Product API Documentation-

This API allows you to manage a list of products in a simple database. You can retrieve, create, update, and delete products via HTTP requests.

Base URL:

http://127.0.0.1:5000

1. GET /products

Description:

Retrieves a list of all products.

Request:

- **Method**: `GET`
- **Endpoint**: `/products`
- **Headers**: None

Response:

- **Status Code**: `200 OK`
- **Body**: A JSON array of products, where each product is represented as:

```
{
   "id": 1,
   "title": "Product Title",
   "description": "Product Description",
  "price": 19.99
}
```

2. GET /products/<id>

Description:

Retrieves a specific product by its ID.

Request:

- **Method**: `GET`
- **Endpoint**: `/products/<id>`
- **Headers**: None

Parameters:

- `id`: The unique identifier of the product to be fetched.

Response:

- **Status Code**: `200 OK`
- **Body**: A JSON object representing the product:

```
{
  "id": 1,
  "title": "Product Title",
  "description": "Product Description",
  "price": 19.99
}
```

Status Code: `404 Not Found` if the product does not exist.

- **Body**:

```
{
  "error": "Product not found"
}
```

3. POST /products

Description:

Creates a new product.

Request:

- **Method**: `POST`
- **Endpoint**: `/products`
- **Headers**: `Content-Type: application/json`
- **Body**: A JSON object with product details:

```
{
  "title": "Product Title",
  "description": "Product Description",
  "price": 19.99
}
```

Response:

- **Status Code**: `201 Created`
- **Body**: A JSON object representing the newly created product:

```
{
  "id": 1,
  "title": "Product Title",
  "description": "Product Description",
  "price": 19.99
}
```

4. PUT /products/<id>

Description:

Updates the details of an existing product by its ID.

Request:

- **Method**: `PUT`
- **Endpoint**: `/products/<id>`
- **Headers**: `Content-Type: application/json`
- **Body**: A JSON object with product details (any fields can be updated):

```
{
  "title": "Updated Product Title",
  "description": "Updated Product Description",
  "price": 25.99
}
```

Parameters:

- id: The unique identifier of the product to be updated.

Response:

- **Status Code**: `200 OK`
- **Body**: A JSON object representing the updated product:

```
{
  "id": 1,
  "title": "Updated Product Title",
  "description": "Updated Product Description",

  "price": 25.99

}
```

Status Code: 404 Not Found if the product with the specified ID does not exist.

- **Body**:

```
{

  "error": "Product not found"

}
```

5. DELETE /products/<id>

Description:

Deletes a specific product by its ID.

Request:

- **Method**: `DELETE`
- **Endpoint**: `/products/<id>`
- **Headers**: None

Parameters:

- `id`: The unique identifier of the product to be deleted.

Response:

- **Status Code**: `200 OK`
- **Body**:

```
{
  "message": "Product deleted successfully"
}
```

Status Code: `404 Not Found` if the product with the specified ID does not exist.

- **Body**:

  ```
  {
    "error": "Product not found"
  }
  ```

Error Handling

404 Not Found: This status is returned when the requested resource (product) is not found in the database.

Example response:

```
{
  "error": "Resource not found"
}
```

500 Internal Server Error: This status is returned if there's an issue with the server.

Example response:

```
        {
          "error": "Internal server error"
        }
```

URL Commands:

1.GET all products:

```
curl -X GET http://127.0.0.1:5000/products
```

2.GET a specific product by ID:

```
curl -X GET http://127.0.0.1:5000/products/1
```

3.POST a new product:

```
curl -X POST http://127.0.0.1:5000/products -H "Content-Type: application/json" -d
'{"title": "New Product", "price": 10.0}'
```

4.PUT update a product:

```
curl -X PUT http://127.0.0.1:5000/products/1 -H "Content-Type: application/json" -d
'{"title": "Updated Product", "description": "Updated description", "price": 20.0}'
```

5.DELETE a product:

```
curl -X DELETE http://127.0.0.1:5000/products/1
```