



KMS TOOLS API

This REST API helps interact resources within the KMS TOOLS SQL Database

Overview:

KMS TOOLS is a hardware and tools brick and mortar store that helps its customers browse, reserve and request transfer orders from the store branch without needing to be in the store physically. Admin/ Sales Associates can access and update this system to keep it up to date and make sure the stock and orders are fulfilled as per expectations. This REST API is designed to be typically interacted by customer/ admins through a web browser. The API provides functionality to developers to act on behalf of either the Admin or the Customer. The KMS TOOLS Website implementation uses a php curl html implementation of the KMS TOOLS REST API as a layer to interact with the KMS SQL Database.

The following resources are accessible via the REST API that you should familiarize yourself with:

Aisle A Store Branch has multiple aisles that store products in its sections. Each Aisle has a unique AID to identify them and each aisle is associated with the `branch_id` of the store branch it is located in.

Carries Each Store Branch carries products to sell. Each Carries entry has the SKU, quantity and `on_display` identification parameters along with the `branch_id` of the branch being referred to.

Contains Each Transfer Order created by the customer contains information about the product requested. Each Contains entry has a `customer_id` of the customer requesting the order along with the `order_id`, SKU, quantity and cost of the requested product.

Customer Each customer accessing the system has his/her details stored in the database. Each Customer entry has a unique `customer_id` and username along with `first_name`, `last_name`, `password`, `email_id` and `phone_no` of the customer.

Displayed_in Each Product can be displayed in a special section in an Aisle to gauge more interest from customers. Each Displayed_in entry has the SKU of the product along with the AID, `branch_id` and `segment_of_aisle` the product is displayed in.

Found_in Each Product is found in an Aisle to be sold to customers. Each Found_in entry has the SKU of the product along with the AID, `branch_id` and `segment_of_aisle` the product is found in.

Orders Store Branches can order Product stock from its Suppliers. Each order contains the SKU, quantity, cost of the product being ordered along with the supplier information such as `supplier_id`, `ship_date`, `expected_receive_date` as well as the `branch_id` of the store requesting the stock.

Places_Hold A customer can place a hold request for a product to pick up instore. Each Hold request has a `customer_id`, SKU, quantity, `date_placed`, `date_released` and the price of the product requested.

Product A Store Branch contains a stock of products which have their information stored in the database.

Each product has a unique SKU, its type, name, description, additional_information, ~~UPC~~, regular_price, sale_price, club_price as well as the quantity available. ▼

Sales_Associate The system can be accessed with admin user privileges through a separate login method. Sales Associates in the company have such designation and access to the system. Each sales associate has a unique SID, first_name, last_name, username, password and section they work under.

Ships_from Each transfer_order ships from a branch. Each Ships_from entry has a customer_id of the customer requesting the transfer, order_id of the order requested, branch_id of the branch the product will ship from along with the ship_date.

Ships_to Each transfer_order ships to a branch. Each Ships_to entry has a customer_id of the customer requesting the transfer, order_id of the order requested, branch_id of the branch the product will ship to along with the receive_date.

Store_Branch The system manages information about the store branches. Each Store Branch has a unique branch_id, email, phone_no, province, city and the street_no the branch is located in.

Supplier Supplier information is also stored in the database who supply product stock to the store branches. Each Supplier entry has a unique id, name, phone_no and its email to contact them.

Transfer_Order A customer can request a product that may not be available at their preferred branch, so they can request a transfer of their order from another branch. Each transfer order entry has the customer_id of the customer requesting the transfer, order_id, date_order and whether the order has been invoiced or not.

Works_for A Sales Associate works for a Store Branch. Each Works_for entry has a branch_id, SID and the shift details of the Sales Associate.

Aisle

Access information regarding aisles from KMS TOOLS database

GET Read Aisles

```
http://localhost/phprest/api/aisle/read.php
```

You can use this endpoint to read information about all aisles stored in the database



Example Request

Read

```
curl --location --request GET 'http://localhost/phprest/api/aisle/read.php'
```

Example Response

200 OK

Body Headers (9)

```
{
  "aisles": [
    {
      "AID": "1",
      "branch_id": "1"
    },
    {
      "AID": "2",
      "branch_id": "1"
    }
  ]
}
```

[View More](#)

GET Read One Aisle

```
http://localhost/phprest/api/aisle/read_one.php
```

You can use this endpoint to read an aisle from table using an AID parameter in the query, eg: To retrieve aisle with AID=1 you can use `http://localhost/phprest/api/aisle/read_one.php?AID=1` (`http://localhost/phprest/api/aisle/read_one.php?AID=1`)

Example Request

Read One

```
curl --location --request GET 'http://localhost/phprest/api/aisle/read_one.php?AID=1'
```

Example Response

200 OK

Body Headers (11)

Documentation Settings ▼

```
{
  "AID": "1",
  "branch_id": "1"
}
```

POST Create Aisle

```
http://localhost/phprest/api/aisle/create.php
```

You can create an aisle entry by sending a body in the request with the AID and branch_id values to be set to the new aisle

Example Request

Create

```
curl --location --request POST 'http://localhost/phprest/api/aisle/create.php' \
--data-raw '{
  "AID": "5",
  "branch_id": "1"
}'
```

Example Response

201 Created

Body Headers (11)

```
{
  "message": "Aisle was created."
}
```

POST Update Aisle

```
http://localhost/phprest/api/aisle/update.php
```

You can updates an aisle entry by supplying the all the aisle data values for AID, branch_id in a raw body Documentation is missing ▼

Example Request

Update

```
curl --location --request POST 'http://localhost/phprest/api/aisle/update.php' \
--data-raw '{
  "AID": "4",
  "branch_id": 1
}'
```

Example Response

200 OK

Body Headers (11)

```
{
  "message": "Aisle was updated."
}
```

DEL Delete Aisle

http://localhost/phprest/api/aisle/delete.php

You can use this endpoint to delete an aisle from table using an AID parameter in the query, eg: To delete aisle with AID=1 you can use http://localhost/phprest/api/aisle/delete.php?AID=1 (http://localhost/phprest/api/aisle/delete.php?AID=1)

Example Request

Delete

```
curl --location --request DELETE 'http://localhost/phprest/api/aisle/delete.php?AID=4'
```

Example Response

200 OK

[Documentation Settings](#) ▼

Body Headers (11)

```
{  
  "message": "Aisle was deleted."  
}
```

Carries

Access information regarding Carries from KMS TOOLS SQL database

GET Read Carries

```
http://localhost/phprest/api/carries/read.php
```

You can use this endpoint to read information about the items carried by a branch stored in the database

Example Request

Read

```
curl --location --request GET 'http://localhost/phprest/api/carries/read.php'
```

Example Response

200 OK

Body Headers (9)

```
⌘
{
  "carries": [
    {
      "SKU": "1000",
      "branch_id": "1",
      "quantity": "5",
      "on_display": "Yes"
    },
    {
      "SKU": "1001".
    }
  ]
}
```

[Documentation Settings](#) ▼[View More](#)

