Restful Api Customer ManagementSystem

The service maintain a list of customers with some information fields about them .
The customer fields are first_name , last_name , email, created_at (Registration date), updated_at . These fields are stored in the customers table of database restful_api_php.
(You can find the database.sql file inside the Database folder)
This Api is not public . The Users needs to be registered and authenticated to access the Api resources . The authentication is done using JWT .
This Api server does not keep any client state . No use of sessions or cookies .
So , the server replies to each user request as if it was the first request the client has made .
The customers data fields will be json encoded and represented to the frontend in the form of json arrays , so the output response of the client requests will be a json format .

Project Directory

The tools Apache , PHP, and Mysql are set up in Windows system .
Under the server's htdocs folder was created a project root directory called restful_api_php .

Base URL

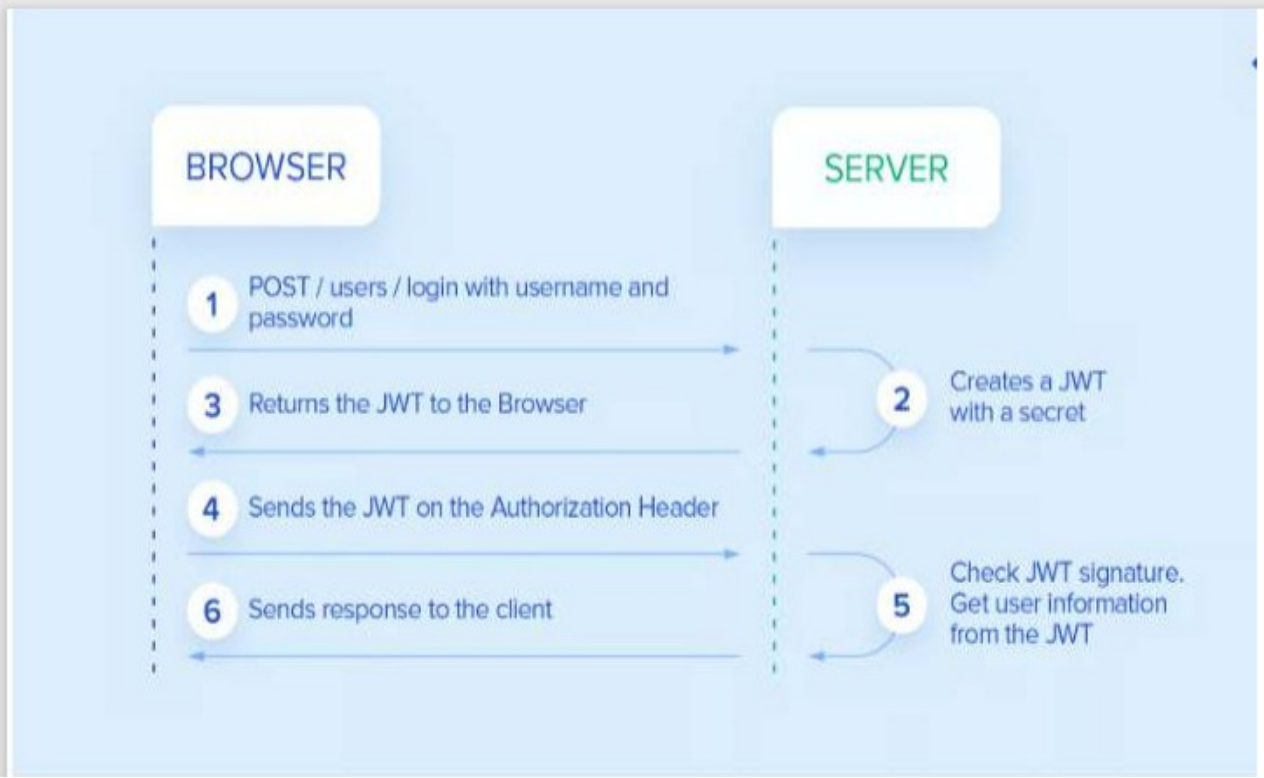
The base URL for all API requests is : http://localhost/restful_api_php/

User Registration

In order to gain access to the system , the user has to first get himself/herself registered by providing the name , email and password informations . For simplicity was not created the registration user part in the system . These fields name , email and password are inserted randomly manually in the user table database . No encryption/decryption /md5 or other technologies for storing the password are used . The user password is stored as a plain text for semplicity. The users that are not registered can not access the system . Once the users are registered they are authenticated to access the system through Json Web Token authentication .

