**API Documentation**

Contents

[Users 4](#_Toc198814058)

[Registration 4](#_Toc198814059)

[Request Type 4](#_Toc198814060)

[Responses 5](#_Toc198814061)

[Login 5](#_Toc198814062)

[Request Type 5](#_Toc198814063)

[Responses 5](#_Toc198814064)

[Product 6](#_Toc198814065)

[Getting all the products. 6](#_Toc198814066)

[Request Type 6](#_Toc198814067)

[Responses 6](#_Toc198814068)

[Adding product 7](#_Toc198814069)

[Request Type 7](#_Toc198814070)

[Responses 8](#_Toc198814071)

[Deleting product 9](#_Toc198814072)

[Request Type 9](#_Toc198814073)

[Responses 9](#_Toc198814074)

[Edit the products 10](#_Toc198814075)

[Request Type 10](#_Toc198814076)

[Responses 11](#_Toc198814077)

[Filter the products 12](#_Toc198814078)

[Request Type 12](#_Toc198814079)

[Responses 13](#_Toc198814080)

[Rating 13](#_Toc198814081)

[Adding Rating 13](#_Toc198814082)

[Request Type 13](#_Toc198814083)

[Responses 14](#_Toc198814084)

[Delete Rating 14](#_Toc198814085)

[Request Type 14](#_Toc198814086)

[Responses 14](#_Toc198814087)

[Edit Rating 15](#_Toc198814088)

[Request Type 15](#_Toc198814089)

[Responses 15](#_Toc198814090)

[GetStores 16](#_Toc198814091)

[Request Type 16](#_Toc198814092)

[Responses 16](#_Toc198814093)

[When a user follows a store 17](#_Toc198814094)

[Request Type 17](#_Toc198814095)

[Responses 17](#_Toc198814096)

[Retrieve stores that user follows 18](#_Toc198814097)

[Request Type 18](#_Toc198814098)

[Responses 18](#_Toc198814099)

[When a user unfollows the store 19](#_Toc198814100)

[Request Type 19](#_Toc198814101)

[Responses 19](#_Toc198814102)

[When a store owner adds a store 20](#_Toc198814103)

[Request Type 20](#_Toc198814104)

[Responses 20](#_Toc198814105)

[SavePrefrences 22](#_Toc198814106)

[Request Type 22](#_Toc198814107)

[Responses 22](#_Toc198814108)

# Users

## Registration

### Request Type

* **Method**: POST
* **Content-Type**: application/json

|  |  |
| --- | --- |
| Parameter | Description |
| name |  |
| email |  |
| password |  |
| description |  |
| launch\_date |  |
| thumbnail |  |
| category |  |
| brand\_id |  |

### Responses

|  |  |
| --- | --- |
| Success | {  "status": "success",  "message" => "Product deleted successfully in the database"  } |
| Error | {  "status": "error",  "Failed to delete product from the database"  } |

## Login

### Request Type

* Method: POST
* Content-Type: application/json

Type: Login

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Email | The email of the user |
| Password | The password of the user |

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode([                  "status" => "success",                  "message" => "Login successful",                  "user" => [                      "user\_id" => $user['user\_id'],                      "name" => $user['name'],                      "email" => $user['email'],                      "user\_type" => $user['user\_type']                  ]              ]); |
| Error in login | echo json\_encode(["status" => "error", "message" => "Invalid email or password"]); |
| If email or password is missing | echo json\_encode(["status" => "error", "message" => "Missing required fields"]); |

# Product

## Getting all the products.

### Request Type

* Method: POST
* Content-Type: application/json

Type: GetAllProducts

No parameters are required here.

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode([              "status" => "success",              "message" => "Products fetched successfully",              "data" => $products          ]); |
| Error | echo json\_encode([              "status" => "error",              "message" => "Failed to fetch products from the database."          ]); |

## Adding product

### Request Type

* Method: POST
* Content-Type: application/json

Type: AddProduct

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| title | The title of the product . |
| price | The price of the product . |
| product\_link | The product link of the product . |
| description | The description of the product. |
| launch\_date | The launch date of the product. |
| thumbnail | The thumbnail of the product. |
| category | The category of the product. |
| brand\_id | The brand\_id of the product. |
| Store\_id | The store\_id of the product |