GET Read One Carries

```
http://localhost/phprest/api/carries/read_one.php
```

You can use this endpoint to read a Carry from table using an SKU parameter in the query, eg: To retrieve Carries with SKU=1000 you can use `http://localhost/phprest/api/carries/create.php?SKU=1000` (`http://localhost/phprest/api/carries/create.php?SKU=1000`)

Example Request

Read One

```
curl --location --request GET 'http://localhost/phprest/api/carries/read_one.php?SKU=1001'
```

Example Response

200 OK

Body Headers (11)

```
{
  "SKU": "1001",
  "branch_id": "1",
  "quantity": "10",
  "on_display": "Yes"
}
```



POST Create Carries

[Documentation Settings](#) ▼

```
http://localhost/phprest/api/carries/create.php
```

You can create an carries entry by sending a body in the request with the SKU, branch_id, quantity and on_display parameters to be set to the new carries

Example Request

[Create](#)

```
curl --location --request POST 'http://localhost/phprest/api/carries/create.php' \
--data-raw '{
  "SKU": "1001",
  "branch_id": "1",
  "quantity": "5",
  "on_display": "Yes"
}'
```

Example Response

201 Created

[Body](#) [Headers \(11\)](#)

```
{
  "message": "Carries was created."
}
```

POST Update Carries

```
http://localhost/phprest/api/carries/update.php
```

You can updates an Carries entry by supplying the all the Carry data values for SKU, branch_id, quantity and on_display parameters in a raw body



Example Request

Update

```
curl --location --request POST 'http://localhost/phprest/api/carries/update.php' \
--data-raw '{
  "SKU": "1001",
  "branch_id": "1",
  "quantity": "5",
  "on_display": "Yes"
}'
```

Example Response

200 OK

Body	Headers (11)
<pre>{ "message": "Carries was updated." }</pre>	

DEL Delete Carries

```
http://localhost/phprest/api/carries/delete.php?SKU=1001
```

You can use this endpoint to delete a Carries entry from table using an SKU parameter in the query, eg: To delete Carries with SKU=1001 you can use `http://localhost/phprest/api/carries/delete.php?SKU=1001` (`http://localhost/phprest/api/carries/delete.php?SKU=1001`)

PARAMS

SKU
1001

Example Request

Delete

```
curl --location --request DELETE 'http://localhost/phprest/api/carries/delete.php?SKU=1001'
```

[Documentation Settings](#) ▼

Example Response

200 OK

[Body](#) [Headers \(11\)](#)

```
{
  "message": "Carries was deleted."
}
```

Contains

Access information regarding information the Transfer_Order Contains from KMS TOOLS SQL database

GET Read Contains

```
http://localhost/phprest/api/contains/read.php
```

You can use this endpoint to read information about what the transfer order contains stored in the database

Example Request

Read

```
curl --location --request GET 'http://localhost/phprest/api/contains/read.php'
```

Example Response

200 OK

[Body](#) [Headers \(9\)](#)

```
⌘  
"contains": [  
  {  
    "customer_id": "1000",  
    "order_id": "2",  
    "SKU": "1000",  
    "quantity": "1",  
    "cost": "10.00"  
  },  
  {
```

[Documentation Settings](#) ▼[View More](#)

GET Read One Contain

```
http://localhost/phprest/api/contains/read_one.php
```

You can use this endpoint to read a contains entry from table using an order_id parameter in the query, eg:
To retrieve contains with order_id=1 you can use `http://localhost/phprest/api/contains/read_one.php?order_id=1` (`http://localhost/phprest/api/contains/read_one.php?order_id=1`)

Example Request

Read One

```
curl --location --request GET 'http://localhost/phprest/api/contains/read_one.php?order_id=2'
```

Example Response

200 OK

Body Headers (11)

```
{  
  "customer_id": "1000",  
  "order_id": "2",  
  "SKU": "1000",  
  "quantity": "1",  
  "cost": "10.00"  
}
```

POST Create Contain

[Documentation Settings](#) ▼

```
http://localhost/phprest/api/contains/create.php
```

You can create a contains entry by sending a body in the request with the customer_id, order_id, SKU, quantity and cost parameters to be set to the new contains

Example Request

[Create](#)

```
curl --location --request POST 'http://localhost/phprest/api/contains/create.php' \
--data-raw '{
  "customer_id": "1000",
  "order_id": "2",
  "SKU": "1000",
  "quantity": "1",
  "cost": "10.00"
}'
```

Example Response

201 Created

Body Headers (11)

```
{
  "message": "contains was created."
}
```

POST Update Contain

```
http://localhost/phprest/api/contains/update.php
```

You can updates an Contains entry by supplying the all the Contains data values for customer_id, order_id, SKU, quantity and cost parameters in a raw body



Example Request

Update

```
curl --location --request POST 'http://localhost/phprest/api/contains/update.php' \
--data-raw '{
  "customer_id": "1000",
  "order_id": "2",
  "SKU": "1000",
  "quantity": "1",
  "cost": "10.00"
}'
```

Example Response

200 OK

Body Headers (11)

```
{
  "message": "contains was updated."
}
```

DEL Delete Contain

```
http://localhost/phprest/api/contains/delete.php
```

You can use this endpoint to delete a Contains entry from table using an order_id parameter in the query, eg: To delete Contains with order_id=2 you can use `http://localhost/phprest/api/contains/delete.php?order_id=2` (`http://localhost/phprest/api/contains/delete.php?order_id=2`)

Example Request

Delete

```
curl --location --request DELETE 'http://localhost/phprest/api/contains/delete.php?order_id=2'
```

Example Response

200 OK

Body Headers (11)

Documentation Settings ▼

```
{  
  "message": "contains was deleted."  
}
```

Customer

Access information regarding customers from KMS TOOLS SQL database

GET Read Customers

```
http://localhost/phprest/api/customer/read.php
```

You can use this endpoint to read information about all customers stored in the database

Example Request

Read

```
curl --location --request GET 'http://localhost/phprest/api/customer/read.php'
```

Example Response

200 OK

Body Headers (9)

[Documentation Settings](#) ▼

```
"customers": [  
  {  
    "customer_id": "1000",  
    "first_name": "customer",  
    "last_name": "1",  
    "username": "customer",  
    "password": "customer",  
    "email_id": "customer1@email.com",  
    "phone no": "1112223333"
```

[View More](#)

GET Read One Customer

```
http://localhost/phprest/api/customer/read_one.php
```

You can use this endpoint to read a Customer entry from table using a customer_id parameter in the query, eg: To retrieve Customer with customer_id=1000 you can use
`http://localhost/phprest/api/customer/read_one.php?customer_id=1000`
(`http://localhost/phprest/api/customer/read_one.php?customer_id=1000`)

Example Request

[Read One Customer](#)

```
curl --location --request GET 'http://localhost/phprest/api/customer/read_one.php?customer_id=1000'
```

Example Response

200 OK

[Body](#) [Headers \(11\)](#)

```
f  
{  
  "customer_id": "1000",  
  "first_name": "customer",  
  "last_name": "1",  
  "username": "customer",  
  "password": "customer",  
  "email_id": "customer1@email.com",  
  "phone_no": "1112223333"  
}
```

[Documentation Settings](#) ▼

POST Create Customer

```
http://localhost/phprest/api/customer/create.php
```

You can create a customer entry by sending a body in the request with the customer_id, first_name, last_name, username, password, email_id and phone_no parameters to be set to the new customer