Json Web Token Authentication

The JWT is supposed to work following this schema



1) Users logs in by sending their credentials to the Identity provider . The file Auth/ JWTServiceProvider.php plays the role of an Identity Provider for this Api .

2) The JWTServiceProvider.php verifies the credentials of a registered user .
If the user is registered in the database system , it retrieves the user data and generates a
The JWT .
The JWT contains 3 parts : First part is the Header . Header contains the information about the algorithm that is used, such as HS256 .
Second part is the payload . The payload contains the claims. There is a set of registered claims, : iss (localhost), exp (expiration time), ect , but also the payload can include extra attributes that define custom claims, such as userId in this Api case .

The third part is the Signature.
To create the signature part, the encoded payload is signed by using the

signature algorithm from the header. The signature is used to verify that the issuer of the JWT is who it says it is and to ensure that the message wasn't changed along the way
A same SECRETE_KEY is used for the encode/decode of the payload .
The function generateToken of class JWTServiceProvider generates the token and it also sets the expiration on the JWT for a specific time . In this Api case the expiration time is set for 1 hour . (please make reference to the function generateToken() , and $expirationTime = 60 x 60 variable) . After one hour the token expires and it is not more valid .

3) JWTServiceProvider encrypts the JWT using the algorithm 'HS256' and sends it to the client as a response to the initial request with credentials.

4) Client stores the JWT for a limited or not limited amount of time, depending on the expiration time set by the identity provider.
Client sends the stored JWT in an Authorization header for every request to the server

5) For each request, the JWTServiceProvider takes the JWT from the Authorization header and decrypts it, validates the signature, and if everything is OK, extracts from the payload part the userId data .
Based only on the userId data the JWTServiceProvider can accept or deny the client request.

