (MM) *

Expiry Year (YYYY) *

CardHolderName

Amount

AddressLine1

AddressLine2

AddressCity

AddressPostcode

AddressCountry

[ProcessPayment](#)

Fill in the details

(Card Holder Name and Addresses are optional)

(Minimum amount: 0.01)

Stripe Charge Example with Stripe.js

Stripe Charge Example

Card Number *

Cvc *

Expiry Month (MM) *

Expiry Year (YYYY) *

CardHolderName

Amount

AddressLine1

AddressLine2

AddressCity

AddressPostcode

AddressCountry

[ProcessPayment](#)

Stripe will start validating card, create token and charge.
(indicated by the loading icon)

Stripe Charge Example with Stripe.js

Stripe Charge Example

Card Number *

Cvc *

Expiry Month (MM) *

Expiry Year (YYYY) *

CardHolderName

Amount

AddressLine1

AddressLine2

AddressCity

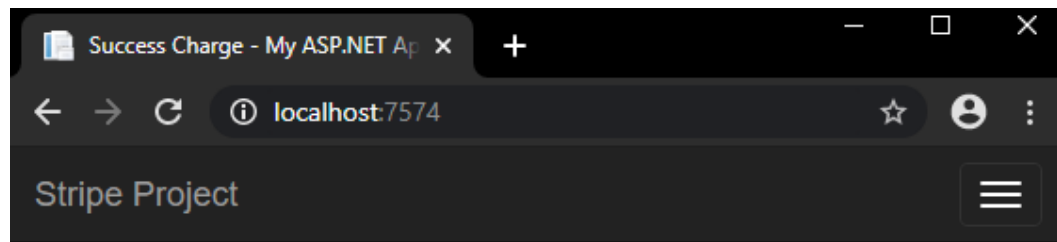
AddressPostcode

AddressCountry

[ProcessPayment](#)



Successful charge will result in successful page with redirect.



Charge Successful! Redirecting back to Stripe Charge Example page...



© 2020 - My ASP.NET Application

Successful charge will be reflected in <https://dashboard.stripe.com/test/payments>.

Payment


TEST DATA

ch_1FxJQkItmXAp9SBazkYD6frI

\$100.00 SGD

Succeeded ✓

↶ Refund...

Date	Customer	Payment method	Risk evaluation
Jan 5, 4:41 AM	None	 **** 5100	14 Normal

Timeline

+ Add note

✓ Payment succeeded

Jan 5, 2020, 4:41 AM


14 Stripe risk evaluation: normal

Jan 5, 2020, 4:41 AM

Payment details

Statement descriptor	Stripe
Amount	\$100.00
Fee	\$3.90 ⓘ
Net	\$96.10
Status	Succeeded
Description	Description for test charge Edit

Payment method

Number	**** 5100	Address	No address
Fingerprint	oqYeuygtSUazlwMn	Origin	United States 
Expires	02 / 2020	CVC check	Passed ✓
Type	Mastercard prepaid card		
ID	card_1FxJQjItmXAp9SBaFmTFWtct		

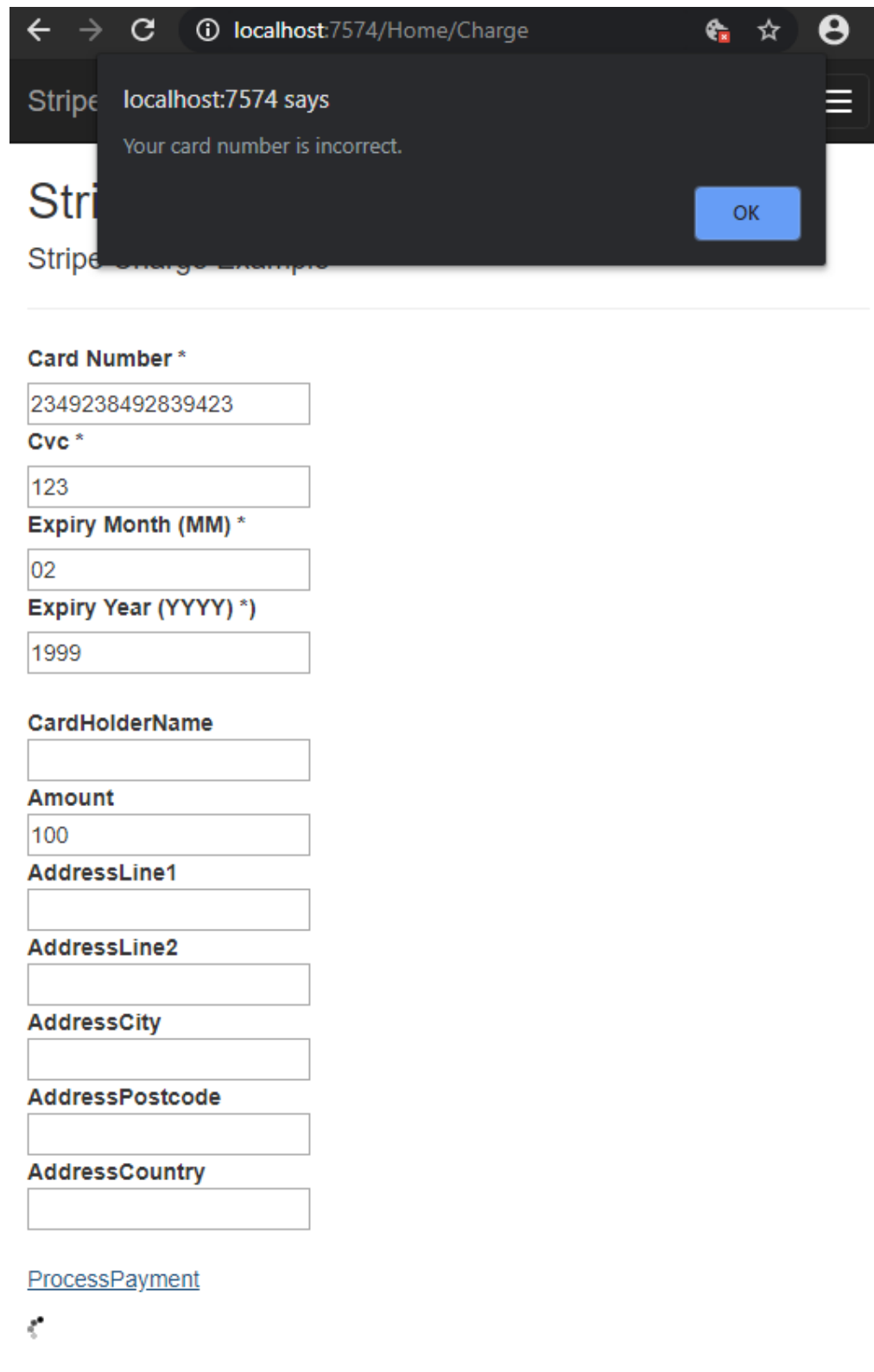
Logs

200 OK	POST /v1/charges	1/5/20, 4:41:06 AM
200 OK	POST /v1/tokens	1/5/20, 4:41:05 AM

Events

A successful payment was made for \$100.00 SGD	1/4/20, 8:41:06 PM
--	--------------------

Error from Stripe if card details are invalid .



The screenshot shows a web browser window with the address bar displaying "localhost:7574/Home/Charge". A dark modal dialog box is overlaid on the page, containing the text "localhost:7574 says" and "Your card number is incorrect." with an "OK" button. Below the dialog, the Stripe payment form is visible. The form includes fields for Card Number (2349238492839423), Cvc (123), Expiry Month (MM) (02), Expiry Year (YYYY) (1999), CardHolderName, Amount (100), AddressLine1, AddressLine2, AddressCity, AddressPostcode, and AddressCountry. A "ProcessPayment" link is at the bottom of the form.

Stripe

localhost:7574 says

Your card number is incorrect.

OK

Stripe

Stripe Charge Example

Card Number *

2349238492839423

Cvc *

123

Expiry Month (MM) *

02

Expiry Year (YYYY) *

1999

CardHolderName

Amount

100

AddressLine1

AddressLine2

AddressCity

AddressPostcode

AddressCountry

[ProcessPayment](#)

Error from Model Validation if amount is invalid.

CardHolderName

Amount

The field Amount must be between 0.01 and ∞ .

AddressLine1

AddressLine2

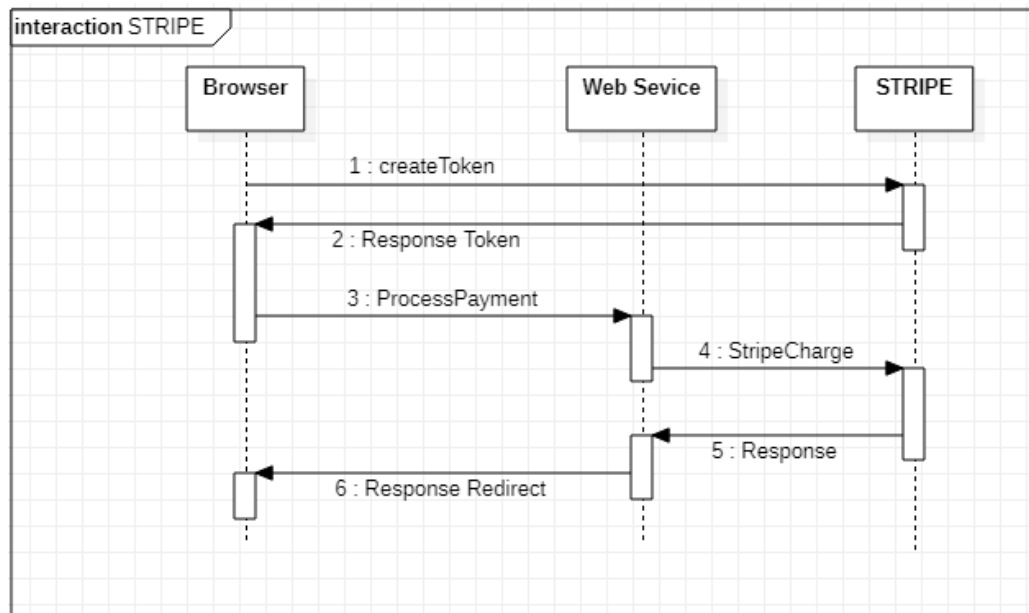
AddressCity

AddressPostcode

AddressCountry

[ProcessPayment](#)

Sequence Diagram



AWS Talents Images

Access: <https://s3-ap-southeast-1.amazonaws.com/talentsweihan>

Images uploaded into AWS S3 Bucket.

aws

Services ▾

Resource Groups ▾

★

🔔

Wong Wei Han ▾

Global ▾

Amazon S3 > talentsweihan

talentsweihan

Overview

Properties

Permissions Public

Management

Access points

🔍 Type a prefix and press Enter to search. Press ESC to clear.

📁 Upload










➕ Create folder

⬇️ Download

⌵ Actions ▾

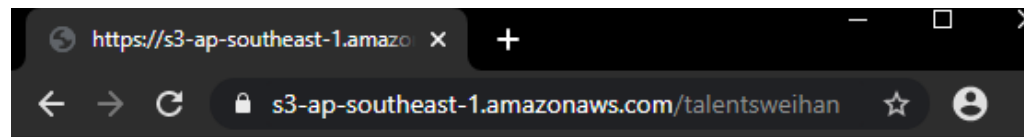
Asia Pacific (Singapore) 🔄

Viewing 1 to 9

<input type="checkbox"/>	Name ▾	Last modified ▾	Size ▾	Storage class ▾
<input type="checkbox"/>	 Barot_Bellingham_tn.jpg	Jan 7, 2020 4:44:48 AM GMT+0800	12.5 KB	Standard
<input type="checkbox"/>	 Constance_Smith_tn.jpg	Jan 7, 2020 4:44:48 AM GMT+0800	12.5 KB	Standard
<input type="checkbox"/>	 Hassum_Harrod_tn.jpg	Jan 7, 2020 4:44:48 AM GMT+0800	11.1 KB	Standard
<input type="checkbox"/>	 Hillary_Goldwynn_tn.jpg	Jan 7, 2020 4:44:48 AM GMT+0800	10.8 KB	Standard
<input type="checkbox"/>	 Jennifer_Jerome_tn.jpg	Jan 7, 2020 4:44:48 AM GMT+0800	15.0 KB	Standard
<input type="checkbox"/>	 Jonathan_Ferrar_tn.jpg	Jan 7, 2020 4:44:48 AM GMT+0800	13.0 KB	Standard
<input type="checkbox"/>	 LaVonne_LaRue_tn.jpg	Jan 7, 2020 4:44:48 AM GMT+0800	12.2 KB	Standard
<input type="checkbox"/>	 Riley_Rewington_tn.jpg	Jan 7, 2020 4:44:48 AM GMT+0800	11.8 KB	Standard
<input type="checkbox"/>	 Xhou_Ta_tn.jpg	Jan 7, 2020 4:44:48 AM GMT+0800	12.8 KB	Standard

Viewing 1 to 9

Public access to S3 Bucket.



This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>talentsweihan</Name>
  <Prefix/>
  <Marker/>
  <MaxKeys>1000</MaxKeys>
  <IsTruncated>>false</IsTruncated>
  ▼<Contents>
    <Key>Barot_Bellingham_tn.jpg</Key>
    <LastModified>2020-01-06T20:44:48.000Z</LastModified>
    <ETag>"2d7605292499f1cf635b2dd7c28982fb"</ETag>
    <Size>12755</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
  ▼<Contents>
    <Key>Constance_Smith_tn.jpg</Key>
    <LastModified>2020-01-06T20:44:48.000Z</LastModified>
    <ETag>"58f2cb1520ceb1bc71107e87ef3a14ab"</ETag>
    <Size>12795</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
  ▼<Contents>
    <Key>Hassum_Harrod_tn.jpg</Key>
    <LastModified>2020-01-06T20:44:48.000Z</LastModified>
    <ETag>"817bd23c23477d4076534ebf956ef4bc"</ETag>
    <Size>11356</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
  ▼<Contents>
    <Key>Hillary_Goldwynn_tn.jpg</Key>
    <LastModified>2020-01-06T20:44:48.000Z</LastModified>
    <ETag>"4de2340b0d4649bc324286b945e13ac9"</ETag>
    <Size>11051</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
  ▼<Contents>
    <Key>Jennifer_Jerome_tn.jpg</Key>
    <LastModified>2020-01-06T20:44:48.000Z</LastModified>
    <ETag>"e1aa96736e3c759d3a9efd519d1fbf3b"</ETag>
    <Size>15361</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
```

Task 7

Browser (ASP.NET)

Insert your Clarifai API key inside ClarifaiModel.cs from <https://portal.clarifai.com/apps>.

API Keys ?

[DETAILS](#)

ClarifaiProject-all-scopes

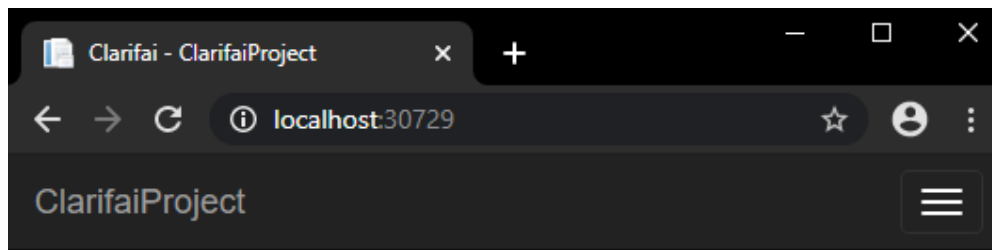
XX

Dec 12, 2019

[CREATE NEW API KEY](#)

```
public class ClarifaiModel
{
    //API key for clarifai project
    0 references | 0 changes | 0 authors, 0 changes | 0 exceptions
    public static string APIKey { get; } = "XXX";
}
```

Debug run the application and you will see the following page on load.



Submit Web image

Submit

Submit Local image

Drag and Drop your Image

Choose File No file chosen

Upload

Image preview

Submit: Web Image

Submit image using web image and wait for tag to load.

ClarifaiProject



Submit Web image

<https://samples.clarifai.com/metro-north.jpg>

Submit

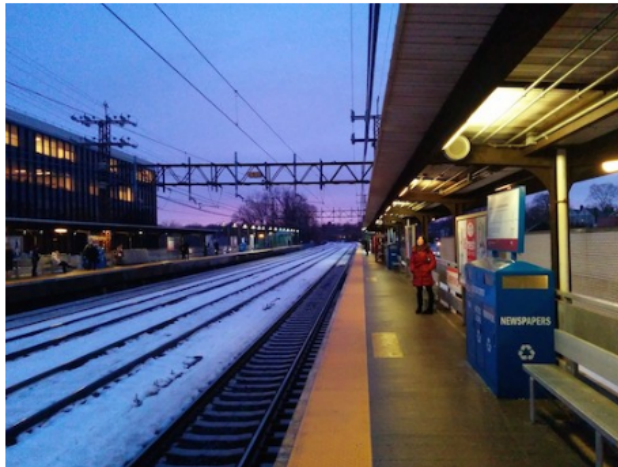
Submit Local image

Drag and Drop your Image

Choose File No file chosen

Upload

Image preview



Tag

It will take a while for the tags to load.

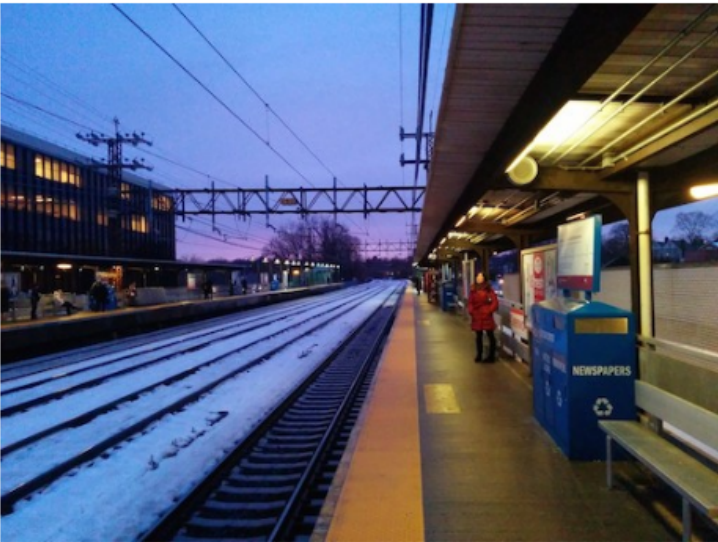


Tag

Confidence

Image tag will be generated after a few seconds.

Image preview

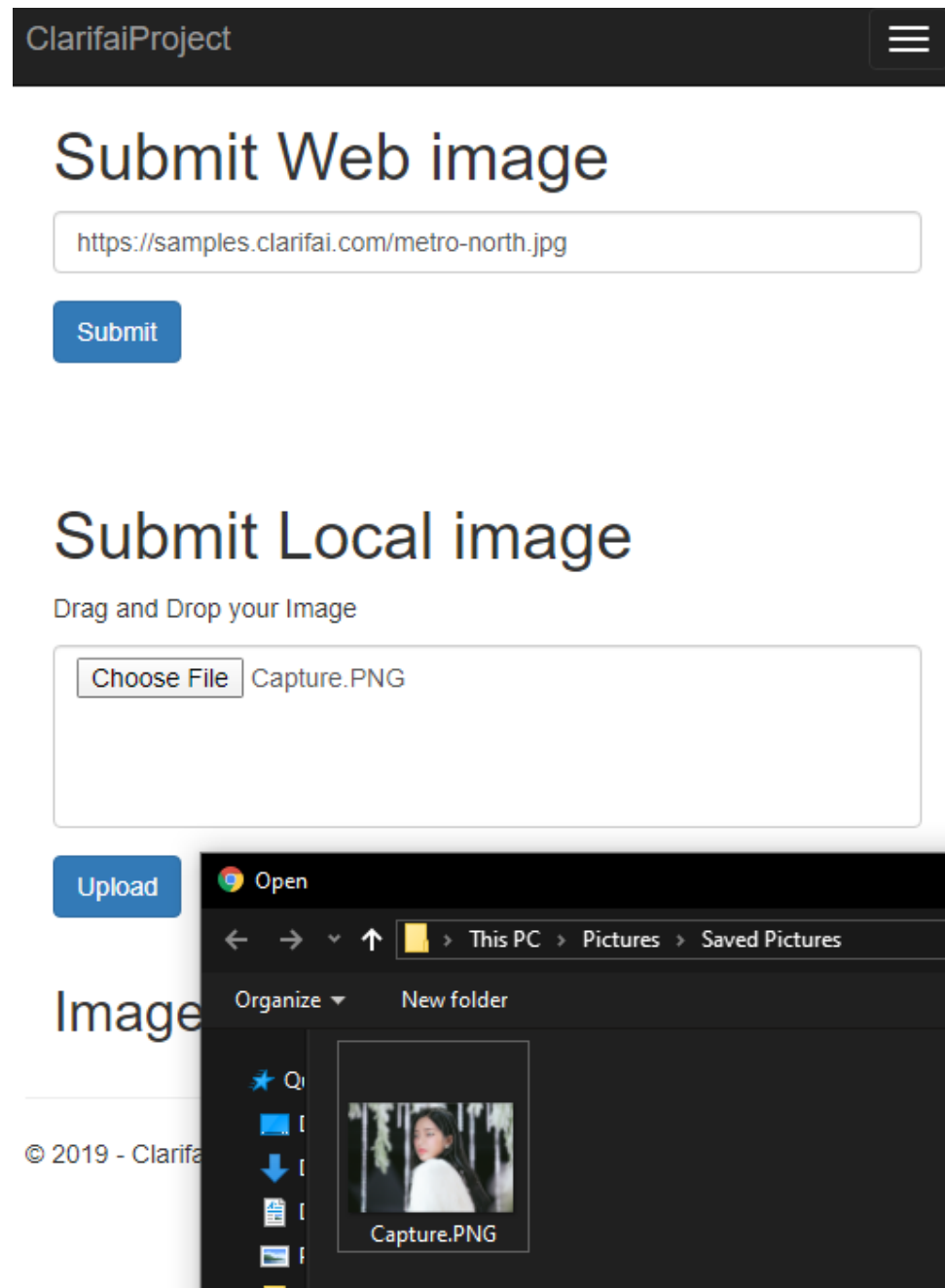


Tag

Tag	Confidence
train	0.9996053
railway	0.9992987
subway system	0.99825156
station	0.9980105
locomotive	0.99725723
transportation system	0.9969802
travel	0.9889797
commuter	0.98087525
platform	0.98064405
light	0.97420406
train station	0.96874046
blur	0.9672204
city	0.9614798
road	0.9613829

Submit: Local Image

Image can be uploaded from local computer via drag and drop.



Uploaded image will undergo tagging.

ClarifaiProject



Submit Web image

<https://samples.clarifai.com/metro-north.jpg>

Submit

Submit Local image