Example Request

[Create Customer](#)

```
curl --location --request POST 'http://localhost/phprest/api/customer/create.php' \  
--data-raw '{  
  "customer_id": "1004",  
  "first_name": "customer",  
  "last_name": "1",  
  "username": "customer",  
  "password": "customer",  
  "email_id": "customer1@email.com",  
  "phone_no": "1112223333"  
}'
```

Example Response

201 Created

Body	Headers (11)
------	--------------

```
{  
  "message": "customer was created."  
}
```


POST Update Customer

http://localhost/phprest/api/customer/update.php

You can update a Customer entry by supplying the all the Customer parameter values for customer_id, first_name, last_name, username, password, email_id and phone_no with the changes in a raw body

Example Request

Update Customer

```
curl --location --request GET 'http://localhost/phprest/api/customer/update.php' \
--data-raw '{
  "customer_id": "1000",
  "first_name": "customer",
  "last_name": "1",
  "username": "customer",
  "password": "customer",
  "email_id": "customer1@email.com",
  "phone_no": "1112223333"
}'
```

Example Response

200 OK

Body Headers (11)

```
{
  "message": "customer was updated."
}
```

POST Login Authentication

http://localhost/phprest/api/customer/login.php



Example Request

Login Authentication

```
curl --location --request POST 'http://localhost/phprest/api/customer/login.php' \
--data-raw '{
  "username": "customer",
  "password": "customer"
}'
```

Example Response

200 OK

Body Headers (11)

```
{
  "message": "Login successful."
}
```

DEL Delete Customer

http://localhost/phprest/api/customer/delete.php

You can use this endpoint to delete a Customer entry from table using an customer_id parameter in the query, eg: To delete Customer with customer_id=1004 you can use
http://localhost/phprest/api/customer/delete.php?customer_id=1004
(http://localhost/phprest/api/customer/delete.php?customer_id=1004)

Example Request

Delete Customer

```
curl --location --request DELETE 'http://localhost/phprest/api/customer/delete.php?customer_id=1004'
```

Example Response

200 OK

Body Headers (11)

Documentation Settings ▼

```
{  
  "message": "customer was deleted."  
}
```

Displayed_in

Access information regarding where products are displayed_in the store branch from a KMS TOOLS SQL database

GET Read Displayed_in

```
http://localhost/phprest/api/displayed_in/read.php
```

You can use this endpoint to read information where all the products are displayed in the store from the database

Example Request

Read Displayed_in

```
curl --location --request GET 'http://localhost/phprest/api/displayed_in/read.php'
```

Example Response

200 OK

Body Headers (9)

```
⌘
{
  "displayed_in": [
    {
      "SKU": "1000",
      "AID": "1",
      "branch_id": "1",
      "segment_of_aisle": "1"
    },
    {
      "SKU": "1001".
    }
  ]
}
```

[Documentation Settings](#) ▼[View More](#)

GET Read One Displayed_in

```
http://localhost/phprest/api/displayed_in/read_one.php
```

You can use this endpoint to read a Displayed_in entry from table using a product SKU parameter in the query, eg: To retrieve Displayed_in with SKU=1000 you can use
`http://localhost/phprest/api/displayed_in/read_one.php?SKU=1000`
(`http://localhost/phprest/api/displayed_in/read_one.php?SKU=1000`)

Example Request

Read One Displayed_in

```
curl --location --request GET 'http://localhost/phprest/api/displayed_in/read_one.php?SKU=1000'
```

Example Response

200 OK

Body Headers (11)

```
{
  "SKU": "1000",
  "AID": "1",
  "branch_id": "1",
  "segment_of_aisle": "1"
}
```

POST Create Displayed_in

[Documentation Settings](#) ▼

http://localhost/phprest/api/displayed_in/create.php

You can create a Displayed_in entry by sending a body in the request with the SKU, AID, branch_id and segment_of_aisle parameters to be set to the new product Displayed_in entry in the database

Example Request

[Create Displayed_in](#)

```
curl --location --request POST 'http://localhost/phprest/api/displayed_in/create.php' \
--data-raw '{
  "SKU": "1001",
  "AID": "1",
  "branch_id": "1",
  "segment_of_aisle": "2"
}'
```

Example Response

201 Created

Body Headers (11)

```
{
  "message": "displayed_in was created."
}
```

POST Update Displayed_in

http://localhost/phprest/api/displayed_in/update.php

You can update a Displayed_in entry by supplying the all the Displayed_in parameter values for SKU, AID, branch_id and segment_of_aisle with the changes in a raw body

Example Request

[Update Displayed_in](#)
[Documentation Settings](#) ▼

```
curl --location --request POST 'http://localhost/phprest/api/displayed_in/update.php' \  
--data-raw '{  
  "SKU": "1000",  
  "AID": "1",  
  "branch_id": "1",  
  "segment_of_aisle": "1"  
}'
```

Example Response

200 OK

[Body](#) [Headers \(11\)](#)

```
{  
  "message": "displayed_in was updated."  
}
```

DEL Delete Displayed_in

```
http://localhost/phprest/api/displayed_in/delete.php
```

You can use this endpoint to delete a Displayed_in entry from table using an SKU parameter of a product in the query, eg: To delete Displayed_in with SKU=1001 you can use
`http://localhost/phprest/api/displayed_in/delete.php?SKU=1001`
(`http://localhost/phprest/api/displayed_in/delete.php?SKU=1001`)

Example Request

[Delete Displayed_in](#)

```
curl --location --request DELETE 'http://localhost/phprest/api/displayed_in/delete.php?SKU=1001'
```

Example Response

200 OK

[Body](#) [Headers \(11\)](#)

```
{
  "message": "displayed_in was deleted."
}
```

Found_in

Access information regarding where products are found_in the store branch from a KMS TOOLS SQL database

GET Read Found_in

http://localhost/phprest/api/found_in/read.php

You can use this endpoint to read information where all the products are found in the store from the database

Example Request

Read Found_in

```
curl --location --request GET 'http://localhost/phprest/api/found_in/read.php'
```

Example Response

200 OK

Body Headers (9)

```
f  
"found_in": [  
  {  
    "SKU": "1000",  
    "AID": "1",  
    "branch_id": "1",  
    "segment_of_aisle": "1"  
  },  
  {  
    "SKU": "1001".
```

[Documentation Settings](#) ▼[View More](#)

GET Read One Found_in

```
http://localhost/phprest/api/found_in/read_one.php
```

You can use this endpoint to read a Found_in entry from table using a product SKU parameter in the query, eg: To retrieve Found_in with SKU=1000 you can use `http://localhost/phprest/api/found_in/read_one.php?SKU=1000` (`http://localhost/phprest/api/found_in/read_one.php?SKU=1000`)

Example Request

Read One Found_in

```
curl --location --request GET 'http://localhost/phprest/api/found_in/read_one.php?SKU=1000'
```

Example Response

200 OK

Body Headers (11)

```
{  
  "SKU": "1000",  
  "AID": "1",  
  "branch_id": "1",  
  "segment_of_aisle": "1"  
}
```