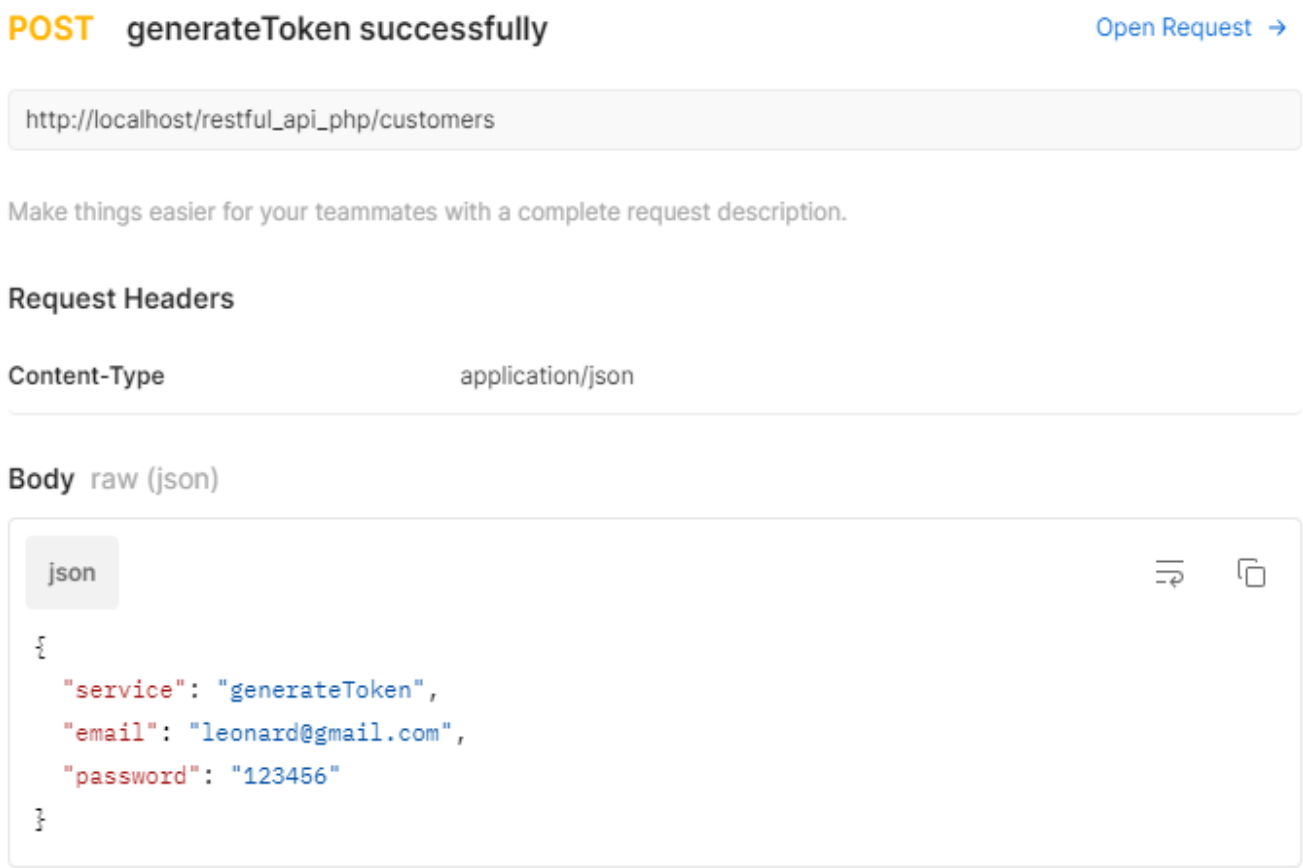
JWT Generation

We will access the generateToken function to generate the JWT Api . This function will be accessed by making e a post request to the endpoint . Also is necessary to install through composer the library firebase/php-jwt": "^5.0.0 "

```
"require": {
        "firebase/php-jwt": "^5.0.0 ",
    },
```

The Post request to http://localhost/restful_api_php/customers is done usually for creating a new customer in database but we are going to use it also for the generation Token .
So a post request can have 2 purposes .
The first purpose is to create a new customer and the second to generate token .
In order to make the system understand that we are going to generate a token with our request we must specify the content- type application/json , and the input field 'service' with value 'generateToken' .
Also is necessary to insert the user credentials (email and password fields) .

In this case by setting the input field service and input fields email and password the system will redirect the post request to generate the token and not to create a new Customer . (Please make reference to this part in file CustomerController.pdf and function processPostRequest) .The reported Postman figure shows the generated token for a register user with valid credentials in database . In case the post request for token generation is successfully , then the output response will be a json array with a key named 'response' having for value an array with 2 elements .
The first element is a key called 'statusCode' with value an integer and the second element a key 'result' with value an array composed with only one element (a key named 'token' and having for value a string)



generateToken successfully


```
{
  "response": {
    "statusCode": 200,
    "result": {
      "token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE2OTE0MTEyNDEsImlzcyI6ImxvY2FsaG9zdCIsImV4cCI6MTY5N
    }
  }
}
```


In case the user is not registered in database the system will not generate the token and will throw an error 'Email or password incorrect' with status code '401 Unauthorized' .
The figure reported below shows this case with a user non existent in database .
The output will be a json array with key 'error' and value an array of elements with key 'statusCode' having for value an integer and key 'message' with value a string data type.

Request Headers

Content-Type	application/json

Body raw (json)


```
{
  "service": "generateToken",
  "email": "beatrice@gmail.com",
  "password": "4785123456"
}
```

Response

Body Headers


```
{
  "error": {
    "statusCode": 401,
    "message": "Email or Password incorrect"
  }
}
```


The client will use this token for each of the (GET, POST, PUT, PATCH ,DELETE) request . First the development of the Api was done using Postman tool as an HTTP client that tests HTTP requests . So with postman we are simulating a client which send an authorized request to the system . To enable requests with authorization for all the HTTP methods , is necessary to set the Authorization in Postman Headers with the Bearer value the value of the token generated previously. A variable 'jwt_token' was set globaly to be recognized by all the HTTP request with authorization .

