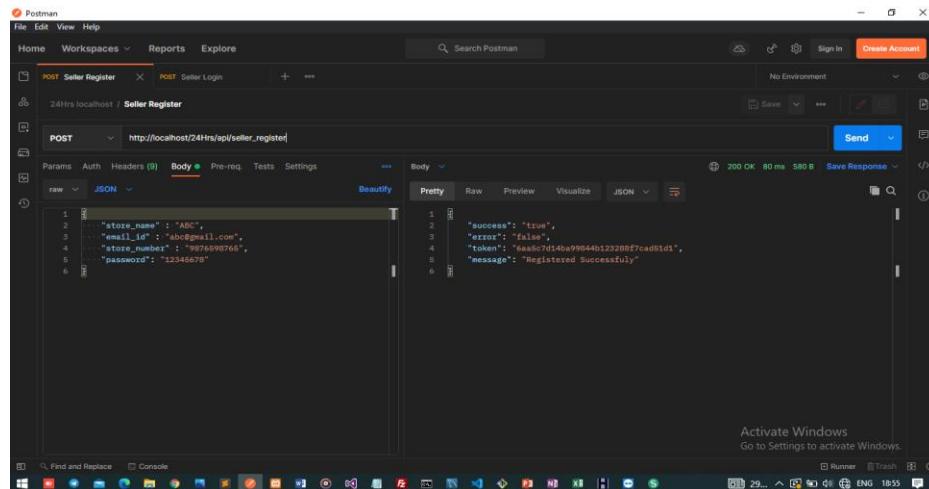


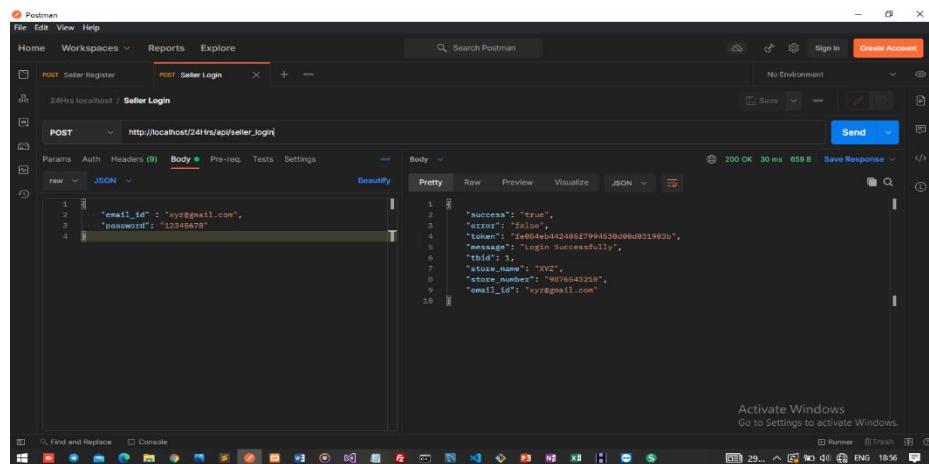
Seller Register:



Postman screenshot showing the Seller Register API call. The request URL is `http://localhost:24hrs/api/seller_register`. The response status is 200 OK, 80 ms, 580 B. The response body is:

```
1 "success": true,
2 "error": false,
3 "token": "eaascd7d4ba99844b123208f7cad51d1",
4 "message": "Registered Successfully"
```

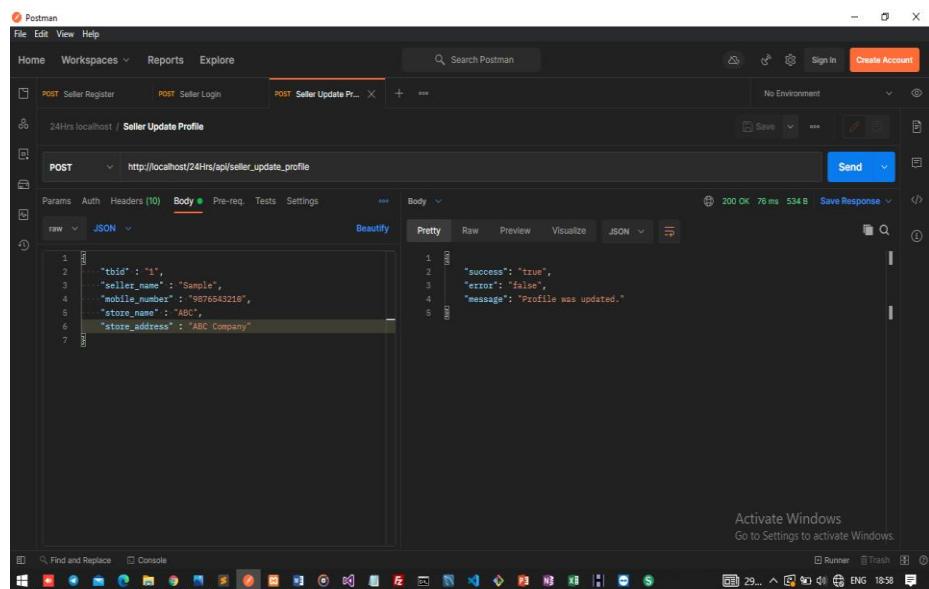
Seller Login:



Postman screenshot showing the Seller Login API call. The request URL is `http://localhost:24hrs/api/seller_login`. The response status is 200 OK, 30 ms, 659 B. The response body is:

```
1 "success": true,
2 "error": false,
3 "token": "fe664eb42488f7994830d08d019850",
4 "message": "Login successfully",
5 "uid": 1,
6 "store_name": "XYZ",
7 "store_number": "9876543210",
8 "email_id": "xyz@gmail.com"
```

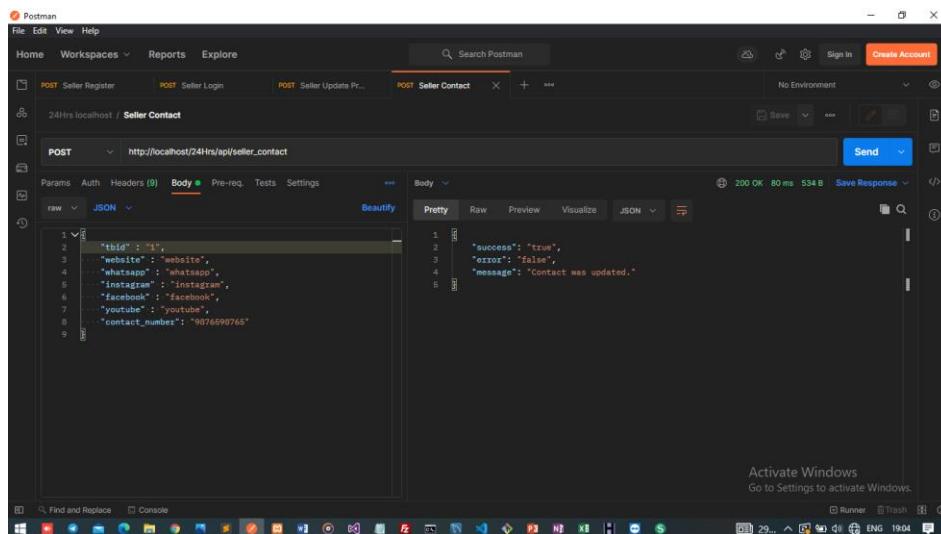
Seller Update:



Postman screenshot showing the Seller Update Profile API call. The request URL is `http://localhost:24hrs/api/seller_update_profile`. The response status is 200 OK, 76 ms, 534 B. The response body is:

```
1 "success": true,
2 "error": false,
3 "message": "Profile was updated."
```

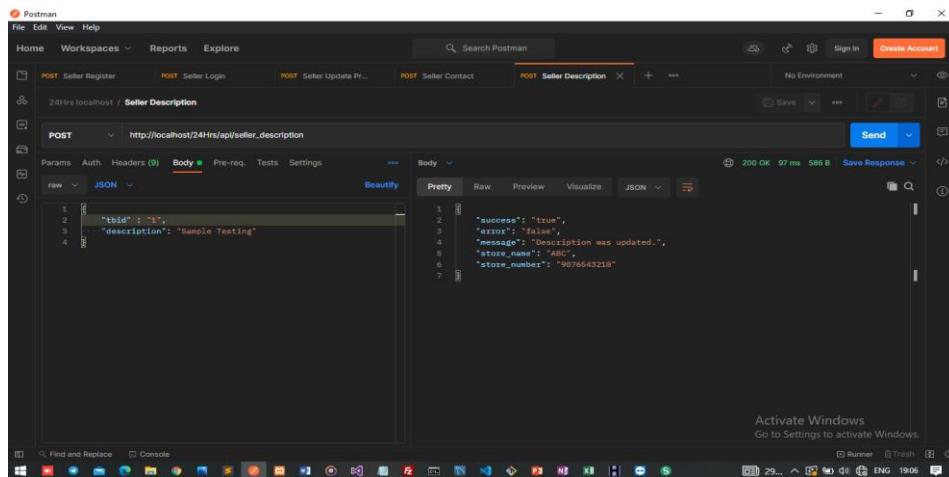
Seller Contact:



Postman screenshot showing a successful POST request to the Seller Contact endpoint. The response body is:

```
1 "success": true,
2 "error": false,
3 "message": "Contact was updated."
```

