



School of Computing | Diploma in Information Technology

ST0280 Cloud and Service Development

Continuous Assessment 1

Done by:

Zachary Tan Chien Loong (p1829176)

Lee Wei Xian(p1828878)

[LINK TO GITHUB REPO](#)

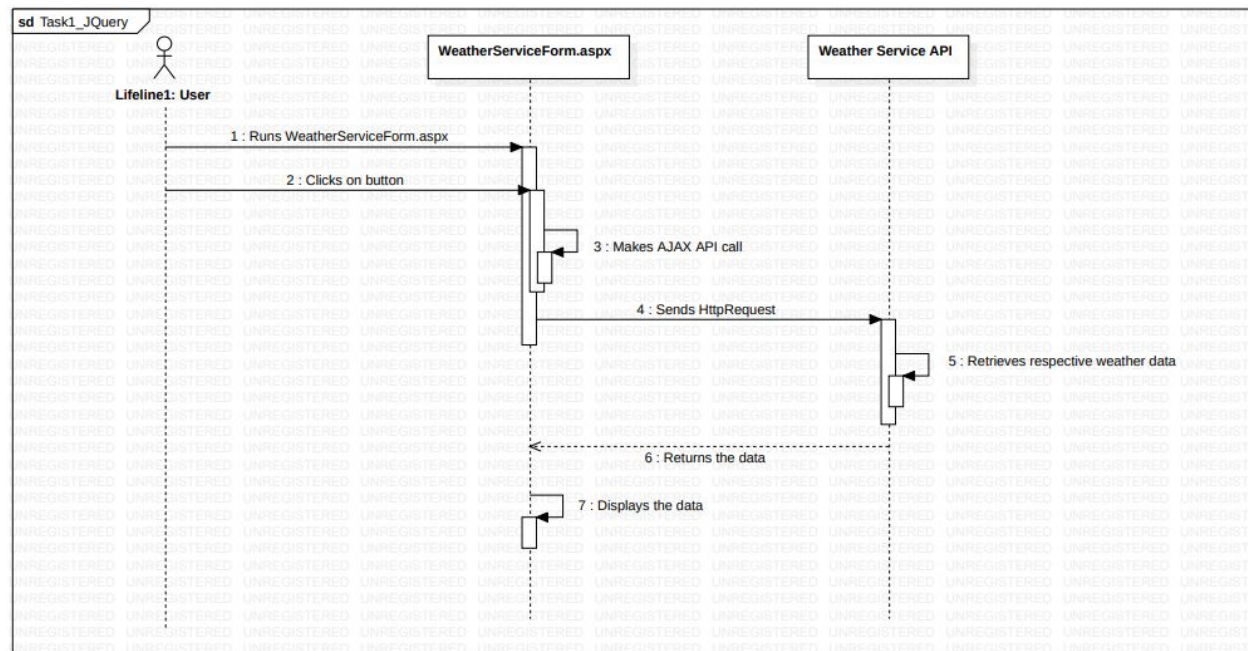
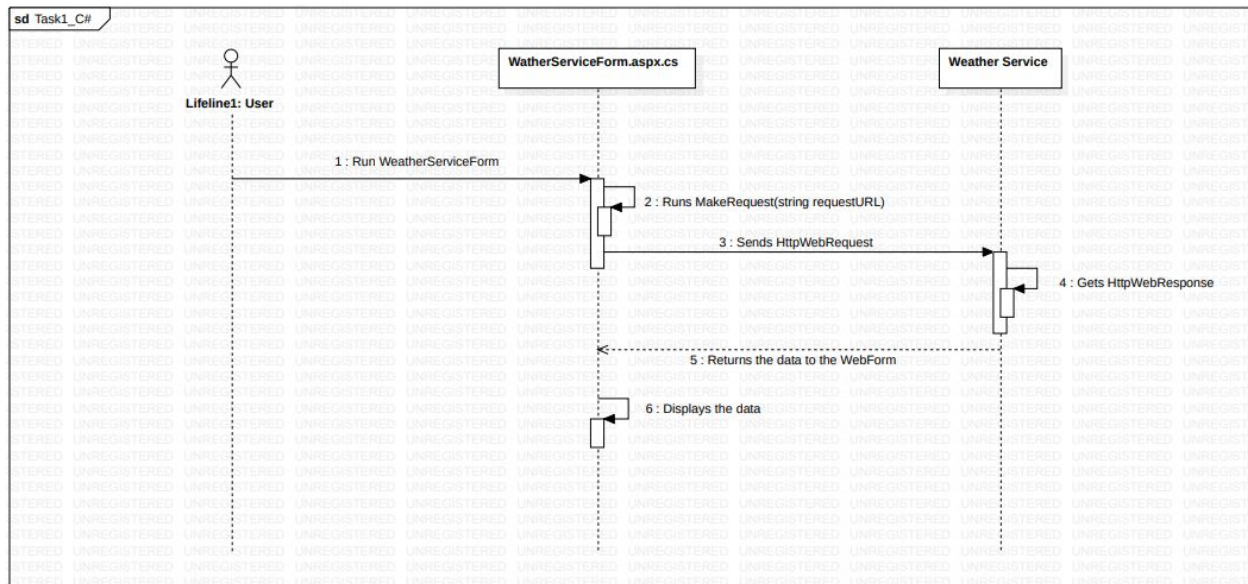
# Task1

## Web-API Documentation:

### 1. Get weather service

METHOD	GET
ROUTE	https://api.data.gov.sg/v1/environment/2-hour-weather-forecast
OPTIONAL PARAMETERS	date_time: String date: String
EXAMPLE	<div>GET    https://api.data.gov.sg/v1/environment/2-hour-weather-forecast</div>
RESPONSE	<pre>"area_metadata": [   {     "name": "Ang Mo Kio",     "label_location": {       "latitude": 1.375,       "longitude": 103.839     }   },   {     "name": "Bedok",     "label_location": {       "latitude": 1.321,       "longitude": 103.924     }   },   {     "name": "Bishan",     "label_location": {       "latitude": 1.350772,       "longitude": 103.839     }   } ],</pre>