GET http://localhost/restful_api_php/customers Send

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

Type Bearer Token

Token {{jwt_token}}

The authorization header will be automatically generated when you send the request. Learn more about authorization ↗

RESTful API endpoints

The RESTful API implementation with have the following endpoints:

All the HTTP requests methods may have a successfully output result or may end with generating an error as output .

Below reported are presented cases when the request has a successful response and when it has error responses .

GET /customers

Input parameters : None

case : getRequest Successfully

The response is a json array with a key named 'response' having as value an array of 2 elements.
The first element is a key named 'statusCode' with value an integer , and the second element is key 'result' having for value an array of customers object data fields that are present in customers table .

- statusCode: The HTTP header code .
- result: result is a key having for value an array of customers .

  Each element of this array will have the following properties:
  - id: The unique identifier of the customer.
  - first_name: The first_name of the customer.
  - last_name: The Last name of the customer .
  - email: Email of the customer
  - created_at: Registration date .
  - updated_at: Update date

The id field data type is an integer , is the primary key of the customer table
The first_name field data type is a string ,
The last_name field is a string
The created_at field data type is a Date in format (Y-m-d)
The updated_at field data type is a Date in format (Y-m-d)

In case the request was processed correctly the output will be a json array
with parameters a key named 'response' having for value an array with 2 elements . The first element is a key called 'statusCode with a value integer data type ' , and the second element is key named 'result' with value an array of customers objects .


```
{
 "response": {
   "statusCode": 200,
   "result": [
     {
       "id": "1",
       "first_name": "Alessandro",
       "last_name": "Tara",
       "email": "alessandrotara@gmail.com",
       "created_at": "2023-07-29",
       "updated_at": "2023-07-29"
     },
     {
       "id": "2",
       "first_name": "Francesco",
       "last_name": "Tarao",
       "email": "alessandrotara@gmail.com",
       "created_at": "2023-07-29",
       "updated_at": "2023-08-07"
     },
     {
       "id": "3",
       "first_name": "Mateo",
       "last_name": "Fornara",
       "email": "mateofornara@gmail.com",
       "created_at": "2023-07-29",
       "updated_at": "2023-08-07"
     },
     {
       "id": "4",
       "first_name": "Helene",
       "last_name": "Fischer",
       "email": "helenefischer@gmail.com",
```

"created_at": "2023-07-29",
"updated_at": "2023-07-31"
},
{
"id": "6",
"first_name": "Aurora",
"last_name": "Gaze",
"email": "auroragaze@gmail.com",
"created_at": "2023-07-31",
"updated_at": "0000-00-00"
},
{
"id": "7",
"first_name": "Klea",
"last_name": "Tare",
"email": "kleatare@gmail.com",
"created_at": "2023-07-31",
"updated_at": "0000-00-00"
},
{
"id": "8",
"first_name": "Antonetta",
"last_name": "Carra",
"email": "antonettacarra@gmail.com",
"created_at": "2023-07-31",
"updated_at": "0000-00-00"
},
,,,,,,, until the last customer object present in database

case : getRequest generated with an invalid token

In case GET request will be generated by an invalid token the output will be a json array . with a key named 'error' having for value an array with 2 elements .
The first element is a key called 'statusCode' with value an integer and the second element a key 'message' with value a string . The figure belows shows the error generation of the getRequest with an invalid token .

JWT_AUTHORIZATION_RESTFUL_API_... / getAllCustomers with invalid token error gener... / getAllCustomers with invalid token error gener...