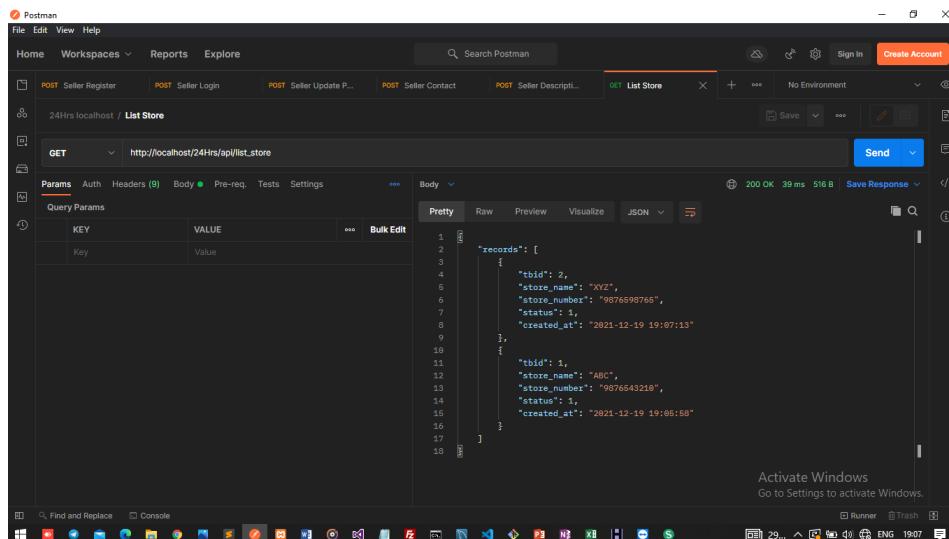
Seller Description:



Postman screenshot showing a successful POST request to the Seller Description endpoint. The response body is:

```
1 "success": true,
2 "error": false,
3 "message": "Description was updated.",
4 "store_name": "ABC",
5 "store_number": "9876543210"
```

List Store:



Postman screenshot showing a successful GET request to the List Store endpoint. The response body is:

```
1 "records": [
2   {
3     "tbid": 2,
4     "store_name": "XYZ",
5     "store_number": "9876598768",
6     "status": 1,
7     "created_at": "2021-12-19 19:07:13"
8   },
9   {
10     "tbid": 1,
11     "store_name": "ABC",
12     "store_number": "9876543210",
13     "status": 1,
14     "created_at": "2021-12-19 19:05:58"
15   }
16 ]
```

Create Category:

The screenshot shows the Postman application interface. A POST request is made to `http://localhost:24Hrs/api/create_category`. The request body contains the JSON object `{"category_name": "Grocery"}`. The response status is 201 Created, with a response time of 59 ms and a size of 540 B. The response body is a JSON object with keys `success`, `error`, and `message`, all set to their respective values.

Update Category:

The screenshot shows the Postman application interface. A POST request is made to `http://localhost:24Hrs/api/update.category`. The request body contains the JSON object `{"cbid": "4", "category_name": "Grocery"}`. The response status is 200 OK, with a response time of 59 ms and a size of 535 B. The response body is a JSON object with keys `success`, `error`, and `message`, all set to their respective values.

Read Category:

The screenshot shows the Postman application interface. A GET request is made to `http://localhost:24Hrs/api/read_category`. The request includes various headers such as Cache-Control, Postman-Token, Host, User-Agent, Accept, Accept-Encoding, Connection, and token. The response status is 200 OK, with a response time of 17 ms and a size of 385 B. The response body is a JSON object with a key `records` containing a single item with fields `cbid`, `category`, `status`, and `created_at`.

Read One Category:

The screenshot shows the Postman application interface. The top navigation bar includes 'File', 'Edit', 'View', 'Help', 'Home', 'Workspaces', 'Reports', and 'Explore'. A search bar 'Search Postman' is at the top right. The main workspace shows a 'POST Read One category' request. The URL is 'http://localhost:24Hrs/api/read_one_category?c=Grocery'. The 'Headers' tab is selected, showing standard auto-generated headers like Cache-Control, Postman-Token, Content-Length, Host, User-Agent, Accept, Accept-Encoding, Connection, and token. The 'Body' tab shows a JSON response with a success status of true, an error of false, and a single record with tbid 1, category_name 'Grocery', and status 1. The status bar at the bottom indicates '200 OK 32 ms 490 B'.

Search Category:

The screenshot shows the Postman application interface. The top navigation bar includes 'File', 'Edit', 'View', 'Help', 'Home', 'Workspaces', 'Reports', and 'Explore'. A search bar 'Search Postman' is at the top right. The main workspace shows a 'GET Search Category' request. The URL is 'http://localhost:24Hrs/api/search_category?c=Grocery'. The 'Headers' tab is selected, showing standard auto-generated headers like Cache-Control, Postman-Token, Content-Length, Host, User-Agent, Accept, Accept-Encoding, Connection, and token. The 'Body' tab shows a JSON response with a records array containing one item: { tbid: 1, category_name: 'Grocery', status: 1, created_at: '2021-12-19 23:14:19'}. The status bar at the bottom indicates '200 Ok 58 ms 385 B'.