## Sequence Diagram:



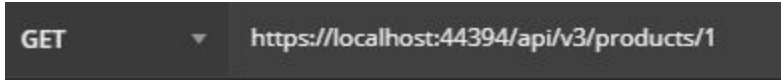

## Task2

### Web-API Documentation:

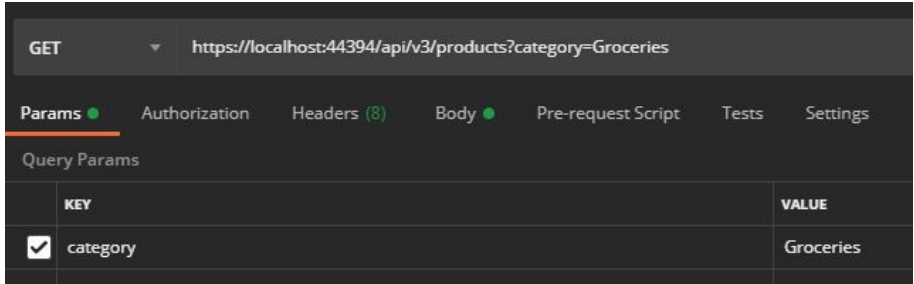

#### 1. Get all products

METHOD	GET
ROUTE	/api/v3/products
EXAMPLE	<div>GET <a href="https://localhost:44394/api/v3/products">https://localhost:44394/api/v3/products</a></div>
RESPONSE	<pre>[   {     "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   },   {     "Id": 5,     "Name": "Pokeball",     "Category": "Level 99",     "Price": 99.0   } ]</pre>

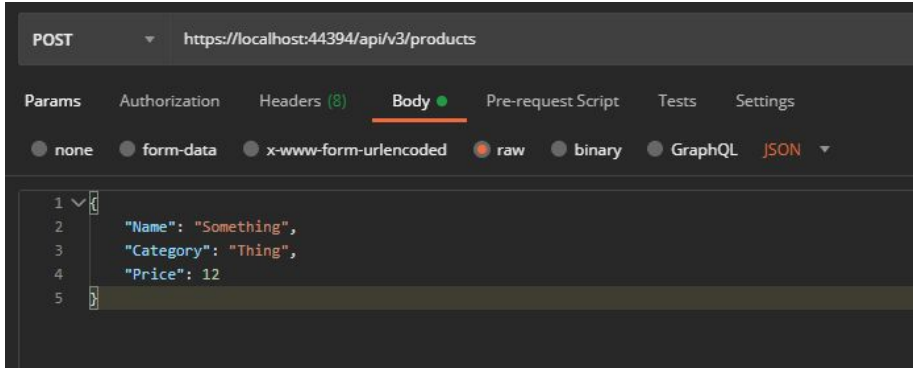

## 2. Get product by id

METHOD	GET
ROUTE	/api/v3/products/{id:INT:min(1)}
EXAMPLE	
RESPONSE	

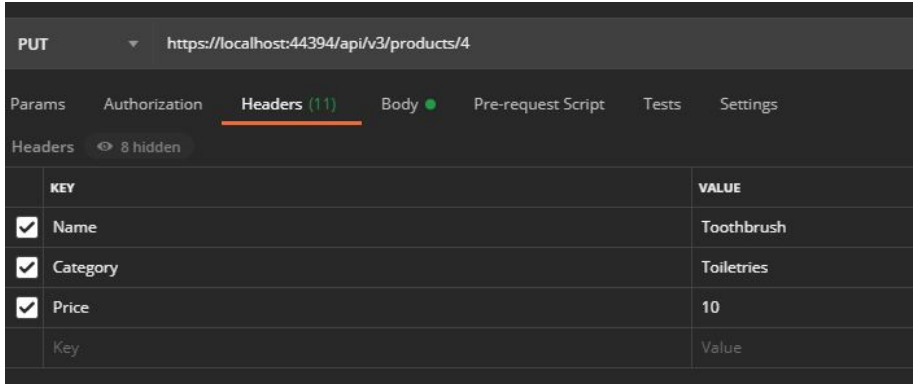
### 3. Get product by category

METHOD	GET				
ROUTE	/api/v3/products?category={categoryName}				
EXAMPLE	 <p>The screenshot shows a REST client interface with the following details:</p> <ul style="list-style-type: none"><li>Method: GET</li><li>URL: https://localhost:44394/api/v3/products?category=Groceries</li><li>Active Tab: Params</li><li>Query Params Table:</li></ul> <table><thead><tr><th>KEY</th><th>VALUE</th></tr></thead><tbody><tr><td><input checked="" type="checkbox"/> category</td><td>Groceries</td></tr></tbody></table>	KEY	VALUE	<input checked="" type="checkbox"/> category	Groceries
KEY	VALUE				
<input checked="" type="checkbox"/> category	Groceries				
RESPONSE	 <pre>{   "Id": 1,   "Name": "Tomato soup",   "Category": "Groceries",   "Price": 1.39 }</pre>				

#### 4. Create new product

METHOD	POST
ROUTE	/api/v3/products
EXAMPLE	
RESPONSE	

## 5. Update product by id

METHOD	PUT										
ROUTE	/api/v3/products/{id:INT}										
EXAMPLE	 <p>The screenshot shows a REST client interface with the following details:</p> <ul style="list-style-type: none"><li><b>Method:</b> PUT</li><li><b>URL:</b> https://localhost:44394/api/v3/products/4</li><li><b>Headers:</b> 11 headers are listed, with 8 hidden. The visible headers are:<table><thead><tr><th>KEY</th><th>VALUE</th></tr></thead><tbody><tr><td><input checked="" type="checkbox"/> Name</td><td>Toothbrush</td></tr><tr><td><input checked="" type="checkbox"/> Category</td><td>Toiletries</td></tr><tr><td><input checked="" type="checkbox"/> Price</td><td>10</td></tr><tr><td>Key</td><td>Value</td></tr></tbody></table></li></ul>	KEY	VALUE	<input checked="" type="checkbox"/> Name	Toothbrush	<input checked="" type="checkbox"/> Category	Toiletries	<input checked="" type="checkbox"/> Price	10	Key	Value
KEY	VALUE										
<input checked="" type="checkbox"/> Name	Toothbrush										
<input checked="" type="checkbox"/> Category	Toiletries										
<input checked="" type="checkbox"/> Price	10										
Key	Value										