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode([              "status" => "success",              "message" => "Product added successfully to the database",          ]); |
| Error | echo json\_encode([              "status" => "error",              "message" => "Failed to add product to the database."          ]); |
| If a field is missing | echo json\_encode([                  "status" => "error",                  "message" => "Missing required field: $field"              ]); |
| If there is an invalid field | echo json\_encode([                  "status" => "error",                  "message" => "Invalid field: $key"              ]); |
| Adding a product when you are not a store owner | echo json\_encode([              "status" => "error",              "message" => "Sorry, You are not a store owner of this product"          ]); |

## Deleting product

### Request Type

* Method: POST
* Content-Type: application/json

Type: DeleteProduct

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| prod\_id | The product\_id of the product |

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode([              "status" => "success",              "message" => "Product deleted successfully in the database"          ]); |
| Error | echo json\_encode([              "status" => "error",              "message" => "Failed to delete product from the database"          ]); |
| If the fields are missing | echo json\_encode([              "status" => "error",              "message" => "Missing required fields"          ]); |
| Deleting a product when you are not a store owner of the product. | echo json\_encode([              "status" => "error",              "message" => "Sorry, You are not a store owner of this product"          ]); |
| When deleting a product that’s not inside your store. | echo json\_encode([              "status" => "error",              "message" => "This product does not belong to your store."          ]); |

## Edit the products

### Request Type

* Method: POST
* Content-Type: application/json

Type: EditProduct

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Product\_id | The product\_id of the product. |
| title | The title of the product . |
| price | The price of the product . |
| product\_link | The product link of the product . |
| description | The description of the product. |
| launch\_date | The launch date of the product. |
| thumbnail | The thumbnail of the product. |
| category | The category of the product. |
| brand\_id | The brand\_id of the product. |
| Store\_id | The store\_id of the product |

### Responses

|  |  |
| --- | --- |
| Success | {  "status": "success",  "message" => "Product deleted successfully in the database"  } |
| Error | {  "status": "error",  "Failed to delete product from the database"  } |
| Editing a product when you are not a store owner of the product. | echo json\_encode([              "status" => "error",              "message" => "Sorry, You are not a store owner of this product"          ]); |
| When Editing a product that’s not inside your store. | echo json\_encode([              "status" => "error",              "message" => "This product does not belong to your store."          ]); |

## Filter the products

### Request Type

* Method: POST
* Content-Type: application/json

Type: GetFilteredProducts

**The user can choose what to filter by using the following options:**

* **The brand ID** can be sent through brand\_id.
* **The category** can be sent through category.
* **The minimum price** can be sent through min\_price.
* **The maximum price** can be sent through max\_price.
* **To search by title**, the query must be sent through search.

The results will be sent through the **data**.

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode([          "status" => "success",          "data" => $products      ]); |

# Rating

## Adding Rating

### Request Type

* Method: POST
* Content-Type: application/json

Type: SubmitRating

|  |  |
| --- | --- |
| Parameter | Description |
| prod\_id | The product\_id of the product |
| rating | The rating should be between 1 to 5. |
| comment | The comment of the product |
| User\_id | The user\_id of the user that created the product. |

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode(["status" => "success", "message" => "Rating submitted successfully."]); |
| Error | echo json\_encode(["status" => "error", "message" => "Failed to submit rating."]); |
| Missing fields | echo json\_encode([              "status" => "error",              "message" => "Missing required fields"          ]); |
| Check if the user is signed-in | echo json\_encode([              "status" => "error",              "message" => "Authentication failed. Invalid credentials."          ]); |

## Delete Rating

### Request Type

* **Method**: POST
* **Content-Type**: application/json

Type: DeleteRating

|  |  |
| --- | --- |
| Parameter | Description |
| rating\_id | The rating\_id of the product |

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode(["status" => "success", "message" => "Rating deleted successfully."]); |
| Error | echo json\_encode(["status" => "error", "message" => "Failed to delete rating."]); |

## Edit Rating

### Request Type

* **Method**: POST
* **Content-Type**: application/json

Type: EditRating

|  |  |
| --- | --- |
| Parameter | Description |
| rating\_id | The rating\_id of the product |
| rating | The rating of the product between 1 to 5 |
| comment | The comment of the product |

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode(["status" => "success", "message" => "Rating updated successfully."]); |
| Error | echo json\_encode(["status" => "error", "message" => "Failed to update rating."]); |
| Can only edit his own comment | echo json\_encode(["status" => "error", "message" => "Unauthorized: You can only edit your own rating."]); |

