API Document- SELECT * FROM Chief koders

Please Note: All APIs accept and return JSON object and are POST requests.

Login:

Parameter	Required	Type	Description
api	*	String	Name of API (login)
Email	*	String	Email user signed up with
Password	*	String	User Password

Return Type:

```
{
 "Status": "success",
"timestamp":" 1715149632",
"data": {
"message": "Login successful",
"apikey": "206cb7c99345047d70f152dfa5579dcd
"name": "John",
"surname": "Doe"
        }
}
```

Register:

Parameter Name	Required	Type	Description
api	*	String	Name of API (register)
Name	*	String	User's first name
Surname	*	String	User's surname

Email	*****	**String**	**User’s email (validated to be an email and unique in table)**
password	*****	**String**	**User’s password (validated to be 8 characters, contain upper and lower cased letters and a special character)**
PhoneNumber	*****	**String**	**A user’s phone number has to be unique**

Return Type:

```
{
“status”:”success”,
“timestamp”:”17267242742”,
“data”: {
            “message”: “Registration successful”,
            “apikey”:” 5c331d9c15d564d3d0de0f1f2937b92e"
      }
}
```

GetAllProducts:

Paramete r	Require d	Type	Description
type	*	String	Name of API (GetAllProducts)
api	*	String	User’s API key
return	*	String/Arra y	Fields to return (“*” or array of field names)

limit		Int	A number between 1 and 800 indicating how many results should be returned
sort		String	The results can be sorted by any of the following attributes: [‘category’,’price’,’brand’,’country_of_origin’]
order		String	The results can be ordered by "ASC" or "DESC" for ascending or descending respectively
search		JSON object	An object where the keys are columns of the data and the values are the search terms. Columns can be any of the following: ['id', 'title', 'brand', 'description', 'categories','manufacturer', 'department', 'features', 'country_of_origin'] Note: You can search multiple fields at a time.
compare		JSON object	Comapre the prices of a certain product by name and then return the product with its associated retailer.

Return Type:

{

“status”: “success”,

“data”:[“ ....returned data here....”]

}

GetAllRetailers

Parameter Name	**Required**	**Type**	**Description**
api	*****	**String**	**Users’s api key**
apikey	*****	**String**	**The user making the request**

limit		Int	A number between 1 and 20 indicating how many results should be returned

Return Type:

```
{
“status”:”success”,
“timestamp”: 1233216353,
“data”:[
{
“RetailerID”: “1”,
“Name”: “Takealot”,
“URL”: "https://www.takealot.com"
}
        ]
}
```

GetDistinct:

Parameter Name	Required	Type	Description
api	*	String	Specifies the request
apikey	*	String	The user’s apikey that is making the request
table	*	String	The table from which the distinct values will be fetched(So you

			specify what do you distinctly want, products, retailers ect)
limit		Int	Can optionally limit the amount if distinct values returned
field	If used, has to be used with the search parameter	String	The column name that will be searched
search	If used, has to be used with the field parameter	String	A search is performed in a specified field

This request can be used for a wide variety of tasks, firstly it can return any sensible distinct values from our tables, we can set a limit on how many distinct values we want returned, we can search for a specific value in a specific field and lastly we can join tables on distinct values, meaning, with one distinct value we can get different information from different tables related to this distinct value.

Return Type (I’ll be giving examples with the products table, but they can be made on any table):

```
{
“status”:”success”,
“timestamp” : 1736288384394083
“data”: {
“table”: “products”,
“count”: (depends on how many values there are),
“results”: [ {
“ProductID”: “1”,
“RID”: “1”,
“quantity”: “878”,
“price”:”578.53”,
```

“remaining”:”878”

}

Rating:

Parameter Name	**Required**	**Type**	**Description**
api	*****	**String**	**Name of API (rating)**
operation	*****	**String**	**It is either get/set meaning a user can leave a review or you can get reviews(this can return either all the reviews left by a user on different products, or all the reviews on one product)**
apikey	*****	**String**	**The user that is interacting (leaving a review or looking at reviews)**
productID		**Int**	**If this parameter is added then all the reviews of that product is returned**

```
{
“status”: “success”,
“timestamp”: 17394743938,
“data”: {
```

```
“userId”: “a1b2c3d4e5",
“userRatings”: [
{
"productId": "111",
            "Rating": "2",
            "Comment": "Not that great!",
            "Date": "2025-05-26",
            "productName": "MaxGear Acrylic Donation Box, 9.8\" x 9.8\" x 9.8\"
Square Suggestion Box with Lock, Large Comment Case"
         },
         {
            "productId": "1",
            "Rating": "5",
            "Comment": "Excellent tea!",
            "Date": "2025-01-15",
            "productName": "Soundcore Select 4 Go Bluetooth Shower Speaker
by Anker, IP67 Waterproof/Dustproof, Portable Speaker"
         }
      ],
      "totalUserRatings": 2
   }
}
```

Logout:

Parameter Name	**Required**	**Type**	**Description**

api	*****	**String**	**Name of the API**
apikey	*****	**String**	**The apikey of the user logging out**

```
{
“status”: “success”,
“timestamp”: 1736372464839,
“data”: {
“message”: “Logout successful”
    }
}
```

Wishlist:

Parameter	Required	Type	Description
api	*	String	Name of the API (wishlist)
apikey	*	String	Apikey of the user that is interacting with his/her wishlist
ProductID	(Required for operation set)	Int	The id of the product being put into the wishlist
Operation	*	String	Has to be one of 3 options: Get/Set/Unset Get: Returns an array of productID’s Set: How a user sets their

			favourite product Unset: Removes the productid from the user’s favourites

Cart
Wishlist
Checkout
Compare
AdminControls {Has different optional parameters edit, delete}