Delete Category:

The screenshot shows the Postman application interface. The left sidebar has sections for 'Scratch Pad', 'Collections', 'APIs', 'Environments', 'Mock Servers', 'Monitors', and 'History'. The 'Scratch Pad' section lists various API endpoints. The main workspace shows a 'POST Delete Category' request. The URL is 'http://localhost:24Hrs/api/delete_category'. The 'Body' tab shows a JSON body with a tbid field set to 3. The response in the 'Body' tab shows a success status of true, an error of false, and a message 'category was deleted.'. The status bar at the bottom indicates '200 OK 74 ms 535 B'.

Create Subcategory:

The screenshot shows the Postman application interface. A POST request is made to `http://localhost:24Hrs/api/create_subcategory`. The request body contains the following JSON:

```
1 "category": "1",
2 "subcategory_name": "Vegetables"
```

The response status is 201 Created, with a response body indicating success:

```
1 "success": true,
2 "error": false,
3 "message": "Subcategory was created."
```

Update Subcategory:

The screenshot shows the Postman application interface. A POST request is made to `http://localhost:24Hrs/api/update_subcategory`. The request body contains the following JSON:

```
1 "tbid": "1",
2 "category": "1",
3 "subcategory_name": "Vegetable"
```

The response status is 200 OK, with a response body indicating success:

```
1 "success": true,
2 "error": false,
3 "message": "Subcategory was Updated."
```

Read Subcategory:

The screenshot shows the Postman application interface. A GET request is made to `http://localhost:24Hrs/api/read_subcategory`. The request includes the following headers:

KEY	VALUE
Cache-Control	no-cache
Postman-Token	<calculated when request is sent>
Host	<calculated when request is sent>
User-Agent	PostmanRuntime/7.28.4
Accept	*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive
token	fe054eb442405f7994530d00d031903b

The response status is 200 OK, with a response body containing a list of records:

```
1 "records": [
2     [
3         "tbid": 1,
4         "category": "Grocery",
5         "subcategory": "Vegetables",
6         "status": 1,
7         "created_at": "2021-12-28 12:01:23"
8     ]
9 ]
```

Read One Subcategory:

The screenshot shows the Postman application interface. A POST request is selected with the URL `http://localhost:24Hrs/api/read_one_subcategory?s=Vegetables`. The Headers tab is active, displaying the following configuration:

KEY	VALUE
Cache-Control	no-cache
Postman-Token	<calculated when request is sent>
Content-Length	0
Host	<calculated when request is sent>
User-Agent	PostmanRuntime/7.28.4
Accept	/*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive
token	fe054eb442405f7994530d00d031903b

The Body tab shows a JSON response with the following structure:

```
1
2   "success": "true",
3   "error": "false",
4   "Total Records": [
5     {
6       "tbid": 1,
7       "category": "Grocery",
8       "subcategory_name": "Vegetables",
9       "status": 1
10    }
11 ]
```

Search Subcategory:

The screenshot shows the Postman application interface. A GET request is selected with the URL `http://localhost:24Hrs/api/search_subcategory?s=Vegetables`. The Headers tab is active, displaying the following configuration:

KEY	VALUE
Cache-Control	no-cache
Postman-Token	<calculated when request is sent>
Host	<calculated when request is sent>
User-Agent	PostmanRuntime/7.28.4
Accept	/*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive
token	fe054eb442405f7994530d00d031903b

The Body tab shows a JSON response with the following structure:

```
1
2   "records": [
3     {
4       "tbid": 1,
5       "category": "Grocery",
6       "subcategory": "Vegetables",
7       "status": 1,
8       "created_at": "2021-12-20 12:01:23"
9     }
10   ]
```

Delete Subcategory:

The screenshot shows the Postman application interface. A POST request is selected with the URL `http://localhost:24Hrs/api/delete_subcategory`. The Headers tab is active, displaying the following configuration:

KEY	VALUE
tbid	2

The Body tab shows a JSON response with the following structure:

```
1
2   "success": "true",
3   "error": "false",
4   "message": "Subcategory was deleted."
```

Create Product:

The screenshot shows the Postman application interface. A POST request is made to `http://localhost:24hrs/api/create_product`. The request body is a JSON object with fields: `tbid`, `category`, `subcategory`, `product_name`, `description`, and `cost`. The response status is 201 Created, with a message indicating the product was created.

```
1 {
2   "tbid": "1",
3   "category": "1",
4   "subcategory": "1",
5   "product_name": "Big Onions",
6   "description": "Small Size Onion 1KG - Rs. 60",
7   "cost": "60"
8 }
```

```
1 {
2   "success": "true",
3   "error": "false",
4   "message": "Product was created."
5 }
```

Update Product:

The screenshot shows the Postman application interface. A POST request is made to `http://localhost:24hrs/api/update_product`. The request body is a JSON object with fields: `tbid`, `category`, `subcategory`, `product_name`, `description`, and `cost`. The response status is 200 OK, with a message indicating the product was updated.

```
1 {
2   "tbid": "1",
3   "category": "1",
4   "subcategory": "1",
5   "product_name": "Small Onions",
6   "description": "Small Size Onion 1KG - Rs. 60",
7   "cost": "60"
8 }
```

```
1 {
2   "success": "true",
3   "error": "false",
4   "message": "Product was Updated."
5 }
```