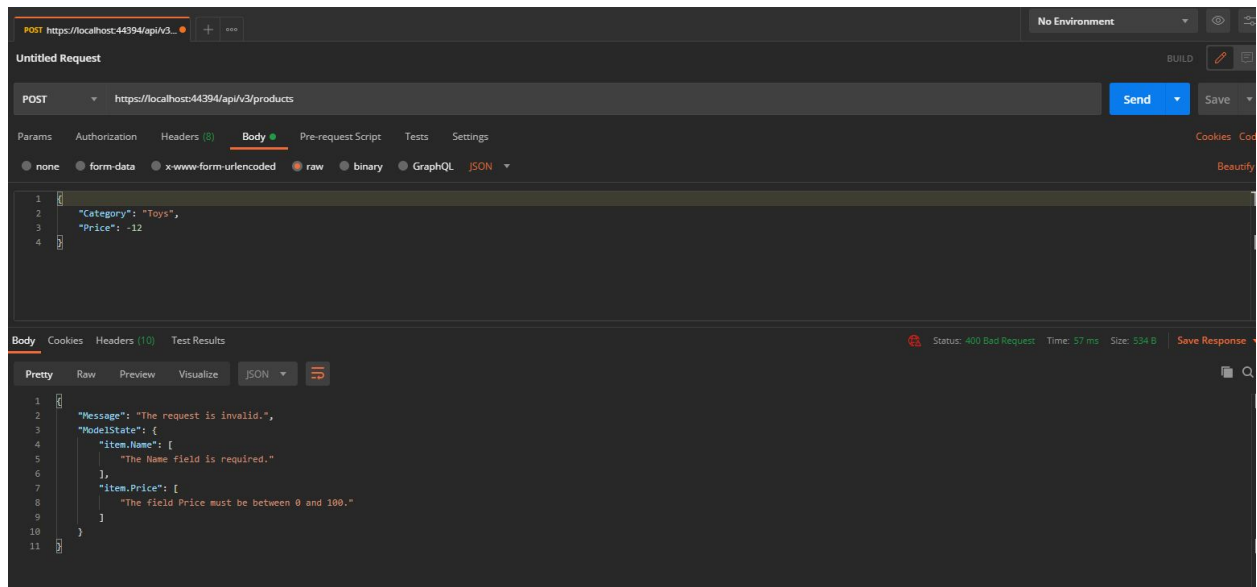


## 6. Delete product

METHOD	DELETE
ROUTE	/api/v3/products/{id:INT}
EXAMPLE	<div>DELETE ▼ https://localhost:44394/api/v3/products/1</div>

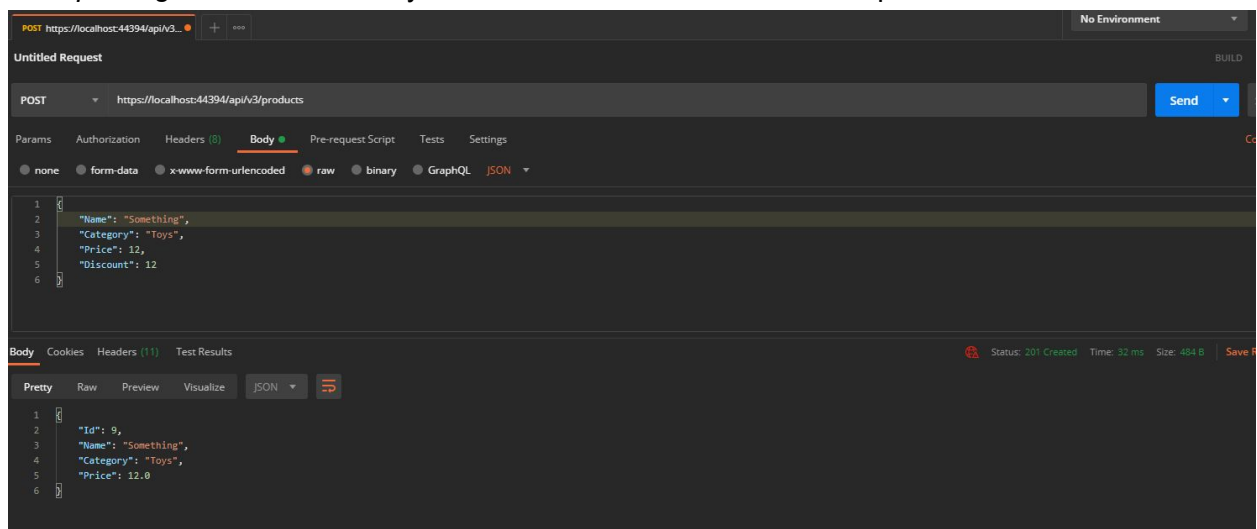
## Screenshots:

*Under-posting*: Situation whereby the client leaves out some properties when sending request(s) to the controller.



[This happens when there is a case of under-posting]

*Over-posting*: Situation whereby the client sends more data than expected to the controller.