POST Create Found_in

[Documentation Settings](#) ▼

```
http://localhost/phprest/api/found_in/create.php
```

You can create a Found_in entry by sending a body in the request with the SKU, AID, branch_id and segment_of_aisle parameters to be set to the new product Found_in entry in the database

Example Request

[Create Found_in](#)

```
curl --location --request POST 'http://localhost/phprest/api/found_in/create.php' \
--data-raw '{
  "SKU": "1000",
  "AID": "1",
  "branch_id": "1",
  "segment_of_aisle": "1"
}'
```

Example Response

201 Created

Body Headers (11)

```
{
  "message": "found_in was created."
}
```

POST Update Found_in

```
http://localhost/phprest/api/found_in/update.php
```

You can update a Found_in entry by supplying the all the Found_in parameter values for SKU, AID, branch_id and segment_of_aisle with the changes in a raw body



Example Request

Update Found_in

```
curl --location --request POST 'http://localhost/phprest/api/found_in/update.php' \
--data-raw '{
  "SKU": "1000",
  "AID": "1",
  "branch_id": "1",
  "segment_of_aisle": "2"
}'
```

Example Response

200 OK

Body Headers (11)

```
{
  "message": "found_in was updated."
}
```

DEL Delete Found_in

```
http://localhost/phprest/api/found_in/delete.php
```

You can use this endpoint to delete a Found_in entry from table using an SKU parameter of a product in the query, eg: To delete Found_in with SKU=1001 you can use
`http://localhost/phprest/api/found_in/delete.php?SKU=1001`
(`http://localhost/phprest/api/found_in/delete.php?SKU=1001`)

Example Request

Delete Found_in

```
curl --location --request DELETE 'http://localhost/phprest/api/found_in/delete.php?SKU=1000'
```

Example Response

200 OK

Body Headers (11)

```
{  
  "message": "found_in was deleted."  
}
```

Documentation Settings ▼

Order

Access information regarding orders made by the store branch to its supplier from a KMS TOOLS SQL database

GET Read Orders

```
http://localhost/phprest/api/orders/read.php
```

You can use this endpoint to read information about all orders saved in the database

Example Request

Read Orders

```
curl --location --request GET 'http://localhost/phprest/api/orders/read.php'
```

Example Response

200 OK

Body Headers (9)



```
"orders": [  
  {  
    "SKU": "1001",  
    "supplier_id": "1001",  
    "branch_id": "1",  
    "quantity": "5",  
    "cost": "200.00",  
    "ship_date": "2020-12-16",  
    "expected_receive_date": "2020-12-23"
```

[View More](#)

GET Read One Order

```
http://localhost/phprest/api/orders/read_one.php
```

You can use this endpoint to read a Order entry from table using a product SKU parameter in the query, eg:
To retrieve Order with SKU=1000 you can use `http://localhost/phprest/api/orders/read_one.php?SKU=1000`
(`http://localhost/phprest/api/orders/read_one.php?SKU=1000`)

Example Request

[Read One Order](#)

```
curl --location --request GET 'http://localhost/phprest/api/orders/read_one.php?SKU=1000'
```

Example Response

200 OK

Body Headers (11)

```
{  
  "SKU": "1000",  
  "supplier_id": "1000",  
  "branch_id": "1",  
  "quantity": "5",  
  "cost": "100.00",  
  "ship_date": "2020-12-26",  
  "expected_receive_date": "2020-12-29"  
}
```



POST Update Order

http://localhost/phprest/api/orders/update.php

You can update a Order entry by supplying the all the Order parameter values for SKU, supplier_id, branch_id, quantity, cost, ship_date and expected_recieve_date with the changes in a raw body

Example Request

Update Order

```
curl --location --request POST 'http://localhost/phprest/api/orders/update.php' \
--data-raw '{
  "SKU": "1000",
  "supplier_id": "1000",
  "branch_id": "1",
  "quantity": "5",
  "cost": "100.00",
  "ship_date": "2020-12-26",
  "expected_receive_date": "2020-12-29"
}'
```

Example Response

200 OK

Body Headers (11)

```
{
  "message": "order was updated."
}
```

POST Create Order

http://localhost/phprest/api/orders/create.php

You can create a Order entry by sending a body in the request with the SKU, supplier_id, branch_id, quantity, cost, ship_date and expected_receive_date parameters to be set to the new Order



Example Request

Create Order

```
curl --location --request POST 'http://localhost/phprest/api/orders/create.php' \
--data-raw '{
  "SKU": "1000",
  "supplier_id": "1000",
  "branch_id": "1",
  "quantity": "5",
  "cost": "100.00",
  "ship_date": "2020-12-26",
  "expected_receive_date": "2020-12-29"
}'
```

Example Response

201 Created

Body Headers (11)

```
{
  "message": "order was created."
}
```

DEL Delete Order

<http://localhost/phprest/api/orders/delete.php>

You can use this endpoint to delete an Order entry from table using an product SKU parameter in the query, eg: To delete Order with SKU=1000 you can use <http://localhost/phprest/api/orders/delete.php?SKU=1000> (<http://localhost/phprest/api/orders/delete.php?SKU=1000>)

Example Request

Delete Order

```
curl --location --request DELETE 'http://localhost/phprest/api/orders/delete.php?SKU=1000'
```

Example Response

200 OK

[Documentation Settings](#) ▼

Body

 Headers (11)

```
{  
  "message": "order was deleted."  
}
```

Places_hold

Access information regarding Holds placed by customers on a specific product from a KMS TOOLS SQL database

GET Read Holds

```
http://localhost/phprest/api/places_hold/read.php
```

You can use this endpoint to read information about all holds placed by customers stored in the database

Example Request

Read Holds

```
curl --location --request GET 'http://localhost/phprest/api/places_hold/read.php'
```

Example Response

200 OK

Body

 Headers (9)

```
⌘  
{  
  "places_hold": [  
    {  
      "customer_id": "1000",  
      "SKU": "1000",  
      "quantity": "1",  
      "date_placed": "2020-12-10",  
      "date_released": "2020-12-17",  
      "price": "10.00"  
    }  
  ]  
}
```

[Documentation Settings](#) ▼[View More](#)

GET Read One Hold

```
http://localhost/phprest/api/places_hold/read_one.php
```

You can use this endpoint to read a Hold entry from table using a customer_id parameter in the query, eg:

To retrieve Hold with customer_id=1000 you can use

```
http://localhost/phprest/api/places_hold/read_one.php?customer_id=1000
```

```
(http://localhost/phprest/api/places_hold/read_one.php?customer_id=1000)
```

Example Request

Read One Hold

```
curl --location --request GET 'http://localhost/phprest/api/places_hold/read_one.php?customer_id=1000'
```

Example Response

200 OK

Body Headers (11)

```
{  
  "customer_id": "1000",  
  "SKU": "1000",  
  "quantity": "1",  
  "date_placed": "2020-12-10",  
  "date_released": "2020-12-17",  
  "price": "10.00"  
}
```




POST Update Hold

```
http://localhost/phprest/api/places_hold/update.php
```

You can update a Hold entry by supplying the all the Hold parameter values for customer_id, SKU, quantity, date_placed, date_released and price with the changes in a raw body

