Fake Store API — Internal Reference

Base URL: **https://fakestoreapi.com**

Source compiled from provided docs. Use as a single source of truth for the MCP implementation.

Products

Get all products

Method: GET **Path**: /products

Retrieve a list of all available products.

Responses

• **200** — Success

```
[
  {
    "id": "integer",
    "title": "string",
    "price": "number (float)",
    "description": "string",
    "category": "string",
    "image": "string (uri)"
  }
]
```

• **400** — Bad request

Examples

js

```
fetch('https://fakestoreapi.com/products')
  .then(r=>r.json())
  .then(console.log)
```

curl

```
curl -s https://fakestoreapi.com/products
```

Add a new product

Method: POST **Path**: /products

Create a new product.

Request Body (application/json)

```
{
  "id": "integer",
  "title": "string",
  "price": "number (float)",
  "description": "string",
  "category": "string",
  "image": "string (uri)"
}
```

Responses

• **201** — Product created successfully

```
{
  "id": "integer",
  "title": "string",
  "price": "number (float)",
  "description": "string",
  "category": "string",
  "image": "string (uri)"
}
```

• **400** — Bad request

Examples

curl

```
curl -s -X POST https://fakestoreapi.com/products -H 'Content-Type: application/json' -d '{"id":0,"
title":"string","price":0.1,"description":"string","category":"string","image":"http://example.com"
}'
```

Get a single product

Method: GET   **Path**: /products/{id}

Retrieve details of a specific product by ID.

Path Param	Type/Notes
id	integer

Responses

• **200** — Success

```
{
  "id": "integer",
  "title": "string",
  "price": "number (float)",
  "description": "string",
  "category": "string",
  "image": "string (uri)"
}
```

• **400** — Bad request

Examples

js

```js
fetch('https://fakestoreapi.com/products/1')
  .then(r=>r.json())
  .then(console.log)
```

Update a product

Method: PUT   **Path**: /products/{id}

Update an existing product by ID.

Path Param	Type/Notes
id	integer

Request Body (application/json)

```
{
  "id": "integer",
  "title": "string",
  "price": "number (float)",
  "description": "string",
  "category": "string",
  "image": "string (uri)"
}
```

Responses

• **200** — Product updated successfully

```
{
  "id": "integer",
  "title": "string",
  "price": "number (float)",
  "description": "string",
  "category": "string",
  "image": "string (uri)"
}
```

• **400** — Bad request

Delete a product

Method: DELETE   **Path**: /products/{id}

Delete a specific product by ID.

Path Param	Type/Notes
id	integer

Responses

• **200** — Product deleted successfully

• **400** — Bad request

Examples

js

```js
fetch('https://fakestoreapi.com/products/1', { method: 'DELETE' })
  .then(r=>r.json())
  .then(console.log)
```

Carts

Get all carts

Method: GET **Path**: /carts

Retrieve a list of all available carts.

Responses

• **200** — Success

```
[
  {
    "id": "integer",
    "userId": "integer",
    "products": "array"
  }
]
```

• **400** — Bad request

Examples

js

```
fetch('https://fakestoreapi.com/carts')
  .then(r=>r.json())
  .then(console.log)
```

Add a new cart

Method: POST **Path**: /carts

Create a new cart.

Request Body (application/json)

```
{
  "id": "integer",
  "userId": "integer",
  "products": "array"
}
```

Responses

• **201** — Cart created successfully

```
{
  "id": "integer",
  "userId": "integer",
  "products": "array"
}
```

• **400** — Bad request

Get a single cart

Method: GET **Path**: /carts/{id}

Retrieve details of a specific cart by ID.

Path Param	Type/Notes
id	integer

Responses

• **200** — Success

```
{
  "id": "integer",
  "userId": "integer",
  "products": "array"
}
```

• **400** — Bad request

Update a cart

Method: PUT **Path**: /carts/{id}

Update an existing cart by ID.

Path Param	Type/Notes
id	integer

Request Body (application/json)

```
{
  "id": "integer",
  "userId": "integer",
  "products": "array"
}
```

Responses

• **200** — Cart updated successfully

```
{
  "id": "integer",
  "userId": "integer",
  "products": "array"
}
```

• **400** — Bad request

Delete a cart

Method: DELETE   **Path**: /carts/{id}

Delete a specific cart by ID.

Path Param	Type/Notes
id	integer

Responses

• **200** — Cart deleted successfully

• **400** — Bad request

Examples

js

```
fetch('https://fakestoreapi.com/carts/1', { method: 'DELETE' })
  .then(r=>r.json())
  .then(console.log)
```

Users

Get all users

Method: GET **Path**: /users

Retrieve a list of all users.

Responses

• **200** — Success

```
[
  {
    "id": "integer",
    "username": "string",
    "email": "string",
    "password": "string"
  }
]
```

• **400** — Bad request

Add a new user

Method: POST **Path**: /users

Create a new user.

Request Body (application/json)

```
{
  "id": "integer",
  "username": "string",
  "email": "string",
  "password": "string"
}
```

Responses

• **201** — User created successfully

```
{
  "id": "integer",
  "username": "string",
  "email": "string",
  "password": "string"
}
```

• **400** — Bad request

Get a single user

Method: GET **Path**: /users/{id}

Retrieve details of a specific user by ID.

Path Param	Type/Notes
id	integer

Responses

• **200** — Success

```
{
  "id": "integer",
  "username": "string",
  "email": "string",
  "password": "string"
}
```

• **400** — Bad request

Update a user

Method: PUT **Path**: /users/{id}

Update an existing user by ID.

Path Param	Type/Notes
id	integer

Request Body (application/json)

```
{
  "id": "integer",
  "username": "string",
  "email": "string",
  "password": "string"
}
```

Responses

• **200** — User updated successfully

```
{
  "id": "integer",
  "username": "string",
  "email": "string",
  "password": "string"
}
```

• **400** — Bad request

Delete a user

Method: DELETE **Path**: /users/{id}

Delete a specific user by ID.

Path Param	Type/Notes
id	integer

Responses

• **200** — User deleted successfully

• **400** — Bad request

Examples

js

```
fetch('https://fakestoreapi.com/users/1', { method: 'DELETE' })
  .then(r=>r.json())
  .then(console.log)
```

Auth

Login

Method: POST **Path**: /auth/login

Authenticate a user.

Request Body (application/json)

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

Responses

• **200** — Login successful

```
{
  "token": "string"
}
```

• **400** — Bad request

Examples

curl

```
curl -s -X POST https://fakestoreapi.com/auth/login -H 'Content-Type: application/json' -d
'{"username":"string","password":"string"}'
```