Read Product:

The screenshot shows the Postman application interface. A GET request is made to `http://localhost:24hrs/api/read_product`. The request includes various headers such as Cache-Control, Postman-Token, Host, User-Agent, Accept, Accept-Encoding, Connection, and token. The response status is 200 OK, returning a JSON array of products. The first product in the array has a store name of "ABC", tbid of 2, category of "Grocery", subcategory of "Vegetables", product name of "Big Onions", description of "Big Size Onion 1KG - Rs. 60", cost of "60", product image of `http://localhost:24hrs/`, status of 1, and a creation timestamp of `2021-12-28 12:17:33`.

KEY	VALUE
Cache-Control	no-cache
Postman-Token	<calculated when request is sent>
Host	<calculated when request is sent>
User-Agent	PostmanRuntime/7.28.4
Accept	/*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive
token	fe054eb442405f7994530d00d031903b

```
1 {
2   "records": [
3     {
4       "store_name": "ABC",
5       "tbid": 2,
6       "category": "Grocery",
7       "subcategory": "Vegetables",
8       "product_name": "Big Onions",
9       "description": "Big Size Onion 1KG - Rs. 60",
10      "cost": "60",
11      "product_image": "http://localhost:24hrs/",
12      "status": 1,
13      "created_at": "2021-12-28 12:17:33"
14    },
15    {
16      "store_name": "ABC",
17      "tbid": 1,
18      "category": "Grocery",
19      "subcategory": "Vegetables",
20      "product_name": "Small Onion",
21      "description": "Small Size Onion 1KG - Rs. 80",
22      "cost": "80"
23    }
24  ]
25 }
```

Read One Product:

The screenshot shows the Postman application interface. At the top, there's a navigation bar with 'File', 'Edit', 'View', 'Help', 'Home', 'Workspaces', 'Reports', and 'Explore'. A search bar says 'Search Postman'. On the right, there are icons for 'Sign In' and 'Create Account'. Below the navigation is a toolbar with buttons for 'POST Create Product', 'POST Update Product', 'GET Read Product', 'POST Read One Product' (which is selected), and other options like '+', '...', and 'No Environment'. The main area has tabs for '24Hrs localhost' and 'Read One Product'. Under 'Read One Product', it says 'POST http://localhost:24Hrs/api/read_one_product?product_name=Big+Onions'. There are dropdowns for 'Save' and '...' on the right. Below this, there's a table for 'Params', 'Auth', 'Headers (9)', 'Body', 'Pre-req.', 'Tests', and 'Settings'. The 'Headers' section is expanded, showing fields like 'Cache-Control', 'Postman-Token', 'Content-Length', 'Host', 'User-Agent', 'Accept', 'Accept-Encoding', 'Connection', and 'token'. The 'Body' section is set to 'JSON' and contains a JSON response:

```
1 "success": "true",
2 "error": "false",
3 "Total Records": {
4     "store_name": "ABC",
5     "tbid": 2,
6     "category": "Grocery",
7     "listings": "Vegetables",
8     "product_name": "Big Onions",
9     "description": "Big Size Onion 1KG - Rs. 60",
10    "cost": "60"
11    "product_image": "http://localhost:24Hrs/",
12    "status": 1,
13    "created_at": "2021-12-28 12:17:33"
14}
15
16
```

At the bottom, there are buttons for 'Send', '200 OK', '33 ms', '701 B', and 'Save Response'. The status bar at the bottom says 'Activate Windows Go to Settings to activate Windows.' The taskbar at the very bottom shows various application icons.

Search Product:

The screenshot shows the Postman application interface. The top navigation bar includes File, Edit, View, Help, Home, Workspaces, Reports, Explore, and a search bar. Below the navigation is a toolbar with icons for Create Product, Update Product, Read Product, Read One Product, Search Product, and others. A status bar at the bottom indicates "24Hrs localhost / Search Product", "200 OK 15 ms 539 B", and "Save Response".

The main workspace displays a GET request to `http://localhost:24Hrs/api/search_product?name=Small+Onion`. The response body is shown as JSON:

```
        "records": [
            {
                "store_name": "ABC",
                "tid": 1,
                "category": "Grocery",
                "subcategory": "Vegetables",
                "product_name": "Small Onion",
                "product_desc": "Small Size Onion 1kg - Rs. 00",
                "cost": "00",
                "product_image": null,
                "status": 1,
                "created_at": "2021-12-28 12:14:34"
            }
        ]
```

The Headers section shows the following configuration:

Key	Value
Cache-Control	no-cache
Postman-Token	<calculated when request is sent>
Host	<calculated when request is sent>
User-Agent	PostmanRuntime/7.28.4
Accept	*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive
token	fe054eb442405f7994530d0dd031903b

