

## Steps

```
docker-compose build
docker-compose up -d
docker-compose logs -f
docker exec -it cafe-app bash -c "sudo -u devuser /bin/bash"
composer install
php artisan migrate:install
```

```
php artisan make:migration create_table_drink --create=drink
php artisan make:migration create_table_consumer --create=consumer
php artisan make:migration create_table_consumption --create=consumption
```

```
php artisan migrate
php artisan make:model Consumption
php artisan make:controller Api/ConsumptionController
```

```
php artisan make:model Consumer
php artisan make:controller Api/ConsumerController
```

```
php artisan tinker
```

```
$user = new \App\User();
$user->name = "Gustavo";
$user->email = 'gustavo@caffe.com';
$user->password = bcrypt('12356');
$user->save();
```

```
php artisan make:migration add_unique_to_consumer_table --table=consumer
php artisan make:migration add_unique_to_consumption_table --table=consumption
php artisan make:migration add_unique_to_drink_table --table=drink
```

```
php artisan make:request ConsumerRequest
```

```
php artisan make:model Drink
php artisan make:controller Api/DrinkController
```

```
php artisan make:request DrinkRequest
php artisan make:controller Api/UserController --resource --api
```

```
php artisan make:request UserRequest
php artisan make:request ConsumptionRequest
composer require tyson/jwt-auth:1.0.* --prefer-source
```

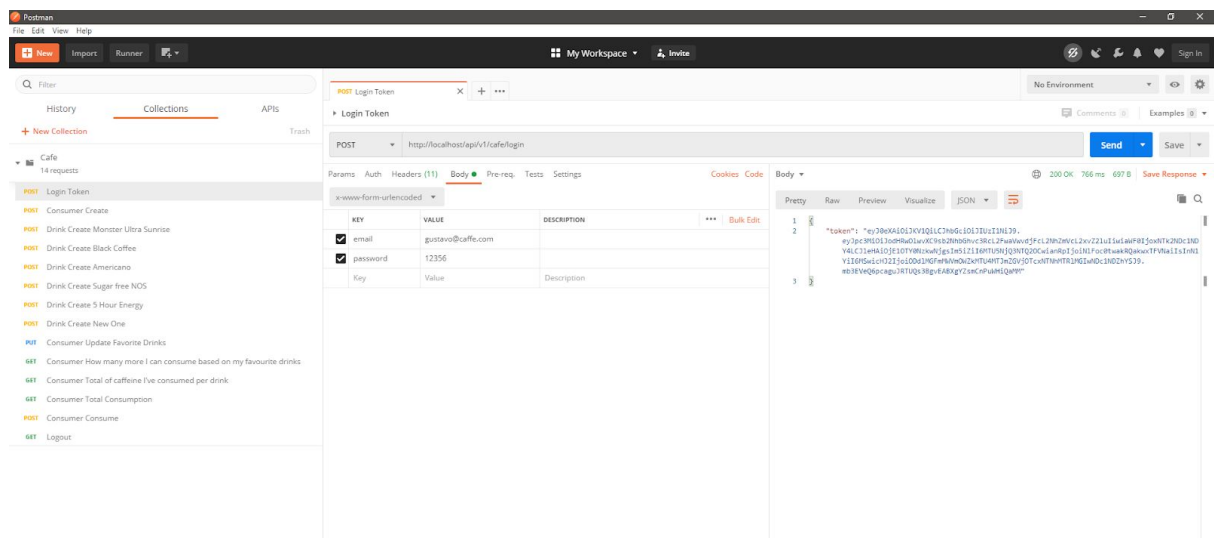
```
php artisan vendor:publish --provider="Tymon\JWTAuth\Providers\LaravelServiceProvider"
php artisan jwt:secret
php artisan make:controller Api/Auth/LoginJwtController
php artisan make:migration create_table_consumer_drink_favorite
--create=consumer_drink_favorite
```

```
php artisan migrate
php artisan make:model ConsumerDrinkFavorite
```

## Tests on POSTMAN

1. Login
2. Consumer create
3. Drink Create Monster Ultra Sunrise
4. Drink Create Black Coffee
5. Drink Create Americano
6. Drink Create Sugar free NOS
7. Drink Create 5 Hour Energy
8. Drink Create New One
9. Consumer Update Favorite Drinks
10. Consumer How many more I can consume based on my favourite drinks
11. Consumer Total of caffeine I've consumed per drink
12. Consumer Total Consumption
13. Consumer Consume
14. Logout

## Login Request



## EDIT COLLECTION



Name

Cafe

Description

Authorization ●

Pre-request Scripts

Tests

Variables

This authorization method will be used for every request in this collection. You can override this by specifying one in the request.