Example Request

Update Hold

```
curl --location --request POST 'http://localhost/phprest/api/places_hold/update.php' \
--data-raw '{
  "customer_id": "1000",
  "SKU": "1000",
  "quantity": "2",
  "date_placed": "2020-12-10",
  "date_released": "2020-12-17",
  "price": "10.00"
}'
```

Example Response

200 OK

Body Headers (11)

```
{
  "message": "places_hold was updated."
}
```

DEL Delete Hold

```
http://localhost/phprest/api/places_hold/delete.php
```

You can use this endpoint to delete a Hold entry from table using an customer_id parameter in the query,

Eg: To delete Hold with customer_id=1000 you can use

[Documentation Settings](#) ▼

http://localhost/phprest/api/places_hold/delete.php?customer_id=1000
(http://localhost/phprest/api/places_hold/delete.php?customer_id=1000)

Example Request

Delete Hold

```
curl --location --request DELETE 'http://localhost/phprest/api/places_hold/delete.php?customer_id=1000'
```

Example Response

200 OK

Body Headers (10)

```
{  
  "message": "places_hold was deleted."  
}
```

POST Create Hold

```
http://localhost/phprest/api/places_hold/create.php
```

You can create a Hold entry by sending a body in the request with the customer_id, SKU, quantity, date_placed, date_released and price parameters to be set to the new Hold

Example Request

Create Hold

```
curl --location --request POST 'http://localhost/phprest/api/places_hold/create.php' \
--data-raw '{
  "customer_id": "1000",
  "SKU": "1000",
  "quantity": "2",
  "date_placed": "2020-12-10",
  "date_released": "2020-12-17",
  "price": "10.00"
}'
```

[Documentation Settings](#) ▼

Example Response

200 OK

[Body](#) [Headers \(10\)](#)

```
{
  "message": "Places_hold created successfully."
}
```

Product

API connection towards sales_associate

POST Create Sales Product

http://localhost:8080/phprest/api/product/create.php

API endpoint that allows user to create information inside the database **product**.

Example Request

create

```
curl --location --request POST 'http://localhost:8080/phprest/api/product/create.php' \
--data-raw '{
  "SKU": "1005",
  "type": "Electric",
  "name": "Lightbulbs",
  "description": "Metal and Glass",
  "additional_info": "Manufacturer4",
  "section": "General",
  "UPC": "1111 1123",
  "regular_price": "6.99".
}
```

[Documentation Settings](#) ▼[View More](#)

Example Response

201 Created

[Body](#) [Headers \(0\)](#)

```
{
  "message": "Product was created."
}
```

DEL Delete Sales Product

```
http://localhost:8080/phprest/api/product/delete.php
```

API endpoint that allows user to delete information inside the database **product**.

ex: Delete SKU = 1000

```
http://localhost:8080/phprest/api/product/delete.php?SKU=1000
(http://localhost:8080/phprest/api/product/delete.php?SKU=1000)
```

Example Request

delete

```
curl --location --request DELETE 'http://localhost:8080/phprest/api/product/delete.php?SKU=1000'
```

Example Response

200 OK

Body Headers (0)

[Documentation Settings](#) ▼

```
{
  "message": "Product was deleted."
}
```

POST Update Sales Product

```
http://localhost:8080/phprest/api/product/update.php
```

API endpoint that allows user to update already given information inside the database **product**.

Example Request

update

```
curl --location --request POST 'http://localhost:8080/phprest/api/product/update.php' \
--data-raw '{
  "SKU": "1005",
  "type": "Electric",
  "name": "Lightbulbs",
  "description": "Metal and Glass",
  "additional_info": "Manufacturer4",
  "section": "General",
  "UPC": "1111 1123",
  "regular price": "5.99".
}
```

[View More](#)

Example Response

200 OK

Body Headers (0)

```
{
  "message": "Product was updated."
}
```

GET Read One Sales Product

http://localhost:8080/phprest/api/product/read_one.php

Documentation Settings ▼

API endpoint that allows user to read specific information inside the database **product**.

ex: Read additional_info = manufacturer2

http://localhost:8080/phprest/api/product/read_one.php?additional_info=manufacturer2
(http://localhost:8080/phprest/api/product/read_one.php?additional_info=manufacturer2)

Example Request

read_one

```
curl --location --request GET 'http://localhost:8080/phprest/api/product/read_one.php?additional_info=manufact
```

Example Response

200 OK

Body Headers (0)

```
{
  "sales_associate": [
    {
      "SKU": "1001",
      "type": "General Tools",
      "name": "Wrench",
      "description": "Metal",
      "additional_info": "manufacturer2",
      "section": "General",
      "UPC": "1111 1111".
```

[View More](#)

GET Read Sales Product

http://localhost:8080/phprest/api/product/read.php

API endpoint that allows user to read all information inside the database **product**.



Example Request

read.php

```
curl --location --request GET 'http://localhost:8080/phprest/api/product/read.php'
```

Example Response

200 OK

Body Headers (0)

```
{
  "sales_associate": [
    {
      "SKU": "1000",
      "type": "General Tools",
      "name": "Hammer",
      "description": "Metal",
      "additional_info": "manufacturer1",
      "section": "General",
      "UPC": "1111 1110".
    }
  ]
}
```

[View More](#)

Sales Associate

API connection towards sales_associate

GET Read One Sales Associate

```
http://localhost:8080/phprest/api/sales_associate/read_one.php
```

Endpoint that allows user to read specific information inside the database **sales_associate**.

ex: Read section = cashier

```
http://localhost:8080/phprest/api/sales_associate/read_one.php?section=cashier
(http://localhost:8080/phprest/api/sales_associate/read_one.php?section=cashier)
```



Example Request

read_one

```
curl --location --request GET 'http://localhost:8080/phprest/api/sales_associate/read_one.php?section=cashier'
```

Example Response

200 OK

Body Headers (0)

```
{
  "sales_associate": [
    {
      "SID": "2",
      "first_name": "B",
      "last_name": "name",
      "username": "emp2",
      "password": "password",
      "section": "cashier"
    }
  ]
}
```

[View More](#)

POST Update Sales Associate

```
http://localhost:8080/phprest/api/sales_associate/update.php
```

API endpoint that allows user to update already given information inside the database **sales_associate**.

Example Request

update


```
curl --location --request POST 'http://localhost:8080/phprest/api/sales_associate/update.php' \
--data-raw '{
  "SID": "5",
  "first_name": "Z",
  "last_name": "name",
  "username": "emp5",
  "password": "hello",
  "section": "cashier"
}'
```

[Documentation Settings](#) ▼

Example Response

200 OK

[Body](#) [Headers \(0\)](#)

```
{
  "message": "sales_associate was updated."
}
```

GET Read Sales Associate

http://localhost:8080/phprest/api/sales_associate/read.php

API endpoint that allows user to read all information inside the database **sales_associate**.

