ExpensesManager

Table of Contents

Chapter 1. Overview

ExpensesManager is a service to expenses management.

1.1. Version information

Version : 0.1.5

1.2. License information

License : Proprietary
Terms of service : null

1.3. URI scheme

Host : localhost:8080
BasePath : /
Schemes : HTTPS

1.4. Tags

- BillOfSale
- Budget
- Order
- Product

Chapter 2. Resources

2.1. BillOfSale

2.1.1. POST-endpoint to create BillOfSale.

```
POST /billofsales
```

Description

Method allow to create a new bill of sale.

Responses

HTTP Code	Description	Schema
200	successful operation	BillOfSale

Example HTTP request

Request path

```
/billofsales
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "productList" : [ {
    "id" : "string",
    "product" : {
      "id" : "string",
      "name" : "string",
      "price" : 0.0
    },
    "quanity" : 0
  } ],
  "boughtDate" : 0,
  "description" : "string"
}
```

2.1.2. GET-endpoint to find BillOfSale by bought date.

```
GET /billofsales
```

Description

Method allow to find a bill of sale by bought date.

Responses

HTTP Code	Description	Schema
200	successful operation	< BillOfSale > array

Example HTTP request

Request path

```
/billofsales
```

Example HTTP response

Response 200

```
[ {
  "id" : "string",
  "productList" : [ {
    "id" : "string",
    "product" : {
      "id" : "string",
      "name" : "string",
      "price" : 0.0
    },
    "quanity" : 0
  } ],
  "boughtDate" : 0,
  "description" : "string"
} ]
```

2.1.3. PUT-endpoint to update BillOfSale.

```
PUT /billofsales
```

Description

Method allow to update a bill of sale.

Responses

HTTP Code	Description	Schema
200	successful operation	BillOfSale

Example HTTP request

Request path

```
/billofsales
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "productList" : [ {
    "id" : "string",
    "product" : {
      "id" : "string",
      "name" : "string",
      "price" : 0.0
    },
    "quanity" : 0
  } ],
  "boughtDate" : 0,
  "description" : "string"
}
```

2.1.4. GET-endpoint to find BillOfSale by id.

```
GET /billofsales/{id}
```

Description

Method allow to find a bill of sale by id.

Responses

HTTP Code	Description	Schema
200	successful operation	BillOfSale

Example HTTP request

Request path

```
/billofsales/{id}
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "productList" : [ {
    "id" : "string",
    "product" : {
      "id" : "string",
      "name" : "string",
      "price" : 0.0
    },
    "quanity" : 0
  } ],
  "boughtDate" : 0,
  "description" : "string"
}
```

2.1.5. PUT-endpoint to update BillOfSale of id.

```
PUT /billofsales/{id}
```

Description

Method allow to update a bill of sale by id.

Responses

HTTP Code	Description	Schema
200	successful operation	BillOfSale

Example HTTP request

Request path

```
/billofsales/{id}
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "productList" : [ {
    "id" : "string",
    "product" : {
      "id" : "string",
      "name" : "string",
      "price" : 0.0
    },
    "quanity" : 0
  } ],
  "boughtDate" : 0,
  "description" : "string"
}
```

2.1.6. DELETE-endpoint to removeById BillOfSale.

```
DELETE /billofsales/{id}
```

Description

Method allow to removeById a bill of sale.

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

Example HTTP request

Request path

```
/billofsales/{id}
```

2.2. Budget

2.2.1. POST-endpoint to create Budget.

```
POST /budgets
```

Description

Method allow to create a new budget.

Responses

HTTP Code	Description	Schema
200	successful operation	Budget

Example HTTP request

Request path

```
/budgets
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "name" : "string",
  "budgetValue" : 0.0,
  "billsOfSaleList" : [ {
    "id" : "string",
    "productList" : [ {
      "id" : "string",
      "product" : {
        "id" : "string",
        "name" : "string",
        "price" : 0.0
      },
      "quanity" : 0
    } ],
    "boughtDate" : 0,
    "description" : "string"
  } ]
}
```

2.2.2. GET-endpoint to find Budget by name.

```
GET /budgets
```

Description

Method allow to find a budget by name.