# GetStores

### Request Type

* Method: POST
* Content-Type: application/json

Type: GetStores

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| apiKey | The api\_Key of the user |

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode([          "status" => "success",          "message" => "All stores available, retrieved successfully.",          "data" => $stores      ]); |
| Error | echo json\_encode([              "status" => "error",              "message" => "Authentication failed. Invalid credentials."          ]); |
| Missing fields | echo json\_encode([              "status" => "error",              "message" => "Missing required fields"          ]); |
| Checks if the user exists | echo json\_encode([              "status" => "error",              "message" => "Authentication failed. Invalid credentials."          ]); |
| When there is no stores | echo json\_encode([              "status" => "success",              "message" => "There are currently no stores available",              "data" => []          ]); |

# When a user follows a store

### Request Type

* Method: POST
* Content-Type: application/json

Type: Follow

This checks which user follows which store

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| apiKey | The api\_Key of the user |
| store\_id | The store\_id of the store |

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode([          "status" => "success",          "message" => "Successfully followed store",          "data" => $store\_id      ]); |
| Error | die(json\_encode([              "status" => "error",              "message" => "Failed to follow store"          ])); |
| Fields missing | echo json\_encode(["status" => "error",              "message" => "Missing required fields"          ]); |
| Checks if user is in database | echo json\_encode([              "status" => "error",              "message" => "Authentication failed. Invalid credentials."          ]); |

### Retrieve stores that user follows

### Request Type

* Method: POST
* Content-Type: application/json

Type: GetFollowing

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| apiKey | The api\_Key of the user |

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode([              "status" => "success",              "message" => "Successfully retrieved followed stores",              "data" => $stores          ]); |
| Error | die(json\_encode([              "status" => "error",              "message" => "Failed to retrieve followed stores"          ])); |
| Missing fields | echo json\_encode([              "status" => "error",              "message" => "Missing required fields"          ]); |
| Checks if user is in database | echo json\_encode([              "status" => "error",              "message" => "Authentication failed. Invalid credentials."          ]); |
| When the user does not follow any stores | echo json\_encode([                  "status" => "success",                  "message" => "User is not following any stores",                  "data" => []              ]); |

### When a user unfollows the store

### Request Type

* Method: POST
* Content-Type: application/json

Type: Unfollow

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| apiKey | The api\_Key of the user |
| store\_id | The store\_id of the store |

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode([          "status" => "success",          "message" => "Successfully removed follow",          "data" => $store\_id      ]); |
| Error | echo json\_encode([              "status" => "error",              "message" => "Failed to remove follow"          ]); |
| Missing fields | echo json\_encode([              "status" => "error",              "message" => "Missing required fields"          ]); |
| Checks if user is in database | echo json\_encode([              "status" => "error",              "message" => "Authentication failed. Invalid credentials."          ]); |

### When a store owner adds a store

### Request Type

* Method: POST
* Content-Type: application/json

Type: RegisterStoreOwner

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| apiKey | The api\_Key of the user |
| store\_id | The store\_id of the store |

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode([          "status" => "success",          "message" => "Successfully added store and assigned owner"      ]); |
| Error | echo json\_encode([              "status" => "error",              "message" => "Failed to add owner to the created store"          ]); |
| Missing fields | echo json\_encode([              "status" => "error",              "message" => "Missing required fields"          ]); |
| Checks if user is in database | echo json\_encode([              "status" => "error",              "message" => "Authentication failed. Invalid credentials."          ]); |
| Error if the store failed to be added to the store database | echo json\_encode([              "status" => "error",              "message" => "Failed to add store"          ]); |
| Error if the store failed to be added to the store\_ownwer databse | echo json\_encode([              "status" => "error",              "message" => "Failed to add owner to the created store"          ]); |

# SavePrefrences

### Request Type

* Method: POST
* Content-Type: application/json

Type: SavePrefrences

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| apiKey | The api\_Key of the user |
| min\_price | The min price |
| max\_price | The max price |
| theme | The theme of the user |

### Responses

|  |  |
| --- | --- |
| Success | echo json\_encode(["status" => "success", "message" => "Updated user settings"]); |
| Error | echo json\_encode(["status" => "error", "message" => "Error updating the user settings"]); |
| Missing fields | echo json\_encode([              "status" => "error",              "message" => "Missing required fields"          ]); |