GET http://localhost/restful_api_php/customers

Params Headers (1) Body

Query Params

	KEY	VALUE	DESCRIPTION
☐	authorization		
	Key	Value	Description

Body Headers (11) Status Code

Pretty Raw Preview JSON


```
{
    "error": {
        "statusCode": 498,
        "message": "Signature verification failed"
    }
}
```

Case: getRequest generated with an expired token

In case the token expires the system will generate an error and the output of the get response will be as showed below ,

JWT_AUTHORIZATION_RESTFUL_API... / getAllCustomers with token expired error gene... / **getAllCustomers with token expired error gene...** Save

GET http://localhost/restful_api_php/customers

Params Headers (1) Body

Query Params

	KEY	VALUE	DESCRIPTION	ooo
☐	authorization			
	Key	Value	Description	

Body Headers (11) Status Code 498 INVALID_TOKEN

Pretty Raw Preview JSON

```
1 {
2     "error": {
3         "statusCode": 498,
4         "message": "Expired token"
5     }
6 }
```

GET /customers/{id}

Input parameters : customer Id

The customer Id should be an integer as data type and must also exist in customers table , otherwise the system will generate an error .

Case: getCustomer request successfully

The system returns a JSON array with a key named 'response' having for value an array of 2 elements.
The first element is a key named 'statusCode' with value an integer , and the second element is key 'result' having as value the customer object data fields found with that specific id .

- statusCode: The HTTP header code .
- result: An array of customer objects, each element of the array has the following properties:
    - id: The unique identifier of the customer.
    - first_name: The first_name of the customer.
    - last_name: The Last name of the customer .
    - email: Email of the customer
    - created_at: Registration date .
    - updated_at: Update date

The id field data type is an integer , is the primary key of the customer table
The first_name field data type is a string ,
The last_name field is a string
The created_at field data type is a Date in format (Y-m-d)
The updated_at field data type is a Date in format (Y-m-d)

In case the request was processed correctly the output will be a json array
with key parameters 'response' which is an array with keys statusCode(integer data) , and key result with value an array of customers .

JWT_AUTHORIZATION_RESTFUL_API_PHP / getCustomerBy Id successfully / **getCustomerBy Id successfullly**

GET http://localhost/restful_api_php/customers/3

Params Headers (1) Body

Query Params

	KEY	VALUE	DESCRIPTION
	Key	Value	Description

Body Headers (11) Status Code 200 OK

Pretty Raw Preview JSON


```
{
    "response": {
        "statusCode": 200,
        "result": {
            "id": "3",
            "first_name": "Mateo",
            "last_name": "Fornara",
            "email": "mateofornara@gmail.com",
            "created_at": "2023-07-29",
            "updated_at": "2023-08-07"
        }
    }
}
```

case: getCustomer request with a customer Id not present in database .

In case the request was processed with a customer Id notpresent in database the system will generate an error and the output will be a json array with key parameter 'error ' and value an array of 2 elements . The first element is a key with statusCode and value an integer and the second element is a key named 'message having for value a string .

JWT_AUTHORIZATION_RESTFUL_... / getCustomerBy Id not existent in database error ... / getCustomerBy Id not existent in database error ...

GET http://localhost/restful_api_php/customers/74

Params Headers (1) Body

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Headers (11) Status Code 404 Not Found

Pretty Raw Preview JSON


```
{
    "error": {
        "statusCode": 404,
        "message": "Customer not found"
    }
}
```

case: getCustomer request with a customer Id not an integer data type .

In case the request was processed with a customer Id not integer the system will generate an error and the output will be a json array with key parameter 'error ' and value an array of 2 elements . The first element is a key with statusCode and value an integer and the second element is a key named 'message having for value a string .

JWT_AUTHORIZATION_RESTFU... / getCustomer By Id that is not an integer data type ... / getCustomer By Id that is not an integer data type ... Sa

GET http://localhost/restful_api_php/customers/sadf

Params Headers (1) Body

Query Params

	KEY	VALUE	DESCRIPTION	ooo
	Key	Value	Description	

Body Headers (10) Status Code 400 Bad Request

Pretty Raw Preview JSON


```
1 {
2     "error": {
3         "statusCode": 400,
4         "message": "Datatype not valid for customerId. Should be numeric"
5     }
6 }
```

POST /customers

Input parameters :

There are 3 input fields mandatory : first_name , last_name , and email .
The field of data registration called 'created_at' with be generate by the system and inserted in the database automatically . In case these 3 fields are not set in input the system will generate an error .