Delete Product:

The screenshot shows the Postman application interface. The top navigation bar includes 'File', 'Edit', 'View', 'Help', 'Home', 'Workspaces', 'Reports', 'Explore', 'Search Postman', 'Sign In', and 'Create Account'. Below the navigation is a toolbar with icons for 'Create Product', 'Update Product', 'Read Product', 'Read One Prod...', 'Search Product', 'Delete Product', and environment management. The main workspace shows a '24Hrs localhost / Delete Product' collection. A 'POST' request is selected with the URL 'http://localhost/24hrs/api/delete_product'. The 'Body' tab is active, containing raw JSON data: { "tbid": "2" }. The response pane shows a 200 OK status with a response time of 99 ms and a size of 534 B. The response body is: { "success": "true", "error": "false", "message": "Product was deleted." }. The bottom of the screen features a taskbar with various application icons and system status indicators.

Create Offer:

The screenshot shows the Postman application interface. A POST request is being made to `http://localhost:24Hrs/api/create_offer`. The request body is a JSON object representing an offer. The response status is 201 Created, with a message indicating the offer was created.

```
1 {
2   "tbid": "1",
3   "category": "1",
4   "subcategory": "1",
5   "product": "4",
6   "description": "1Kg Small onion Rs.60",
7   "offer": "5",
8   "other_offer": "Buy 1 Get 1 Free",
9   "total_cost": "60",
10  "offer_price": "55",
11  "offer_time": "2 Hrs"
12 }
```

```
1 {
2   "success": "true",
3   "error": "false",
4   "message": "Offer was created."
5 }
```

Update Offer:

The screenshot shows the Postman application interface. A POST request is being made to `http://localhost:24hrs/api/update_offer`. The request body is identical to the one in the 'Create Offer' section. The response status is 200 OK, with a message indicating the offer was updated.

```
1 {
2   "tbid": "1",
3   "category": "1",
4   "subcategory": "1",
5   "product": "4",
6   "description": "1Kg Small onion Rs.60",
7   "offer": "5",
8   "other_offer": "Buy 1 Get 1 Free",
9   "total_cost": "60",
10  "offer_price": "55",
11  "offer_time": "1 Hrs"
12 }
```

```
1 {
2   "success": "true",
3   "error": "false",
4   "message": "Offer was updated."
5 }
```

Read Offer:

The screenshot shows the Postman application interface. A GET request is being made to `http://localhost:24Hrs/api/read_offer`. The request includes various headers such as Cache-Control, Host, User-Agent, Accept, Accept-Encoding, Connection, and token. The response status is 200 OK, with a message indicating the offer was retrieved.

KEY	VALUE
Cache-Control	no-cache
Postman-Token	<calculated when request is sent>
Host	<calculated when request is sent>
User-Agent	PostmanRuntime/7.28.4
Accept	*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive
token	fe054eb42405f7904530d00d031903b

```
1 {
2   "records": [
3     {
4       "tbid": 1,
5       "store_name": "ABC",
6       "category": "Grocery",
7       "subcategory": "Vegetables",
8       "product": "Small Onion",
9       "description": "1Kg Small onion Rs.60",
10      "offer": "5",
11      "other_offer": "Buy 1 Get 1 Free",
12      "total_cost": "60",
13      "offer_price": "55",
14      "offer_time": "1 Hrs",
15      "status": 1,
16      "created_at": "2021-12-20 12:39:00"
17    }
18  ]
19 }
```

Delete Offer:

Postman screenshot showing a successful DELETE request to `/api/delete_offer`. The response body is:

```
1
2   "success": "true",
3   "error": "false",
4   "message": "Offer was deleted."
```

List Unit:

Postman screenshot showing a successful GET request to `/api/read_unit`. The response body is:

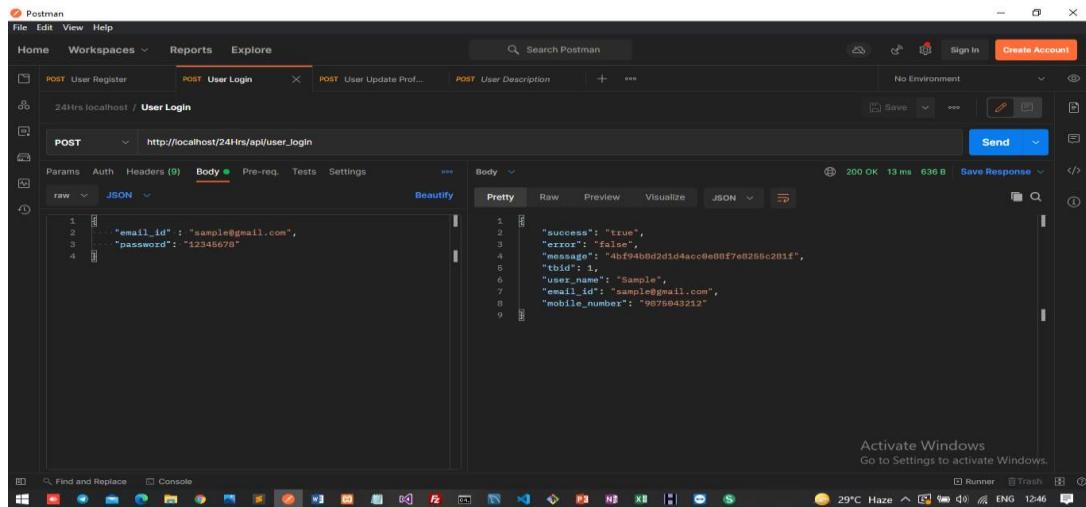
```
1
2   "records": [
3     {
4       "tbid": 2,
5       "unit_name": "liter",
6       "unit": "ltr",
7       "status": 1,
8       "created_at": "2021-12-12 17:34:46"
9     },
10    {
11      "tbid": 3,
12      "unit_name": "milliliters",
13      "unit": "ml",
14      "status": 1,
15      "created_at": "2021-12-12 17:34:54"
16    },
17    {
18      "tbid": 4,
19      "unit_name": "Milligram",
20      "unit": "mg",
21      "status": 1,
22      "created_at": "2021-12-12 17:34:54"
23    }
24  ]
```

User Register:

Postman screenshot showing a successful POST request to `/api/user_register`. The response body is:

```
1
2   "success": "true",
3   "error": "false",
4   "token": "4bf94bd2d1d4acc0e00f7e0258c281f",
5   "message": "Registered Successfully"
```

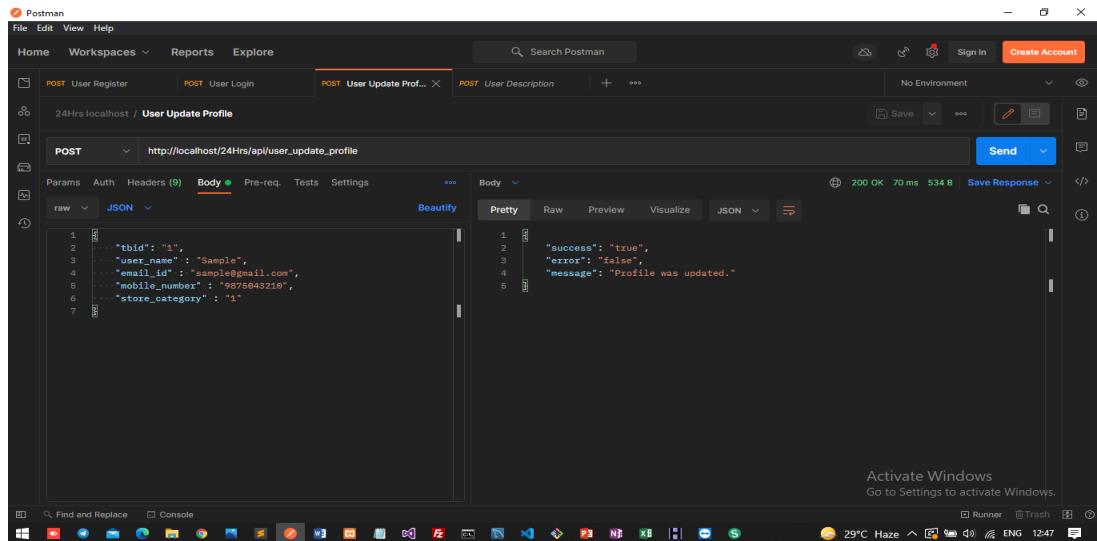
User Login:



The screenshot shows the Postman application interface. A POST request is made to `http://localhost:24Hrs/api/user_login`. The request body is JSON, containing `{"email_id": "sample@gmail.com", "password": "12345678"}`. The response is a 200 OK status with a response time of 13 ms and a size of 636 B. The response body is:

```
1 |   "success": "true",
2 |   "error": "false",
3 |   "message": "4b94b8d2d4acc0e8ff7e8256c201f",
4 |   "tbid": 1,
5 |   "user_name": "Sample",
6 |   "email_id": "sample@gmail.com",
7 |   "mobile_number": "9876043210"
```