TYPE

Bearer Token ▾

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

! Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. [Learn more about variables](#)

Token

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJp
c3MiOiJodHRwOlwwXC9sb2NhOGhvc3RcL2
FwaVwvdjFjL2NhZmVcL2xvZ2luliwiaWF0Ijo
xNTk2NDc1NDY4LCJleHAiOiJlOTY0NzkwN
jgsIm5iZil6MTU5NjQ3NTQ2OCwianRpljoiNI
Foc0twakRQakwxTFVNailsInN1Yil6MSwicHJ
2ljoiodDlMGFmMWVmOWZkMTU4MTJmZ
GVjOTcxNTNhMTRIMGlwNDc1NDZhYSJ9.m
b3EVeQ6pcaguJRTUQs3BgvEABXgYZsmCn
PuWHiQaMM
```

Cancel

Update

# Consumer Create

Postman interface showing the 'Consumer Create' API endpoint. The request is a POST to `http://localhost:api/v1/cafe/consumers` with a JSON body containing `user_id` (1) and `consumption_limit` (500). The response is a 200 OK status with a JSON body indicating 'Consumer was registered'.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> user_id	1	
<input checked="" type="checkbox"/> consumption_limit	500	
Key	Value	Description

```
1 {
2   "data": {
3     "message": "Consumer was registered"
4   }
5 }
```

# Drink Create Monster Ultra Sunrise

Postman interface showing the 'Drink Create Monster Ultra Sunrise' API endpoint. The request is a POST to `http://localhost:api/v1/cafe/drinks` with a JSON body containing `name` (Monster Ultra Sunrise), `description` (A refreshing drink), `caffeine` (75), and `plug` (123). The response is a 200 OK status with a JSON body indicating 'Drink was registered'.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> name	Monster Ultra Sunrise	
<input checked="" type="checkbox"/> description	A refreshing drink	
<input checked="" type="checkbox"/> caffeine	75	
<input checked="" type="checkbox"/> plug	123	
Key	Value	Description

```
1 {
2   "data": {
3     "message": "Drink was registered"
4   }
5 }
```

# Drink Create Black Coffee

Postman interface showing the 'Drink Create Black Coffee' API endpoint. The request is a POST to `http://localhost:api/v1/cafe/drinks` with a JSON body containing `name` (Black Coffee), `description` (The classic, the best), `caffeine` (95), and `plug` (123). The response is a 200 OK status with a JSON body indicating 'Drink was registered'.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> name	Black Coffee	
<input checked="" type="checkbox"/> description	The classic, the best	
<input checked="" type="checkbox"/> caffeine	95	
<input checked="" type="checkbox"/> plug	123	
Key	Value	Description

```
1 {
2   "data": {
3     "message": "Drink was registered"
4   }
5 }
```

## Drink Create Americano

The screenshot shows the Postman application with a collection named 'Cafe' containing 14 requests. The 'Drink Create Americano' request is selected. The request is a POST to 'http://localhost:4040/api/v1/cafe/drinks'. The body is a JSON object: 

```
{  "name": "Americano",  "description": "Sometimes you n",  "caffeine": 77,  "slug": 123}
```

. The response is a 200 OK status with a JSON body: 

```
{  "data": {    "message": "Drink was registered"  }}
```

.

## Drink Create Sugar free NOS

The screenshot shows the Postman application with the 'Drink Create Sugar free NOS' request selected. The request is a POST to 'http://localhost:4040/api/v1/cafe/drinks'. The body is a JSON object: 

```
{  "name": "Sugar free NOS",  "description": "Another orange d",  "caffeine": 130,  "slug": 123}
```

. The response is a 200 OK status with a JSON body: 