Example Request

read.php

```
curl --location --request GET 'http://localhost:8080/phprest/api/sales_associate/read.php'
```

Example Response

200 OK

[Body](#) [Headers \(0\)](#)

```
⌘  
"sales_associate": [  
  {  
    "SID": "1",  
    "first_name": "C",  
    "last_name": "name",  
    "username": "emp1",  
    "password": "welcome",  
    "section": "storage"  
  },  
  ]
```

[Documentation Settings](#) ▼[View More](#)

DEL Delete Sales Associate

```
http://localhost:8080/phprest/api/sales_associate/delete.php
```

API endpoint that allows user to delete information inside the database **sales_associate**.

ex: Delete password = password

```
http://localhost:8080/phprest/api/sales_associate/delete.php?password=password  
(http://localhost:8080/phprest/api/sales_associate/delete.php?password=password)
```

Example Request

delete

```
curl --location --request DELETE 'http://localhost:8080/phprest/api/sales_associate/delete.php?password=passwo
```

Example Response

200 OK

Body Headers (0)

```
{  
  "message": "sales_associate was deleted."  
}
```

POST Admin Authentication

[Documentation Settings](#) ▼

http://localhost/phprest/api/sales_associate/login.php

BODY raw

```
{
  "username": "admin",
  "password": "admin"
}
```

Example Request

Admin Authentication

```
curl --location --request POST 'http://localhost/phprest/api/sales_associate/login.php' \
--data-raw '{
  "username": "admin",
  "password": "admin"
}'
```

Example Response

Body Headers (0)

```
{
  "message": "Login successful."
}
```

POST Create Sales Associate

http://localhost:8080/phprest/api/sales_associate/create.php

API endpoint that allows user to create information inside the database **sales_associate**.



Example Request

create

```
curl --location --request POST 'http://localhost:8080/phprest/api/sales_associate/create.php' \
--data-raw '{
  "SID": "5",
  "first_name": "Z",
  "last_name": "name",
  "username": "emp5",
  "password": "adSfwe4",
  "section": "storage"
}'
```

Example Response

201 Created

Body Headers (0)

```
{
  "message": "sales_associate was created."
}
```

Ships To

API connection towards ships_to

POST Create Ships To

```
http://localhost:8080/phprest/api/ships_to/create.php
```

API endpoint that allows user to create information inside the database **ships_to**.

Example Request

[Documentation Settings](#) [create](#) ▼

```
curl --location --request POST 'http://localhost:8080/phprest/api/ships_to/create.php' \
--data-raw '{
  "customer_id": "2",
  "order_id": "4",
  "branch_id": "2",
  "recieve_date": "15/12/2020"
}'
```

Example Response

201 Created

[Body](#) [Headers \(0\)](#)

```
{
  "message": "ships_to was created."
}
```

POST Update Ships To

```
http://localhost:8080/phprest/api/ships_to/update.php
```

API endpoint that allows user to update already given information inside the database **ships_to**.

Example Request

[update](#)

```
curl --location --request POST 'http://localhost:8080/phprest/api/ships_to/update.php' \
--data-raw '{
  "customer_id": "2",
  "order_id": "4",
  "branch_id": "2",
  "recieve_date": "17/12/2020"
}'
```

Example Response

200 OK

Body Headers (0)

[Documentation Settings](#) ▼

```
{
  "message": "ships_to was updated."
}
```

GET Read Ships To

```
http://localhost:8080/phprest/api/ships_to/read.php
```

API endpoint that allows user to read all information inside the database **ships_to**.

Example Request

read.php

```
curl --location --request GET 'http://localhost:8080/phprest/api/ships_to/read.php'
```

Example Response

200 OK

Body Headers (0)

```
{
  "ships_to": [
    {
      "customer_id": "1",
      "order_id": "1",
      "branch_id": "1",
      "recieve_date": "11/12/2020"
    },
    {
      "customer id": "2".
```

[View More](#)

GET Read One Ships To

```
http://localhost:8080/phprest/api/ships_to/read_one.php
```

API endpoint that allows user to read specific information inside the database **ships_to**.

[Documentation Settings](#)

ex: Read recieve_date=11/12/2020

http://localhost:8080/phprest/api/ships_to/read_one.php?recieve_date=11/12/2020

(http://localhost:8080/phprest/api/ships_to/read_one.php?recieve_date=11/12/2020)

Example Request

read_one

```
curl --location --request GET 'http://localhost:8080/phprest/api/ships_to/read_one.php?recieve_date=11/12/2020'
```

Example Response

200 OK

Body Headers (0)

```
{
  "ships_to": [
    {
      "customer_id": "1",
      "order_id": "1",
      "branch_id": "1",
      "recieve_date": "11/12/2020"
    },
    {
      "customer id": "3".
```

[View More](#)

DEL Delete Ships To

http://localhost:8080/phprest/api/ships_to/delete.php

Api endpoint that allows user to delete information inside the database **ships_to**.

ex: Delete order_id = 2

Documentation Settings ▼

http://localhost:8080/phprest/api/ships_to/delete.php?order_id=2
(http://localhost:8080/phprest/api/ships_to/delete.php?order_id=2)

Example Request

delete

```
curl --location --request DELETE 'http://localhost:8080/phprest/api/ships_to/delete.php?order_id=2'
```

Example Response

200 OK

Body Headers (0)

```
{  
  "message": "ships_to was deleted."  
}
```

Ships From

API connection towards ships_from

POST Create Ships From

http://localhost:8080/phprest/api/ships_from/create.php

API endpoint that allows user to create information inside the database **ships_from**.

BODY raw


```
{
  "SKU" : 1003,
  "type" : "General Tools",
  "name" : "Hammer",
  "description" : "Hammers objects",
  "additional_info" : "General",
  "UPC" : 1114,
  "regular_price" : 10,
  "sale_price" : 8,
  "club_price" : 6,
```

[Documentation Settings](#) ▼[View More](#)

Example Request

[create](#)

```
curl --location --request POST 'http://localhost:8080/phprest/api/ships_from/create.php' \
--data-raw '{
  "customer_id": "4",
  "order_id": "4",
  "branch_id": "4",
  "ship_date": "24/12/2020"
}'
```

Example Response

201 Created

Body Headers (0)

```
{
  "message": "ships_from was created."
}
```

GET Read Ships From

http://localhost:8080/phprest/api/ships_from/read.php

API endpoint that allows user to read all information inside the database **ships_from**.



Example Request

read.php

```
curl --location --request GET 'http://localhost:8080/phprest/api/ships_from/read.php'
```

Example Response

200 OK

Body Headers (0)

```
{
  "ships_from": [
    {
      "customer_id": "1",
      "order_id": "1",
      "branch_id": "1",
      "ship_date": "06/12/2020"
    },
    {
      "customer id": "2".
```

[View More](#)

DEL Delete Ships From

```
http://localhost:8080/phprest/api/ships_from/delete.php
```

API endpoint that allows user to delete information inside the database **ships_from**.

ex: Delete order_id = 2

```
http://localhost:8080/phprest/api/ships_from/delete.php?order_id=2
(http://localhost:8080/phprest/api/ships_from/delete.php?order_id=2)
```

Example Request

delete

```
curl --location --request DELETE 'http://localhost:8080/phprest/api/ships_from/delete.php?order_id=2'
```

[Documentation Settings](#) ▼

Example Response

200 OK

Body Headers (0)

```
{
  "message": "ships_from was deleted."
}
```

POST Update Ships From

```
http://localhost:8080/phprest/api/ships_from/update.php
```

API endpoint that allows user to update already given information inside the database **ships_from**.