[This happens when there is a case of over-posting]

## Task3

### Web-API Documentation:

#### 1. Register an account

METHOD	POST
ROUTE	/api/Account/Register
PARAMETERS REQUIRED	email: String, password: String, confirmPassword: String, reCaptcha Site-Key: String

#### 2. Login to an account

METHOD	POST
ROUTE	/api/Token
PARAMETERS REQUIRED	username: String, password: String, 'password': grant_type

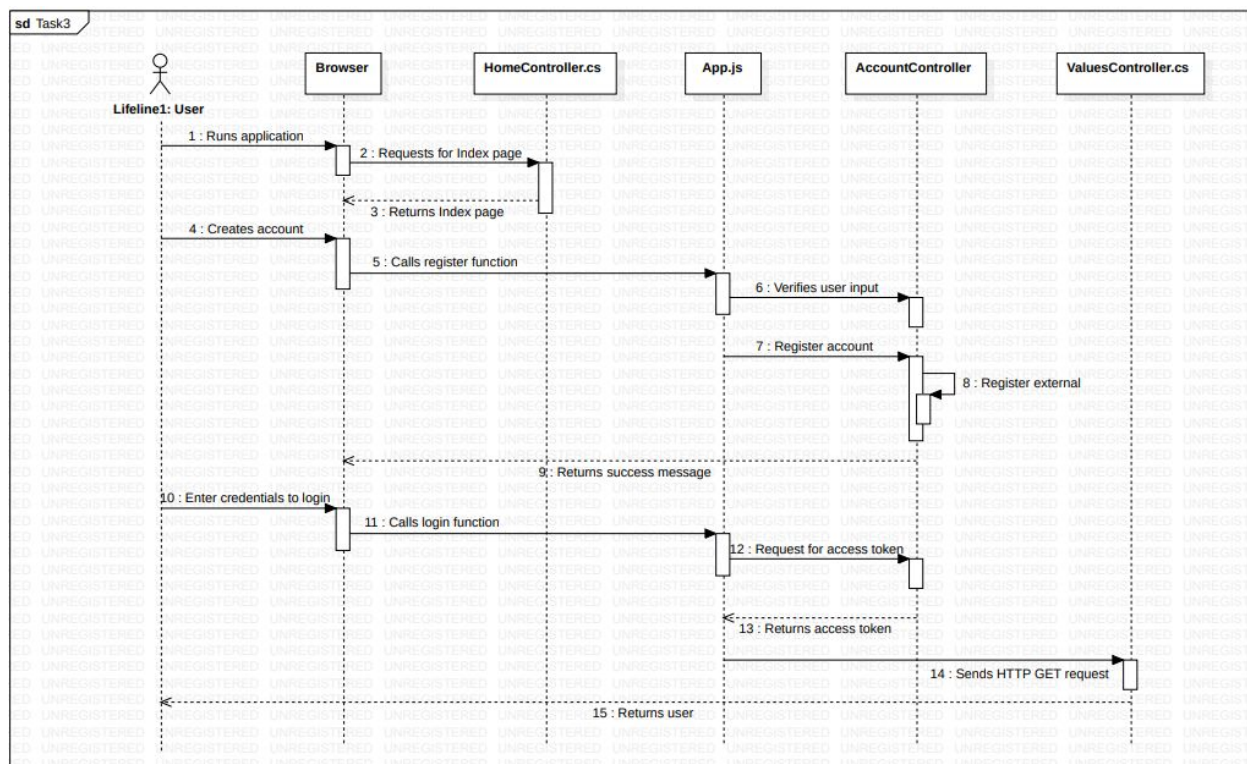
#### 3. Logout of an account

METHOD	POST
ROUTE	/api/Account/Logout

#### 4. Get Values(Authorization)

METHOD	GET
ROUTE	/api/values

#### Sequence Diagram:



## Screenshots:

### Register Successful:

The screenshot shows the Postman interface with a workspace named "My Workspace". A notification at the top indicates an update has been downloaded. The left sidebar shows a "History" tab with a list of recent requests, all of which are POST requests to the endpoint `https://localhost:44320/api/Account/Register`. The main panel displays an "Untitled Request" for the same endpoint. The request is a POST method with the body type set to "x-www-form-urlencoded". The body contains two form fields: "confirmpassword" with the value "Ntuchub0310!" and "RecaptchaSiteKey" with the value "6LffwgaAAAAADFoQijbkD0XJV6oys6ygcPa77wd". The response is displayed in the "Body" tab, showing a status of "200 OK", a time of "299 ms", and a size of "328 B". The response body is a single character "1".

### Register(email invalid!):

The screenshot shows the Postman interface with the same workspace. The left sidebar shows the same history of requests. The main panel displays the same "Untitled Request" for the endpoint `https://localhost:44320/api/Account/Register`. The request is a POST method with the body type set to "x-www-form-urlencoded". The body contains two form fields: "email" with the value "alfred@" and "password" with the value "Ntuchub0310!". The response is displayed in the "Body" tab, showing a status of "400 Bad Request", a time of "7.89 s", and a size of "472 B". The response body is a JSON object indicating an invalid request.

```
1 {
2   "Message": "The request is invalid.",
3   "ModelState": {
4     "": [
5       "Email 'alfred@' is invalid."
6     ]
7   }
8 }
```

## Register(password invalid):

Postman interface showing a failed POST request to `https://localhost:44320/api/Account/Register`. The request body is `x-www-form-urlencoded` with the following data:

Key	Value	Description
confirmpassword	Password	
RecaptchaSiteKey	6LFFwgaAAAAADFoQjBkd0XJV6oys6ygcPa77wd	

The response status is **400 Bad Request** (Time: 98 ms, Size: 559 B). The response body is:

```
1 {
2   "Message": "The request is invalid.",
3   "ModelState": {
4     "": [
5       "Passwords must have at least one non letter or digit character. Passwords must have at least one digit ('0'-'9')."
6     ]
5   }
6 }
```

## Register(password do not match):

Postman interface showing a failed POST request to `https://localhost:44320/api/Account/Register`. The request body is `x-www-form-urlencoded` with the following data:

Key	Value	Description
confirmpassword	Ntuchub031	
RecaptchaSiteKey	6LFFwgaAAAAADFoQjBkd0XJV6oys6ygcPa77wd	

The response status is **400 Bad Request** (Time: 63 ms, Size: 519 B). The response body is:

```
1 {
2   "Message": "The request is invalid.",
3   "ModelState": {
4     "model.ConfirmPassword": [
5       "The password and confirmation password do not match."
6     ]
5   }
6 }
```

## Login(Successful):

The screenshot shows the Postman interface with a successful POST request to `https://localhost:44320/Token`. The request body is in the 'Body' tab, showing a JSON response with an access token and other metadata.