```
{  "data": {    "message": "Drink was registered"  }}
```

.

## Drink Create 5 Hour Energy

The screenshot shows the Postman application with the 'Drink Create 5 Hour Energy' request selected. The request is a POST to 'http://localhost:4040/api/v1/cafe/drinks'. The body is a JSON object: 

```
{  "name": "5 Hour Energy",  "description": "And amazing shor",  "caffeine": 200,  "slug": 123}
```

. The response is a 200 OK status with a JSON body: 

```
{  "data": {    "message": "Drink was registered"  }}
```

.

## Drink Create New One

The screenshot shows the Postman application with a workspace named "My Workspace". The "Collections" tab is active, showing a collection named "Cafe" with 14 requests. The selected request is "Drink Create New One", which is a POST request to the URL `http://localhost:3001/api/v1/cafe/drinks`. The request body is a JSON object with the following fields:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> name	New One	
<input checked="" type="checkbox"/> description	New One	
<input checked="" type="checkbox"/> caffeine	200	
<input checked="" type="checkbox"/> slug	123	
Key	Value	Description

The response is a JSON object:

```
{  "data": {    "message": "Drink was registered"  }}
```

## Consumer Update Favorite Drinks

The screenshot shows the Postman application with a workspace named "My Workspace". The "Collections" tab is active, showing a collection named "Cafe" with 14 requests. The selected request is "Consumer Update Favorite Drinks", which is a PUT request to the URL `http://localhost:3001/api/v1/cafe/consumers/1`. The request body is a JSON object with the following fields:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> user_id	1	
<input checked="" type="checkbox"/> consumption_limit	500	
<input checked="" type="checkbox"/> favorite_drinks[]	1	
<input checked="" type="checkbox"/> favorite_drinks[]	2	
<input checked="" type="checkbox"/> favorite_drinks[]	3	
<input checked="" type="checkbox"/> favorite_drinks[]	4	
<input checked="" type="checkbox"/> favorite_drinks[]	5	
Key	Value	Description

The response is a JSON object:

```
{  "data": {    "message": "Consumer updated favorite drinks"  }}
```

## Consumer How many more I can consume based on my favourite drinks

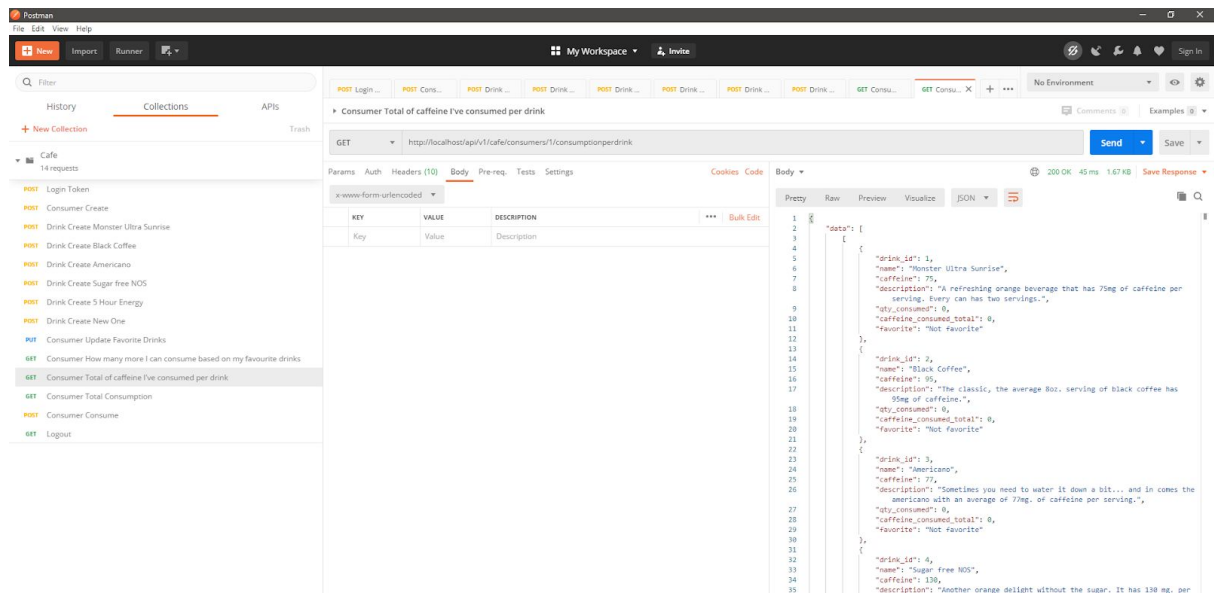
The screenshot shows the Postman application with a workspace named "My Workspace". The "Collections" tab is active, showing a collection named "Cafe" with 14 requests. The selected request is "Consumer How many more I can consume based on my favourite drinks", which is a GET request to the URL `http://localhost:3001/api/v1/cafe/consumers/1/qtyallowedperdrink`. The response is a JSON object with the following fields:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> user_id	1	
<input checked="" type="checkbox"/> consumption_limit	500	
<input checked="" type="checkbox"/> favorite_drinks[]	1	
<input checked="" type="checkbox"/> favorite_drinks[]	2	
<input checked="" type="checkbox"/> favorite_drinks[]	3	
<input checked="" type="checkbox"/> favorite_drinks[]	4	
<input checked="" type="checkbox"/> favorite_drinks[]	5	
Key	Value	Description

The response is a JSON object:

```
{  "data": {    "message": "Consumer can consume 100 more drinks based on their favourite drinks"  }}
```

## Consumer Total of caffeine I've consumed per drink

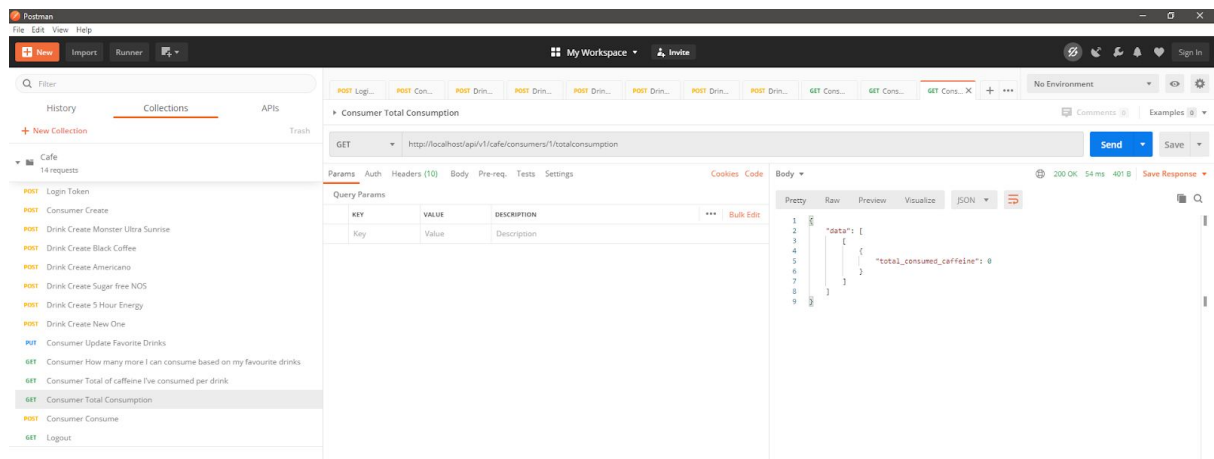


Postman interface showing a GET request to `http://localhost:3001/v1/caffe/consumers/1/consumptionperdrink`. The response is a JSON array of drink objects, each containing details like `drink_id`, `name`, `caffeine`, `description`, `qty_consumed`, `caffeine_consumed_total`, and `favorite`.

KEY	VALUE	DESCRIPTION
Key	Value	Description