Example Request

update

```
curl --location --request POST 'http://localhost:8080/phprest/api/ships_from/update.php' \
--data-raw '{
  "customer_id": "4",
  "order_id": "4",
  "branch_id": "4",
  "ship_date": "23/12/2020"
}'
```

Example Response

200 OK

Body Headers (0)

```
{
  "message": "ships_from was updated."
}
```

GET Read One Ships From

[Documentation Settings](#) ▼

```
http://localhost:8080/phprest/api/ships_from/read_one.php
```

API endpoint that allows user to read specific information inside the database **ships_from**.

ex: Read ship_date = 09/12/2020

```
http://localhost:8080/phprest/api/ships_from/read_one.php?ship_date=09/12/2020
```

```
(http://localhost:8080/phprest/api/ships_from/read_one.php?ship_date=09/12/2020)
```

Example Request

read_one

```
curl --location --request GET 'http://localhost:8080/phprest/api/ships_from/read_one.php?ship_date=09/12/2020'
```

Example Response

200 OK

Body Headers (0)

```
{
  "ships_from": [
    {
      "customer_id": "2",
      "order_id": "2",
      "branch_id": "2",
      "ship_date": "09/12/2020"
    },
    {
      "customer id": "3".
```

[View More](#)

Store Branch

API connection towards store_branch

GET Read Store Branch

```
http://localhost:8080/phprest/api/store_branch/read.php
```

API endpoint that allows user to read all information inside the database **store_branch**.

Example Request

read.php

```
curl --location --request GET 'http://localhost:8080/phprest/api/store_branch/read.php'
```

Example Response

200 OK

Body Headers (0)

```
{
  "store_branch": [
    {
      "branch_id": "1",
      "email": "branch1@KMS-tools.ca",
      "phone_no": "111 1111 1111",
      "province": "Alberta",
      "city": "Calgary",
      "street_no": "22 ave SE"
    }
  ]
}
```

[View More](#)

POST Create Store Branch

```
http://localhost:8080/phprest/api/store_branch/create.php
```

API endpoint that allows user to create information inside the database **store_branch**.



Example Request

create

```
curl --location --request POST 'http://localhost:8080/phprest/api/store_branch/create.php' \
--data-raw '{
  "branch_id": "4",
  "email": "branch4@KMS-tools.ca",
  "phone_no": "111 1111 1114",
  "province": "Alberta",
  "city": "Red Deer",
  "street_no": "12 ave NW"
}'
```

Example Response

201 Created

Body Headers (0)

```
{
  "message": "store_branch was created."
}
```

POST Update Store Branch

http://localhost:8080/phprest/api/store_branch/update.php

API endpoint that allows user to update already given information inside the database **store_branch**.

Example Request

update

```
curl --location --request POST 'http://localhost:8080/phprest/api/store_branch/update.php' \
--data-raw '{
  "branch_id": "4",
  "email": "branch4@KMS-tools.ca",
  "phone_no": "111 1111 1114",
  "province": "Alberta",
  "city": "Red Deer",
  "street_no": "10 ave NW"
}'
```

Documentation Settings ▼

Example Response

200 OK

Body Headers (0)

```
{
  "message": "store_branch was updated."
}
```

GET Read One Store Branch

```
http://localhost:8080/phprest/api/store_branch/read_one.php
```

API endpoint that allows user to read specific information inside the database **store_branch**.

ex: Read city = Calgary

```
http://localhost:8080/phprest/api/store_branch/read_one.php?city=Calgary
(http://localhost:8080/phprest/api/store_branch/read_one.php?city=Calgary)
```

Example Request

read_one

```
curl --location --request GET 'http://localhost:8080/phprest/api/store_branch/read_one.php?city=Calgary'
```

200 OK

Example Response

Body Headers (0)

[Documentation Settings](#) ▼

```
{
  "store_branch": [
    {
      "branch_id": "1",
      "email": "branch1@KMS-tools.ca",
      "phone_no": "111 1111 1111",
      "province": "Alberta",
      "city": "Calgary",
      "street_no": "22 ave SE"
    }
  ]
}
```

[View More](#)

DEL Delete Store Branch

```
http://localhost:8080/phprest/api/store_branch/delete.php
```

API endpoint that allows user to delete information inside the database **store_branch**.

ex: Delete branch_id = 4

```
http://localhost:8080/phprest/api/store_branch/delete.php?branch_id=4
(http://localhost:8080/phprest/api/store_branch/delete.php?branch_id=4)
```

Example Request

delete

```
curl --location --request DELETE 'http://localhost:8080/phprest/api/store_branch/delete.php?branch_id=4'
```

Example Response

200 OK

Body Headers (0)

```
{
  "message": "store_branch was deleted."
}
```




Supplier

API connection towards supplier

POST Update Supplier

```
http://localhost:8080/phprest/api/supplier/update.php
```

API endpoint that allows user to update already given information inside the database **supplier**.

Example Request

update

```
curl --location --request POST 'http://localhost:8080/phprest/api/supplier/update.php' \
--data-raw '{
  "id": "4",
  "name": "Terry the supplier",
  "phone_no": "444 4444 4444"
  "email": "company2@supplier.ca"
}'
```

Example Response

200 OK

Body Headers (0)

```
{
  "message": "supplier was updated."
}
```

GET Read One Supplier

http://localhost:8080/phprest/api/supplier/read_one.php

Documentation Settings ▼

API endpoint that allows user to read specific information inside the database **supplier**.

ex: Read email = company1@supplier.ca (mailto:company1@supplier.ca)

http://localhost:8080/phprest/api/supplier/read_one.php?email=company1@supplier.ca
(http://localhost:8080/phprest/api/supplier/read_one.php?email=company1@supplier.ca)

Example Request

read_one

```
curl --location --request GET 'http://localhost:8080/phprest/api/supplier/read_one.php?email=company1@supplier'
```

Example Response

200 OK

Body Headers (0)

```
{
  "supplier": [
    {
      "id": "1",
      "name": "Jerry the supplier",
      "phone_no": "111 1111 1111",
      "email": "company1@supplier.ca"
    },
    {
      "id": "2".
```

View More

POST Create Supplier

http://localhost:8080/phprest/api/supplier/create.php

API endpoint that allows user to create information inside the database **supplier**.



Example Request

create

```
curl --location --request POST 'http://localhost:8080/phprest/api/supplier/create.php' \
--data-raw '{
  "id": "4",
  "name": "Terry the supplier",
  "phone_no": "444 4444 4444"
  "email": "company3@supplier.ca"
}'
```

Example Response

201 Created

Body Headers (0)

```
{
  "message": "supplier was created."
}
```

GET Read Supplier

```
http://localhost:8080/phprest/api/supplier/read.php
```

API endpoint that allows user to read all information inside the database **supplier**.

Example Request

read.php

```
curl --location --request GET 'http://localhost:8080/phprest/api/supplier/read.php'
```

Example Response

200 OK

Body Headers (0)



```
"supplier": [  
  {  
    "id": "1",  
    "name": "Jerry the supplier",  
    "phone_no": "111 1111 1111",  
    "email": "company1@supplier.ca"  
  },  
  {  
    "id": "2".
```

[View More](#)

DEL Delete Supplier

```
http://localhost:8080/phprest/api/supplier/delete.php
```

API endpoint that allows user to delete information inside the database **supplier**.

ex: Delete id = 4

```
http://localhost:8080/phprest/api/supplier/delete.php?id=4  
(http://localhost:8080/phprest/api/supplier/delete.php?id=4)
```

Example Request

delete

```
curl --location --request DELETE 'http://localhost:8080/phprest/api/supplier/delete.php?id=4' \  
--data-raw ''
```

Example Response

200 OK

Body Headers (0)

```
{  
  "message": "supplier was deleted."  
}
```

Transfer Order

[Documentation Settings](#) ▼

API connection towards transfer_order

POST Create Transfer Order

```
http://localhost:8080/phprest/api/transfer_order/create.php
```

API endpoint that allows user to create information inside the database **transfer_order**.

Example Request

[create](#)

```
curl --location --request POST 'http://localhost:8080/phprest/api/transfer_order/create.php' \
--data-raw '
{
  "customer_id": "21",
  "order_id": "22",
  "date_ordered": "19/12/2020"
  "invoiced": "total owed $21"
}
```

Example Response

[201 Created](#)

Body Headers (0)

```
{
  "message": "transfer_order was created."
}
```

GET Read One Transfer Order

http://localhost:8080/phprest/api/transfer_order/read_one.php

Documentation Settings ▼

API endpoint that allows user to read specific information inside the database **transfer_order**.

ex: Read date_ordered = 15/12/2020

http://localhost:8080/phprest/api/transfer_order/read_one.php?date_ordered=15/12/2020
(http://localhost:8080/phprest/api/transfer_order/read_one.php?date_ordered=15/12/2020)

Example Request

read_one

```
curl --location --request GET 'http://localhost:8080/phprest/api/transfer_order/read_one.php?date_ordered=15/1
```

Example Response

200 OK

Body Headers (0)

```
{
  "transfer_order": [
    {
      "customer_id": "12",
      "order_id": "52",
      "date_ordered": "15/12/2020",
      "invoiced": "total owed $26"
    },
    {
      "customer id": "13".
```

View More

POST Update Transfer Order

http://localhost:8080/phprest/api/transfer_order/update.php

API endpoint that allows user to update already given information inside the database **transfer_order**.



Example Request

update

```
curl --location --request POST 'http://localhost:8080/phprest/api/transfer_order/update.php' \
--data-raw '{
  "customer_id": "21",
  "order_id": "22",
  "date_ordered": "19/12/2020"
  "invoiced": "total owed $11"
}'
```

Example Response

200 OK

Body Headers (0)

```
{
  "message": "transfer_order was updated."
}
```

GET Read Transfer Order

http://localhost:8080/phprest/api/transfer_order/read.php

API endpoint that allows user to read all information inside the database **transfer_order**.

Example Request

read.php

```
curl --location --request GET 'http://localhost:8080/phprest/api/transfer_order/read.php'
```

Example Response

200 OK

Body Headers (0)

```
⌘
{
  "transfer_order": [
    {
      "customer_id": "1",
      "order_id": "50",
      "date_ordered": "12/12/2020",
      "invoiced": "total owed $240"
    },
    {
      "customer id": "12".
    }
  ]
}
```

[Documentation Settings](#) ▼[View More](#)

DEL Delete Transfer Order

```
http://localhost:8080/phprest/api/transfer_order/delete.php
```

API endpoint that allows user to delete information inside the database **transfer_order**.

ex: Delete order_id = 22

```
http://localhost:8080/phprest/api/transfer_order/delete.php?order_id=22
```

```
(http://localhost:8080/phprest/api/transfer_order/delete.php?order_id=22)
```

Example Request

delete

```
curl --location --request DELETE 'http://localhost:8080/phprest/api/transfer_order/delete.php?order_id=22'
```

Example Response

200 OK

Body Headers (0)

```
{
  "message": "transfer_order was deleted."
}
```




Works For

API connection towards works_for

DEL Delete Works For

```
http://localhost:8080/phprest/api/works_for/delete.php
```

Api endpoint that allows user to delete information inside the database **works_for**.

```
ex: Delete branch_id = 2
```

```
http://localhost:8080/phprest/api/works_for/delete.php?branch_id=2  
(http://localhost:8080/phprest/api/works_for/delete.php?branch_id=2)
```

Example Request

delete

```
curl --location --request DELETE 'http://localhost:8080/phprest/api/works_for/delete.php?branch_id=2'
```

Example Response

200 OK

Body Headers (0)

```
{  
  "message": "works_for was deleted."  
}
```

POST Create Works For

```
http://localhost:8080/phprest/api/works_for/create.php
```

[Documentation Settings](#) ▼

API endpoint that allows user to create information inside the database **works_for**.

Example Request

[create](#)

```
curl --location --request POST 'http://localhost:8080/phprest/api/works_for/create.php' \
--data-raw '{
  "branch_id": "2",
  "SID": "200",
  "shift": "day"
}'
```

Example Response

[201 Created](#)

[Body](#) [Headers \(0\)](#)

```
{
  "message": "works_for was created."
}
```

POST Update Works For

```
http://localhost:8080/phprest/api/works_for/update.php
```

API endpoint that allows user to update already given information inside the database **works_for**.

Example Request

[update](#)

```
curl --location --request POST 'http://localhost:8080/phprest/api/works_for/update.php' \
--data-raw '{
  "branch_id": "2",
  "SID": "200",
  "shift": "night"
}'
```

[Documentation Settings](#) ▼

Example Response

200 OK

[Body](#) [Headers \(0\)](#)

```
{
  "message": "works_for was updated."
}
```

GET Read Works For

```
http://localhost:8080/phprest/api/works_for/read.php
```

API endpoint that allows user to read all information inside the database **works_for**.

Example Request

read.php

```
curl --location --request GET 'http://localhost:8080/phprest/api/works_for/read.php'
```

Example Response

200 OK

[Body](#) [Headers \(8\)](#)

```
f
{
  "works_for": [
    {
      "branch_id": "1",
      "SID": "100",
      "shift": "night"
    },
    {
      "branch_id": "1",
      "SID": "50".
    }
  ]
}
```

[Documentation Settings](#) ▼[View More](#)

GET Read One Works For

```
http://localhost:8080/phprest/api/works_for/read_one.php
```

API endpoint that allows user to read specific information inside the database **works_for**.

ex: Read branch_id = 1

```
http://localhost:8080/phprest/api/works_for/read_one.php?branch_id=1
(http://localhost:8080/phprest/api/works_for/read_one.php?branch_id=1)
```

Example Request

[read_one](#)

```
curl --location --request GET 'http://localhost:8080/phprest/api/works_for/read_one.php?branch_id=1'
```

Example Response

200 OK

Body Headers (0)



```
"works_for":[
  {
    "branch_id": "1",
    "SID":"100",
    "shift":"night"
  },
  {
    "branch_id": "1".
    "STD":"50"
```

View More