**Request:**

- Method: POST
- URL: `https://localhost:44320/Token`
- Body (JSON):

```
{  "username": "jjlin12@gmail.com",  "password": "Ntuchub0310!"}
```

**Response:**

- Status: 200 OK
- Time: 197 ms
- Size: 1.49 KB
- Body (JSON):

```
{  "access_token": "t1M5fkrYn59mr4h-28Xnjyz8pU3flvF5uiw9hQ9nQ7_Pwe5Ug1bP8qV39x2F8XF4AZf-B0vsutb13aj4V3Fxy5wGg_fQEUvWd9Xib-wZPFvA_pZt01ngK1HyZ4nKHU67sm-1L2j0whdvyvm8P-fukthw0v4SLiby-57Bw9BA4hpqRrB8M5Ux4X-17z3Wr_gukpgi4ZrvncpFYrce1RxgLfshyOTDK_CukqiHrgh59C2Scn5A8Cg-XY8CefUDKfrRE9Mqhb9hL_VBRGhbME8g7zRpXbGnet5c1C-Ka7roL39g1gK0dsSy6lQdZB9R-Sbw_Vwj7bW9auau0szp557IK030WHPa4Fec6xnc07k7YZXh0n0-7KiZw5Rm5I1s1Ez4SVt8YEBEg1-JRZUu5719LBBRK7ycVn7zuPUmmaQahthsbkGnFDXh_Q6U20Yz1R9nan2VyY11yy1ZPX-LQK16_U8mo6_R9sXTaMuor7P69GxE",  "token_type": "bearer",  "expires_in": 1209599,  "userName": "jjlin12@gmail.com",  ".issued": "Sun, 13 Dec 2020 12:45:06 GMT",  ".expires": "Sun, 27 Dec 2020 12:45:06 GMT"}
```

## Login Unsuccessful:

The screenshot shows the Postman interface with an unsuccessful GET request to `https://localhost:44320/Token`. The response body shows an error message indicating that the user name or password is incorrect.

**Request:**

- Method: GET
- URL: `https://localhost:44320/Token`

**Response:**

- Status: 400 Bad Request
- Time: 31 ms
- Size: 422 B
- Body (JSON):

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



## GetValues(Successful):

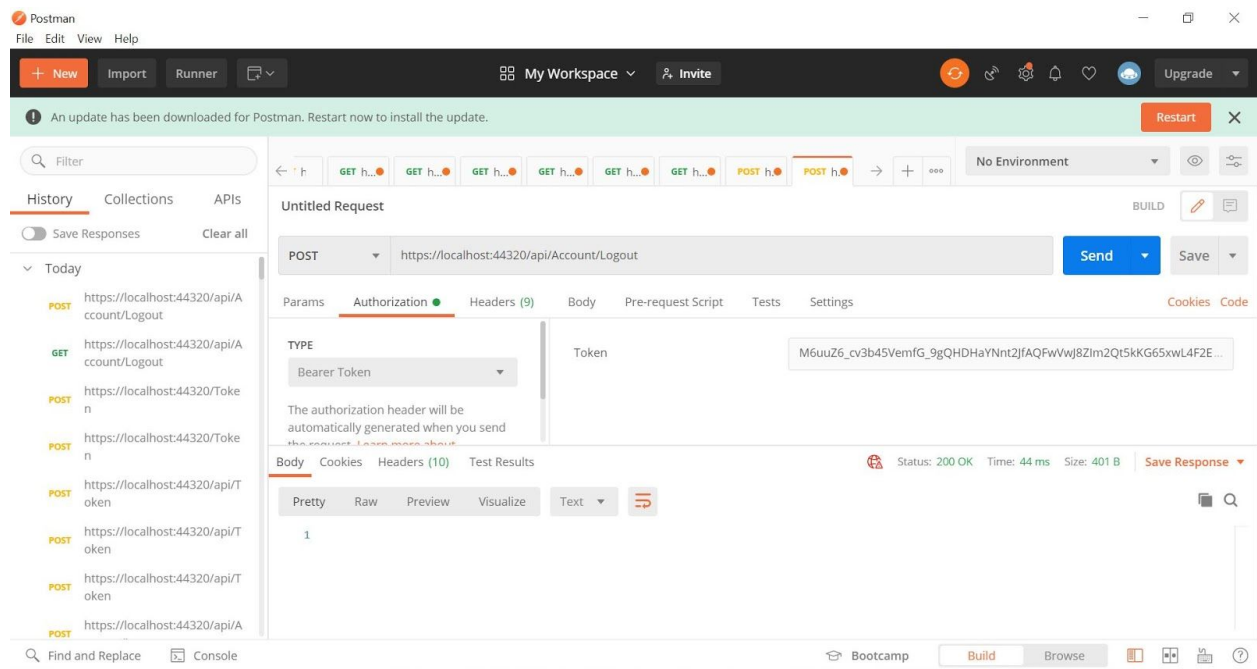
The screenshot shows the Postman application interface. On the left, the 'History' tab is active, displaying a list of recent requests. The main panel shows an 'Untitled Request' for a GET method to the URL 'https://localhost:44320/api/values/1'. The 'Authorization' tab is selected, showing a 'Bearer Token' type with a token value 'V7dkQjv35ZmUn\_FPh1E-gXSjo5I8760qqShgehs1IFy176G0uPeycdIDGtc...'. The 'Body' tab is also visible, showing a JSON response: '1 "Hello, jjlin12@gmail.com."'. The status bar at the bottom indicates 'Status: 200 OK', 'Time: 17 ms', and 'Size: 375 B'.