The system makes a validation for these 3 data types before the insertion in the database

The first_name field data type must be string .
The last_name field datatype must be string .

The system also checks if the email is typed correctly and if it does not exist previously in database .

Each time a new customer object is created , in the customers table is inserted also the field user_id . So the user who created this customer object . The field user_id is not mandatory as an input field parameter for the post request . The insertion of the user_id in the customers table can be totally omitted .
This was done in order to know the user who accessed the system and created that user .

case : postRequestSuccessfully

Response :

Returns a JSON array with a key named 'response' having for value an array of 2 elements. The first element is a key named 'statusCode' with value an integer , and the second element is key 'message' having for value a string .

JWT_AUTHORIZATION_RESTFUL_API_PHP / addCustomer successfully / **addCustomer successfully**

Save

POST http://localhost/restful_api_php/customers

Params Headers (1) Body

none form-data x-www-form-urlencoded raw binary GraphQL Text

```
{
    "first_name": "Lionel",
    "last_name": "Messi",
    "email": "lionelmessi@gmail.com"
}
```

Body Headers (11) Status Code 201 Created

Pretty Raw Preview JSON

```
{
    "response": {
        "statusCode": 201,
        "result": "Created successfully"
    }
}
```

Case : PostRequest with invalid token

Body Headers (11) Status Code 498 INVALID_TOKEN

Pretty Raw Preview JSON


```
1 {
2     "error": {
3         "statusCode": 498,
4         "message": "Signature verification failed"
5     }
6 }
```


case :PostRequest with expired token

Body Headers (11) Status Code 498 INVALID_TOKEN

Pretty Raw Preview JSON


```
1 {
2     "error": {
3         "statusCode": 498,
4         "message": "Expired token"
5     }
6 }
```

case : PostRequest with missing parameter first_name

JWT_AUTHORIZATION_RESTFUL_... / addCustomer missing parameter first_name error... / **addCustomer missing parameter first_name error...** Save

POST http://localhost/restful_api_php/customers

Params Headers (1) Body

none form-data x-www-form-urlencoded raw binary GraphQL Text

```
1 {
2   "first_name" : "",
3   "last_name" : "Gaze",
4   "email" : "auroragaze@gmail.com"
5 }
```

Body Headers (11) Status Code 422 Unprocessable Entity

Pretty Raw Preview JSON

```
1 {
2   "error": {
3       "statusCode": 422,
4       "message": "first_name parameter is required"
5   }
6 }
```

case :postRequest with parameter first_name not a string

JWT_AUTHORIZATION_RESTFU... / addCustomer with first_name data type not valid e... / **addCustomer with first_name data type not valid e...** Save

POST http://localhost/restful_api_php/customers

Params Headers (1) Body

none form-data x-www-form-urlencoded raw binary GraphQL Text

```
1 {
2   "first_name" : 425,
3   "last_name" : "Keci",
4   "email" : "gretakeci@gmail.com"
5 }
```

Body Headers (10) Status Code 400 Bad Request

Pretty Raw Preview JSON

```
1 {
2   "error": {
3       "statusCode": 400,
4       "message": "Datatype not valid for first_name. Should be string"
5   }
6 }
```

case : postRequest with an input email existent in database

JWT_AUTHORIZATION_REST... / addCustomer with an email alredy existent in datab... / addCustomer with an email alredy existent in datab...