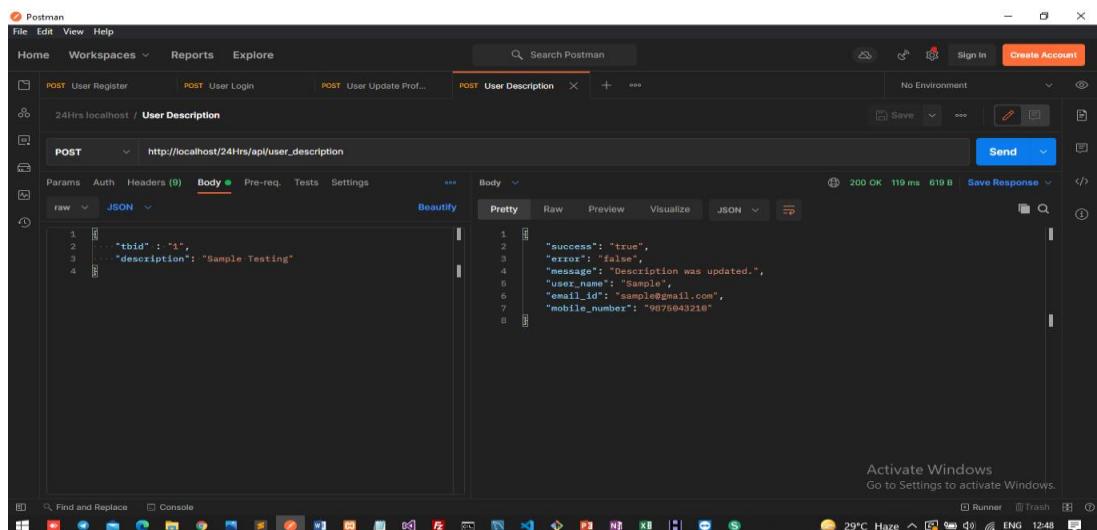
User Update Profile:



The screenshot shows the Postman application interface. A POST request is made to `http://localhost:24Hrs/api/user_update_profile`. The request body is JSON, containing `{"tbid": "1", "user_name": "Sample", "email_id": "sample@gmail.com", "mobile_number": "9876043210", "store_category": "1"}`. The response is a 200 OK status with a response time of 70 ms and a size of 534 B. The response body is:

```
1 |   "success": "true",
2 |   "error": "false",
3 |   "message": "Profile was updated."
```

User Description:



The screenshot shows the Postman application interface. A POST request is made to `http://localhost:24Hrs/api/user_description`. The request body is JSON, containing `{"tbid": "1", "description": "Sample Testing"}`. The response is a 200 OK status with a response time of 119 ms and a size of 619 B. The response body is:

```
1 |   "success": "true",
2 |   "error": "false",
3 |   "message": "Description was updated.",
4 |   "user_name": "Sample",
5 |   "email_id": "sample@gmail.com",
6 |   "mobile_number": "9876043210"
```

Product List Based On Store Category:

The screenshot shows the Postman application interface. At the top, there are tabs for Home, Workspaces, Reports, and Explore. Below the tabs, there are two requests: a GET request to 'List Product' and a POST request to 'User Login'. The GET request is selected. The URL in the request field is `http://localhost:24Hrs/api/list_product`. The response status is 200 OK, with a response time of 33 ms and a size of 812 B. The response body is displayed in JSON format, showing a success message and a list of product data. The product data includes fields like id, store_name, category, subcategory, offer, cost, and offer_time.

```
2 "success": "true",
3 "error": "false",
4 "data": [
5     {
6         "id": 3,
7         "store_name": "1",
8         "category": "Grocery",
9         "subcategory": "Vegetables",
10        "offer": "Buy 1 Get 1 Free",
11        "cost": "80",
12        "offer": "0",
13        "other_offer": "Buy 1 Get 1 Free",
14        "offer_time": "00:00:00",
15        "offer_time": "00:00:00",
16        "offer_time": "00:00:00",
17        "offer_time": "1 Hrs",
18        "status": 1,
19        "created_at": "2022-12-20 12:26:24"
20    }
21 ]
22 }
```

Seller APP API

- **Seller**
 1. Seller Register - (POST)
 2. Seller Login - (POST)
 3. Seller Update - (POST)
 4. Seller Contact - (POST)
 5. Seller Description - (POST)
- **Category**
 1. Create Category - Token (POST)
 2. Update Category - Token (POST)
 3. Read Category - Token (GET)
 4. Read One Category - Token (POST)
 5. Search Category - Token (GET)
 6. Delete Category - (POST)
- **Subcategory**
 1. Create Subcategory - Token (POST)
 2. Update Subcategory - Token (POST)
 3. Read Subcategory - Token (GET)
 4. Read One Subcategory - Token (POST)
 5. Search Subcategory - Token (GET)
 6. Delete Subcategory - (POST)
- **Product**
 1. Create Product - Token (POST)
 2. Update Product - Token (POST)
 3. Read Product - Token (GET)
 4. Read One Product - Token (POST)
 5. Search Product - Token (GET)
 6. Delete Product - (POST)
- **Offer**
 1. Create Product - Token (POST)
 2. Update Product - Token (POST)
 3. Read Product - Token (GET)
 4. Delete Product - (POST)
- **Read Unit - GET**
- **List store – GET**

USER APP API

- **User**
 1. User Register - (POST)
 2. User Login - (POST)
 3. User Update - (POST)
 4. User Description - (POST)

- **List Based on User Store Category**
 1. List Product - (GET)
 2. List Offer - (GET)
- **List Category** - Token (GET)