Responses

HTTP Code	Description	Schema
200	successful operation	Budget

Example HTTP request

Request path

```
/budgets
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "name" : "string",
  "budgetValue" : 0.0,
  "billsOfSaleList" : [ {
    "id" : "string",
    "productList" : [ {
      "id" : "string",
      "product" : {
        "id" : "string",
        "name" : "string",
        "price" : 0.0
      },
      "quanity" : 0
    } ],
    "boughtDate" : 0,
    "description" : "string"
  } ]
}
```

2.2.3. PUT-endpoint to update Budget.

```
PUT /budgets
```

Description

Method allow to update a budget.

Responses

HTTP Code	Description	Schema
200	successful operation	Budget

Example HTTP request

Request path

```
/budgets
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "name" : "string",
  "budgetValue" : 0.0,
  "billsOfSaleList" : [ {
    "id" : "string",
    "productList" : [ {
      "id" : "string",
      "product" : {
        "id" : "string",
        "name" : "string",
        "price" : 0.0
      },
      "quanity" : 0
    } ],
    "boughtDate" : 0,
    "description" : "string"
  } ]
}
```

2.2.4. GET-endpoint to find Budget by id.

```
GET /budgets/{id}
```

Description

Method allow to find a budget by id.

Responses

HTTP Code	Description	Schema
200	successful operation	Budget

Example HTTP request

Request path

```
/budgets/{id}
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "name" : "string",
  "budgetValue" : 0.0,
  "billsOfSaleList" : [ {
    "id" : "string",
    "productList" : [ {
      "id" : "string",
      "product" : {
        "id" : "string",
        "name" : "string",
        "price" : 0.0
      },
      "quanity" : 0
    } ],
    "boughtDate" : 0,
    "description" : "string"
  } ]
}
```

2.2.5. PUT-endpoint to update Budget of id.

```
PUT /budgets/{id}
```

Description

Method allow to update a budget by id.

Responses

HTTP Code	Description	Schema
200	successful operation	Budget

Example HTTP request

Request path

```
/budgets/{id}
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "name" : "string",
  "budgetValue" : 0.0,
  "billsOfSaleList" : [ {
    "id" : "string",
    "productList" : [ {
      "id" : "string",
      "product" : {
        "id" : "string",
        "name" : "string",
        "price" : 0.0
      },
      "quanity" : 0
    } ],
    "boughtDate" : 0,
    "description" : "string"
  } ]
}
```

2.2.6. DELETE-endpoint to removeById Budget.

```
DELETE /budgets/{id}
```

Description

Method allow to removeById a budget.

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

Example HTTP request

Request path

```
/budgets/{id}
```

2.3. Order

2.3.1. POST-endpoint to create Order.

```
POST /orders
```

Description

Method allow to create a new order.

Responses

HTTP Code	Description	Schema
200	successful operation	ProductOrder

Example HTTP request

Request path

```
/orders
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "product" : {
    "id" : "string",
    "name" : "string",
    "price" : 0.0
  },
  "quanity" : 0
}
```

2.3.2. GET-endpoint to find Order by name and price.

```
GET /orders
```

Description

Method allow to find a order by name and price.

Responses

HTTP Code	Description	Schema
200	successful operation	ProductOrder

Example HTTP request

Request path

```
/orders
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "product" : {
    "id" : "string",
    "name" : "string",
    "price" : 0.0
  },
  "quanity" : 0
}
```

2.3.3. PUT-endpoint to update Order.

```
PUT /orders
```

Description

Method allow to update a order.

Responses

HTTP Code	Description	Schema
200	successful operation	ProductOrder

Example HTTP request

Request path

```
/orders
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "product" : {
    "id" : "string",
    "name" : "string",
    "price" : 0.0
  },
  "quanity" : 0
}
```

2.3.4. GET-endpoint to find Order by id.

```
GET /orders/{id}
```

Description

Method allow to find a order by id.

Responses

HTTP Code	Description	Schema
200	successful operation	ProductOrder

Example HTTP request

Request path

```
/orders/{id}
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "product" : {
    "id" : "string",
    "name" : "string",
    "price" : 0.0
  },
  "quanity" : 0
}
```

2.3.5. PUT-endpoint to update Order of id.

```
PUT /orders/{id}
```

Description

Method allow to update a order by id.

Responses

HTTP Code	Description	Schema
200	successful operation	ProductOrder

Example HTTP request

Request path

```
/orders/{id}
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "product" : {
    "id" : "string",
    "name" : "string",
    "price" : 0.0
  },
  "quanity" : 0
}
```

2.3.6. DELETE-endpoint to removeById Order.

```
DELETE /orders/{id}
```

Description

Method allow to removeById a order.

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

Example HTTP request

Request path

```
/orders/{id}
```

2.4. Product

2.4.1. POST-endpoint to create Product.

```
POST /products
```

Description

Method allow to create a new product.

Responses

HTTP Code	Description	Schema
200	successful operation	Product

Example HTTP request

Request path

```
/products
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "name" : "string",
  "price" : 0.0
}
```

2.4.2. GET-endpoint to find Product by name.

```
GET /products
```

Description

Method allow to find a product by name.

Responses

HTTP Code	Description	Schema
200	successful operation	< Product > array

Example HTTP request

Request path

```
/products
```

Example HTTP response

Response 200

```
[ {
  "id" : "string",
  "name" : "string",
  "price" : 0.0
} ]
```

2.4.3. PUT-endpoint to update Product.

```
PUT /products
```

Description

Method allow to update a product.

Responses

HTTP Code	Description	Schema
200	successful operation	Product

Example HTTP request

Request path

```
/products
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "name" : "string",
  "price" : 0.0
}
```

2.4.4. GET-endpoint to find Product by id.

```
GET /products/{id}
```

Description

Method allow to find a product by id.

Responses

HTTP Code	Description	Schema
200	successful operation	Product

Example HTTP request

Request path

```
/products/{id}
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "name" : "string",
  "price" : 0.0
}
```

2.4.5. PUT-endpoint to update Product of id.

```
PUT /products/{id}
```

Description

Method allow to update a product by id.

Responses

HTTP Code	Description	Schema
200	successful operation	Product

Example HTTP request

Request path

```
/products/{id}
```

Example HTTP response

Response 200

```
{
  "id" : "string",
  "name" : "string",
  "price" : 0.0
}
```

2.4.6. DELETE-endpoint to removeById Product.

```
DELETE /products/{id}
```

Description

Method allow to removeById a product.

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

Example HTTP request

Request path

```
/products/{id}
```

Chapter 3. Definitions

3.1. BillOfSale

Name	Description	Schema
boughtDate *optional*	**Example** : `0`	integer (int64)
description *optional*	**Example** : `"string"`	string
id *required*	**Example** : `"string"`	string
productList *optional*	**Example** : `[ "ProductOrder" ]`	< ProductOrder > array

3.2. Budget

Name	Description	Schema
billsOfSaleList *optional*	**Example** : `[ "BillOfSale" ]`	< BillOfSale > array
budgetValue *optional*	**Example** : `0.0`	number (double)
id *required*	**Example** : `"string"`	string
name *optional*	**Example** : `"string"`	string

3.3. Product

Name	Description	Schema
id *required*	**Example** : `"string"`	string
name *optional*	**Example** : `"string"`	string

Name	Description	Schema
price *optional*	**Example** : `0.0`	number (double)

3.4. ProductOrder

Name	Description	Schema
id *required*	**Example** : `"string"`	string
product *optional*	**Example** : Product	Product
quanity *optional*	**Example** : `0`	integer (int32)