POST http://localhost/restful_api_php/customers

Params Headers (1) Body

none form-data x-www-form-urlencoded raw binary GraphQL Text

```
1 {
2   "first_name" : "Greta",
3   "last_name" : "Keci",
4   "email" : "gretakeci@gmail.com"
5 }
```

Body Headers (11) Status Code 409 Conflict

Pretty Raw Preview JSON

```
1 {
2     "error": {
3         "statusCode": 409,
4         "message": "This email already exists"
5     }
6 }
```

case : postRequest missing parameter last name

JWT_AUTHORIZATION_RESTFUL_... / addCustomer missing last name parameter error ... / addCustomer missing last name parameter error ... Save

POST http://localhost/restful_api_php/customers

Params Headers (1) Body

none form-data x-www-form-urlencoded raw binary GraphQL Text

```
1 {
2
3     "first_name": "Antonetta",
4     "last_name": "",
5     "email": "antonettacarra@gmail.com"
6
7 }
```

Body Headers (11) Status Code 422 Unprocessable Entity

Pretty Raw Preview JSON

```
1 {
2     "error": {
3         "statusCode": 422,
4         "message": "last_name parameter is required"
5     }
6 }
```

case : postRequest with email not written correctly

JWT_AUTHORIZATION_RESTFUL... / addCustomer with email not written correctly err... / addCustomer with email not written correctly err...

POST http://localhost/restful_api_php/customers

Params Headers (1) Body

none form-data x-www-form-urlencoded raw binary GraphQL Text

```
1 {
2 
3     "first_name": "Antonetta",
4     "last_name": "Carra",
5     "email": "antonettacarra##gmail.com"
6 
7 }
```

Body Headers (10) Status Code 400 Bad Request

Pretty Raw Preview JSON

```
1 {
2     "error": {
3         "statusCode": 400,
4         "message": "Datatype not valid for email. Should be a valid email"
5     }
6 }
```

case : postRequest with parameter email not set

JWT_AUTHORIZATION_RESTFUL_... / addCustomer parameter field email not set error ... / addCustomer parameter field email not set error ... Save

POST http://localhost/restful_api_php/customers

Params Headers (1) Body

none form-data x-www-form-urlencoded raw binary GraphQL Text

```
1 {
2
3     "first_name": "Antonetta",
4     "last_name": ""
5
6
7 }
```

Body Headers (11) Status Code 422 Unprocessable Entity

Pretty Raw Preview JSON

```
1 {
2     "error": {
3         "statusCode": 422,
4         "message": "Invalid input"
5     }
6 }
```

PUT /customers/{id}

The input parameters are optional . They can or can not be set .

If the input fields first_name,last_name ,email ,created_at are not set then the system will keep the old data . If any of these fields is set then the system will update only that field keeping the others with their values not modified .

The updated_at field is a DATE type in format (Y-m-d) and it will be generated and inserted automatically by the system in the database each time the post request will have a successful result .

Case : putRequest Successfully

The response will be a json array with a key named 'response' having as value an array of 2 elements. The first element is a key named 'statusCode' with value an integer , and the second element is key 'result' having for value a string .

JWT_AUTHORIZATION_RESTFUL_API_PHP / UpdateCustomer successfully / **UpdateCustomer successfully**

PUT http://localhost/restful_api_php/customers/2

Params Headers (1) Body

none form-data x-www-form-urlencoded raw binary GraphQL Text

```
{
    "first_name": "Francesco",
    "last_name": "Tarao",
    "email": "alessandrotara@gmail.com",
    "created_at": "2023-07-29"
}
```

ody Headers (11) Status Code 200 OK

Pretty Raw Preview JSON

```
{
    "response": {
        "statusCode": 200,
        "result": "Updated successfully"
    }
}
```

Case : putRequest with parameter created_at not in a valid date format (Y-m-d)

JWT_AUTHORIZATION_RES... / Update Customer generate error with parameter cre... / Update Customer generate error with parameter cre...

PUT http://localhost/restful_api_php/customers/2

Params Headers (1) Body

none form-data x-www-form-urlencoded raw binary GraphQL Text

```
1 {
2 
3     "first_name": "Francesco",
4     "last_name": "Tarao",
5     "email": "alessandrotara@gmail.com",
6     "created_at": "11111111-07-29"
7 }
```

Body Headers (10) Status Code 400 Bad Request

Pretty Raw Preview JSON

```
1 {
2     "error": {
3         "statusCode": 400,
4         "message": "Datatype not valid for created_at. Should be a Y-m-d format"
5     }
6 }
```

case : putRequest with expired token

JWT_AUTHORIZATION_RESTFUL_API... / UpdateCustomer with token expired error gene... / **UpdateCustomer with token expired error gene...** Save

PUT http://localhost/restful_api_php/customers/2

Params Headers (1) Body

none form-data x-www-form-urlencoded raw binary GraphQL Text

```
1 {
2 
3     "first_name": "Francesco",
4     "last_name": "Tarao",
5     "email": "alessandrotara@gmail.com",
6     "created_at": "2023-07-29"
7 }
```

Body Headers (11) Status Code 498 INVALID_TOKEN

Pretty Raw Preview JSON

```
1 {
2     "error": {
3         "statusCode": 498,
4         "message": "Expired token"
5     }
6 }
```

case : putRequest with parameter last_name not a valid string

JWT_AUTHORIZATION_RESTF... / Update Customer generate error with parameter la... / Update Customer generate error with parameter la...

PUT http://localhost/restful_api_php/customers/2

Params Headers (1) Body

none form-data x-www-form-urlencoded raw binary GraphQL Text

```
1 {
2 
3     "first_name": "Francesco",
4     "last_name": 44,
5     "updated_at": "2023-07-29"
6 }
```

Body Headers (10) Status Code 400 Bad Request

Pretty Raw Preview JSON

```
1 {
2     "error": {
3         "statusCode": 400,
4         "message": "Datatype not valid for last_name. Should be string"
5     }
6 }
```

DELETE /customers/{id}

Input parameters : customer Id

The customer Id should be an integer as data type and must also exist in customers table , otherwise the system will generate an error .

Case : deleteCustomer Successfully

Body Headers (11) Status Code 200 OK

Pretty Raw Preview JSON


```
{
    "response": {
        "statusCode": 200,
        "result": "Deleted successfully"
    }
}
```


case : deleteCustomer with invalid token

Body Headers (11) Status Code 498 INVALID_TOKEN

Pretty Raw Preview JSON


```
{
    "error": {
        "statusCode": 498,
        "message": "Signature verification failed"
    }
}
```

case : deleteCustomer with token expired

Body Headers (11) Status Code 498 INVALID_TOKEN

Pretty Raw Preview JSON


```
1 {
2     "error": {
3         "statusCode": 498,
4         "message": "Expired token"
5     }
6 }
```


case : deleteCustomer with an error in sql query . This will generate a 500 internal server error .

Body Headers (10) Status Code 500 Internal server error

Pretty Raw Preview JSON


```
1 {
2     "error": {
3         "statusCode": 500,
4         "message": "INTERNAL SERVER ERROR"
5     }
6 }
```

Access/Test REST Api with Curl or Guzzle

After having simulated the clients requests with the Postman tool
we will access the Api not anymore with Postman tool but using the Curl or Guzzle which are libraries that helps us making request to the Api . The Guzzle will also be used for testing our RestApi . The installation is done through command 'composer require guzzlehttp/guzzle'
We have to require the file autoload.php in our testing file bcs we have installed them with composer . The tests are done in file CustomerControllerTest.php . This file is located under the tests/controller folder . A phpunit.xml file for the test suites was created manually.
The authorization for the requests are not done anymore setting the key Authorization in Postman headers but instantiating an object client of Class GuzzleHttp\Client
(Please make reference to file CustomerControllerTest.php).
Reported below an example of POST request specifying the Authorization key
inside the headers associative array .

```
 'headers' => [
                 'Content-Type' => 'application/json',
                 'Authorization' => 'Bearer ' . $this->token,
                 'Accept' => 'application/json',
              ],


$response = $this->client->request('POST', 'customers', [

          'http_errors' => false ,

          'headers' => [
             'Content-Type' => 'application/json',
             'Authorization' => 'Bearer ' . $this->token,
             'Accept' => 'application/json',
             ],

          'json' => [
             'first_name'   => "Marlon",
             'last_name'  => "Brando",
             'email'    => "marlonbrando##gmail.com"
          ]

       ]);
```