## Unsuccessful:

The screenshot shows the Postman application interface. On the left, the 'History' tab is active, displaying a list of recent requests. The main panel shows an 'Untitled Request' for a GET method to the URL 'https://localhost:44320/api/values'. The 'Body' tab is selected, showing a JSON response: '1 {"Message": "Authorization has been denied for this request."}'. The status bar at the bottom indicates 'Status: 401 Unauthorized', 'Time: 505 ms', and 'Size: 457 B'. A notification banner at the top states: 'An update has been downloaded for Postman. Restart now to install the update.' with a 'Restart' button.



## AccountLogout:



## Task4

### Web-API Documentation:

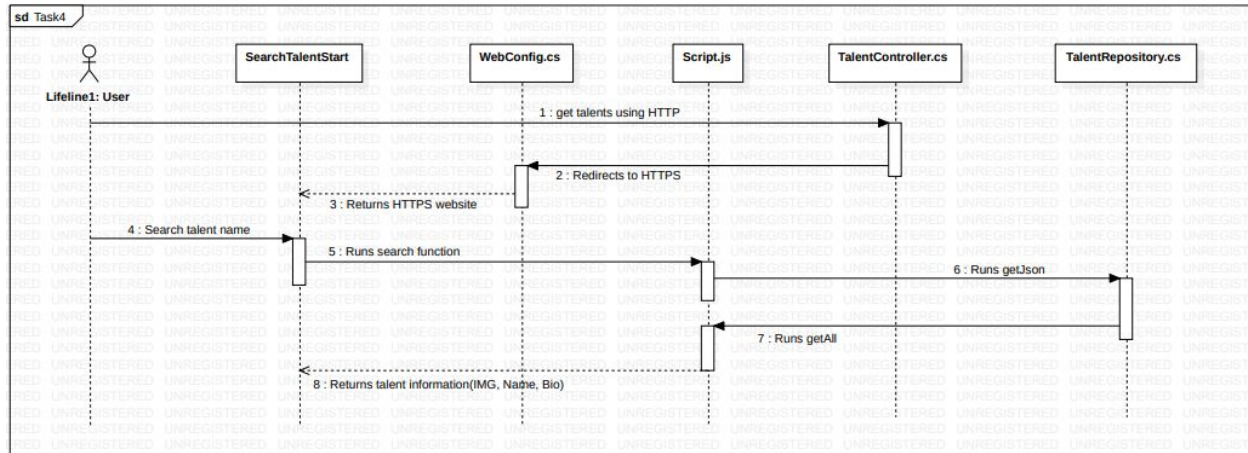
#### 1. Retrieve all talents

METHOD	GET
ROUTE	/api/talents

#### 2. Retrieve talent by id

METHOD	GET
ROUTE	/api/values/{id:INT}

## Sequence Diagram:



## Screenshots:

### Get Api/Talents:

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

- Request:** GET `http://localhost:54602/api/talents`
- Status:** 200 OK
- Time:** 283 ms
- Size:** 2.43 KB
- Response Body (JSON):**

```
[{"Id": 0, "Name": "Barot Bellingham", "ShortName": "Barot_Bellingham", "Reknown": "Royal Academy of Painting and Sculpture", "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"}, {"Id": 1, "Name": "Jonathan G. Ferrar II", "ShortName": "Jonathan Ferrar"}]
```

## Get Api/Talents/{id}:

REST client interface showing a GET request to `http://localhost:54602/api/talents/1`. The response is a JSON object:

```
{
  "Id": 1,
  "Name": "Jonathan G. Ferrar II",
  "ShortName": "Jonathan_Ferrar",
  "Reknown": "Artist to Watch in 2012",
  "Bio": "The Artist to Watch in 2012 by the London Review, Johnathan has already sold one of the highest priced-commissions paid to an art student, ever on record. The piece, entitled Gratitude Resort, a work in oil and mixed media, was sold for $750,000 and Jonathan donated all the proceeds to Art for Peace, an organization that provides college art scholarships for creative children in developing nations"
}
```

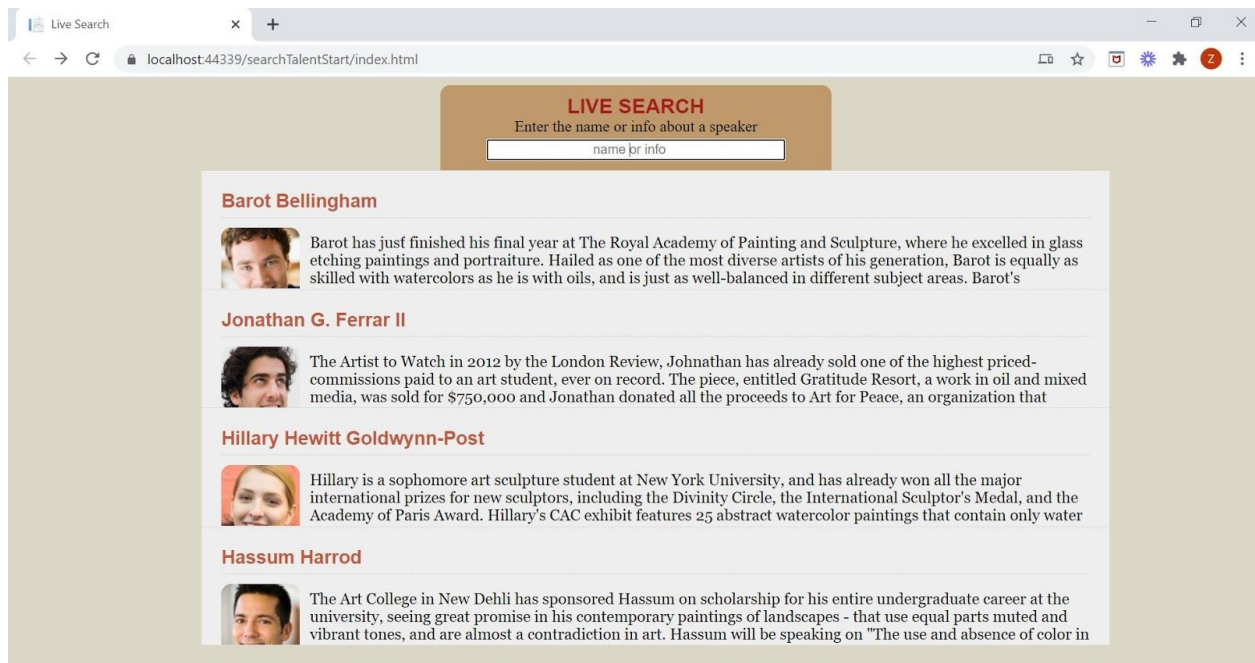
## BrowserHTTPS:

Web browser interface showing the response to `https://localhost:44339/api/talents`. The response is an XML document:

```
<ArrayOfTalent xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://schemas.datacontract.org/2004/07/SearchTalent.Models">
  <Talent>
    <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</Bio>
    <Id>0</Id>
    <Name>Barot Bellingham</Name>
    <Reknown>Royal Academy of Painting and Sculpture</Reknown>
    <ShortName>Barot_Bellingham</ShortName>
  </Talent>
  <Talent>
    <Bio>The Artist to Watch in 2012 by the London Review, Johnathan has already sold one of the highest priced-commissions paid to an art student, ever on record. The piece, entitled Gratitude Resort, a work in oil and mixed media, was sold for $750,000 and Jonathan donated all the proceeds to Art for Peace, an organization that provides college art scholarships for creative children in developing nations</Bio>
    <Id>1</Id>
    <Name>Jonathan G. Ferrar II</Name>
    <Reknown>Artist to Watch in 2012</Reknown>
    <ShortName>Jonathan_Ferrar</ShortName>
  </Talent>
  <Talent>
    <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.</Bio>
    <Id>2</Id>
    <Name>Hillary Hewitt Goldwynn-Post</Name>
  </Talent>
</ArrayOfTalent>
```

The browser's developer tools show the network request details, including the request URL, method (GET), status code (302 Found), and response headers.