```
1 {
2   "data": [
3     {
4       "drink_id": 1,
5       "name": "Monster Ultra Sunrise",
6       "caffeine": 75,
7       "description": "A refreshing orange beverage that has 75mg of caffeine per serving. Every can has two servings.",
8       "qty_consumed": 0,
9       "caffeine_consumed_total": 0,
10      "favorite": "Not Favorite"
11    },
12    {
13      "drink_id": 2,
14      "name": "Black Coffee",
15      "caffeine": 95,
16      "description": "The classic, the average 8oz. serving of black coffee has 95mg of caffeine.",
17      "qty_consumed": 0,
18      "caffeine_consumed_total": 0,
19      "favorite": "Not Favorite"
20    },
21    {
22      "drink_id": 3,
23      "name": "Americano",
24      "caffeine": 77,
25      "description": "Sometimes you need to water it down a bit... and in comes the americano with an average of 77mg. of caffeine per serving.",
26      "qty_consumed": 0,
27      "caffeine_consumed_total": 0,
28      "favorite": "Not Favorite"
29    },
30    {
31      "drink_id": 4,
32      "name": "Sugar free NOS",
33      "caffeine": 130,
34      "description": "Another orange delight without the sugar. It has 130 mg. per"
35    }
36  ]
37 }
```

## Consumer Total Consumption

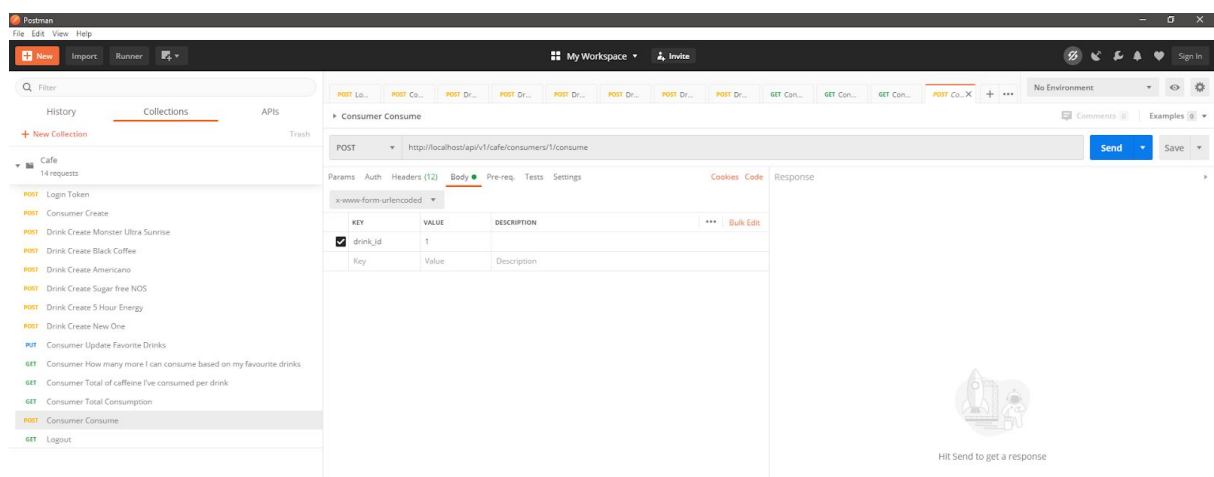


Postman interface showing a GET request to `http://localhost:3001/v1/caffe/consumers/1/totalconsumption`. The response is a JSON object with a single key `total_consumed_caffeine` and a value of 0.

KEY	VALUE	DESCRIPTION
Key	Value	Description

```
1 {
2   "data": {
3     "total_consumed_caffeine": 0
4   }
5 }
```

## Consumer Consume



Postman interface showing a POST request to `http://localhost:3001/v1/caffe/consumers/1/consume`. The request body is a JSON object with `drink_id` and `qty`.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> drink_id	1	
Key	Value	Description

Response: Hit Send to get a response



# Logout

Postman

File Edit View Help

New Import Runner

My Workspace Invite

Filter

History Collections APIs

+ New Collection

Trash

Cafe

14 requests

- POST Login Token
- POST Consumer Create
- POST Drink Create Monster Ultra Sunrise
- POST Drink Create Black Coffee
- POST Drink Create Americano
- POST Drink Create Sugar free NOS
- POST Drink Create 5 Hour Energy
- POST Drink Create New One
- POST Consumer Update Favorite Drinks
- GET Consumer How many more I can consume based on my favourite drinks
- GET Consumer Total of caffeine I've consumed per drink
- GET Consumer Total Consumption
- POST Consumer Consume
- GET Logout

Logout

GET http://localhost:api/v1/cafe/logout

Send Save

Params Auth Headers (10) Body Pre-req. Tests Settings

x-www-form-urlencoded

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body

200 OK 63 ms 369 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {}
2 "Logout"
3 {}
```