Drag and Drop your Image

Choose File Capture.PNG

Upload

Image preview



Tag

It will take a while for the tags to load.



Tag

Confidence

Image tag will be generated after a few seconds.

Image preview



Tag

Tag	Confidence
fashion	0.99193025
girl	0.9905751
winter	0.98973143
model	0.9879178
woman	0.98600954
portrait	0.9820337
beautiful	0.97932243
snow	0.97022843
cold	0.9481993

In the case of error, server will reconnect and prompt to check.

Tag

Web Server has failed to establish a connection to Clarifai. Reconnecting...
(Check your network and API key in ClarifaiModel)



Tag	Confidence

Browser (javascript)

Located in *ClarifaiProject_js* Folder

Insert your Clarifai API key inside keys.js from <https://portal.clarifai.com/apps> .

API Keys

[DETAILS](#)


ClarifaiProject-all-scopes

XX

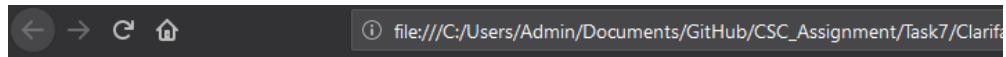
Dec 12, 2019



[CREATE NEW API KEY](#)

```
JS keys.js >  myApiKey  
1  var myApiKey = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX';
```

This Javascript version is the same version as ASP.NET, without a web server.



ClarifaiProject

Submit Web image

<https://samples.clarifai.com/metro-north.jpg>

Submit

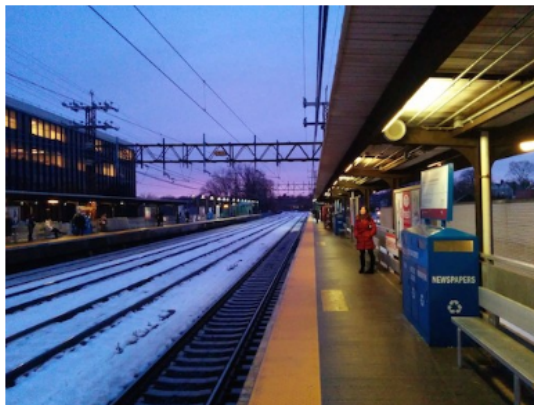
Submit Local image

Drag and Drop your Image

Browse... Capture.PNG

Upload

Image preview



Tag

Tag	Confidence
train	0.9996053
railway	0.9992987
subway system	0.99825156
station	0.9980105
locomotive	0.99725723
transportation system	0.9969802