## LiveSearch:



## Task5

### Web-API Documentation:

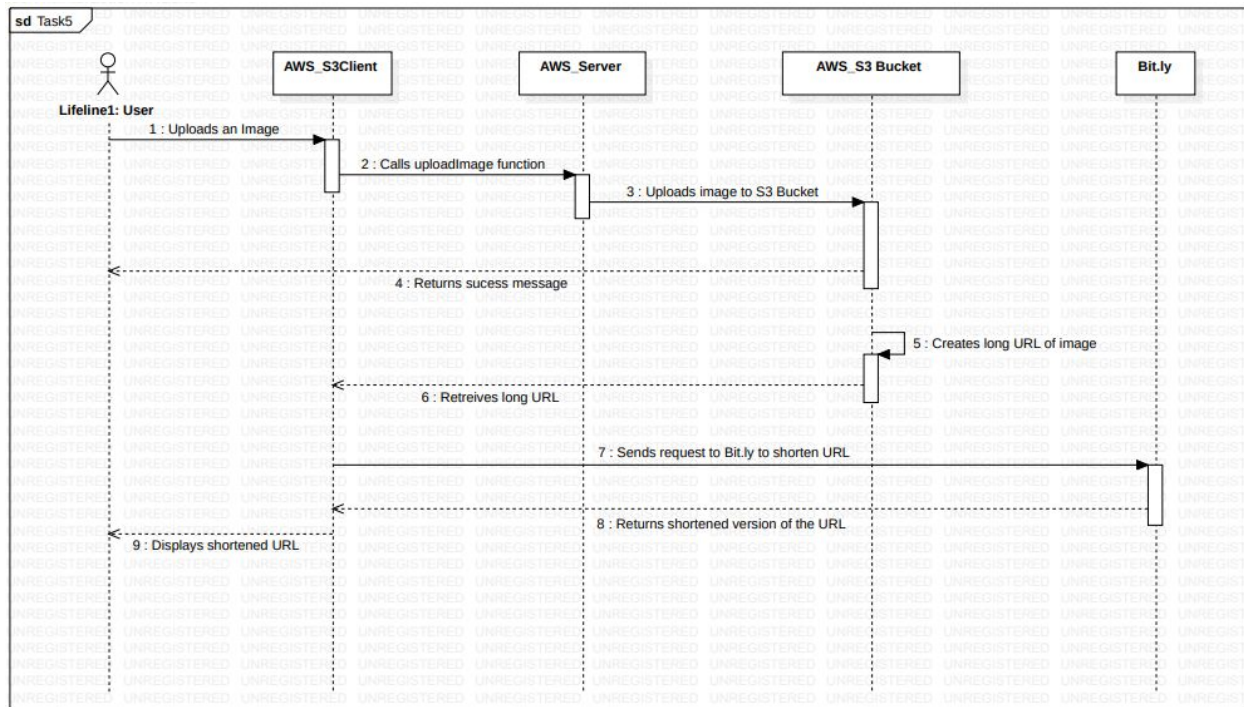
1. Upload image to S3

METHOD	POST
--------	------

2. Convert long URL to Bit.ly URL








METHOD	POST
ROUTE	https://api-ssl.bitly.com/v4/shorten

## Sequence Diagram:



## Screenshots:

### AWS S3 Bucket:

<input type="checkbox"/>	Name ▲	Type ▼	Last modified ▼	Size ▼	Storage class ▼
<input type="checkbox"/>	 <a href="#">Barot_Bellingham_tn.jpg</a>	jpg	December 27, 2020, 01:56:51 (UTC+08:00)	12.5 KB	Standard
<input type="checkbox"/>	 <a href="#">booked.PNG</a>	PNG	January 3, 2021, 12:12:13 (UTC+08:00)	106.9 KB	Standard
<input type="checkbox"/>	 <a href="#">Bowlofrice.PNG</a>	PNG	January 3, 2021, 11:48:09 (UTC+08:00)	59.8 KB	Standard
<input type="checkbox"/>	 <a href="#">Burger.PNG</a>	PNG	December 31, 2020, 00:00:05 (UTC+08:00)	17.0 KB	Standard
<input type="checkbox"/>	 <a href="#">connect.PNG</a>	PNG	January 3, 2021, 11:50:50 (UTC+08:00)	624.0 KB	Standard
<input type="checkbox"/>	 <a href="#">Constance_Smith_tn.jpg</a>	jpg	December 28, 2020, 23:11:02 (UTC+08:00)	12.5 KB	Standard
<input type="checkbox"/>	 <a href="#">drs.JPG</a>	JPG	January 3, 2021, 11:53:40 (UTC+08:00)	50.5 KB	Standard

## Task6

### Web-API Documentation:

#### 1. One time purchase

METHOD	POST
ROUTE	/purchase
PARAMETERS REQUIRED	ID: Integer, Items: Item Object

#### 2. Create checkout session

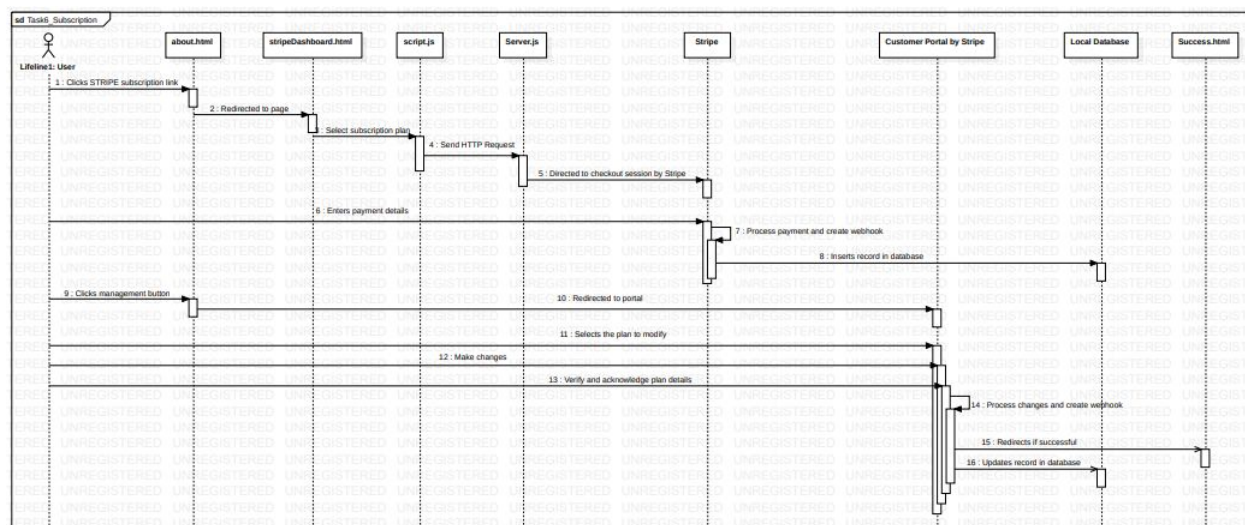
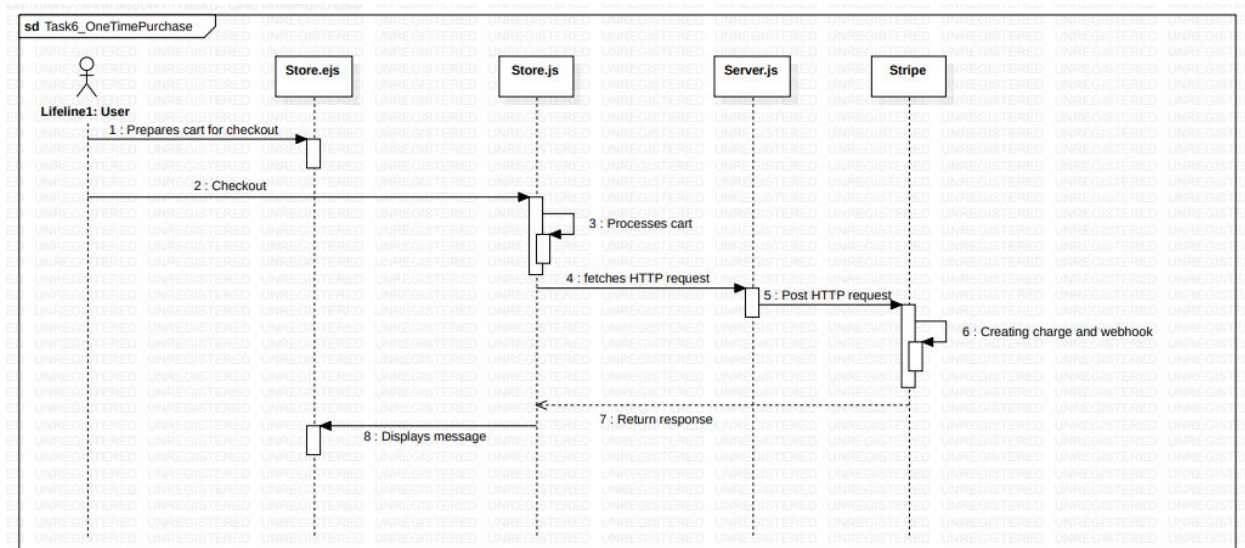
METHOD	POST
ROUTE	/create-checkout-session
PARAMETERS REQUIRED	priceId: String

#### 3. Webhook

METHOD	POST
ROUTE	/webhook
PARAMETERS REQUIRED	event: String



## Sequence Diagram:



# Task7

Sequence Diagram:

