Bookbuddy API

Bookbuddy API Information
Version: 1.0

http://apache.org/licenses/LICENSE-2.0.html

Access

Methods

[Jump to Models]

Table of Contents

UserController

- GET /users/all
- DELETE /users/{userId}
- GET /users/{userId}
- GET /users/email
- POST /users/new
- PUT /users/{userId}

BookCollectionController

Up

`PUT /collections/{userId}/{collectionName}/add_book/{bookId}`

Add a book to a collection (**addBook**)

Path parameters

userId (required)

Path Parameter — Unique ID corresponding to a user format: int64

collectionName (required)

Path Parameter — Name of the collection

bookId (required)

Path Parameter — Unique ID corresponding to a book format: int64

Return type

BookCollectionDTO

Example data

Content-Type: application/json

```
{
  "booksInCollection" : [ {
    "author" : "F. Scott Fitzgerald",
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
  }, {
    "author" : "F. Scott Fitzgerald",
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
  } ],
  "id" : 1,
  "collectionName" : "Favorite Books"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- */*

Responses
200
OK BookCollectionDTO
404
Not Found String

Up
`POST /collections/{userId}/new/{name}`

Add a new collection to the database (**addCollection**)
Path parameters
userId (required)
Path Parameter — Unique ID corresponding to a user format: int64
name (required)
Path Parameter — What the user wants to name the new collection

Return type
BookCollectionDTO
Example data
Content-Type: application/json


```
{
  "booksInCollection" : [ {
    "author" : "F. Scott Fitzgerald",
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
  }, {
    "author" : "F. Scott Fitzgerald",
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
  } ],
  "id" : 1,
  "collectionName" : "Favorite Books"
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- */*

Responses
201
Book Collection added successfully BookCollectionDTO

404
Not Found String

Up
`DELETE /collections/{collectionId}`

Delete collection by id (**deleteCollection**)
Path parameters
collectionId (required)
Path Parameter — Unique ID corresponding to a collection format: int64

Return type
String
Example data
Content-Type: application/json

```
""
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `*/*`

Responses
200
OK String
404
Not Found String

Up
`GET /collections/get/{userId}`

Get all collections from a user (**getAllCollectionsByUser**)
Path parameters
userId (required)
Path Parameter — Unique ID corresponding to a user format: int64

Return type
array[BookCollectionDTO]
Example data
Content-Type: application/json

```
[ {
  "booksInCollection" : [ {
    "author" : "F. Scott Fitzgerald",
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
```

```
  }, {
    "author" : "F. Scott Fitzgerald",
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
  } ],
  "id" : 1,
  "collectionName" : "Favorite Books"
}, {
  "booksInCollection" : [ {
    "author" : "F. Scott Fitzgerald",
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
  }, {
    "author" : "F. Scott Fitzgerald",
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
  } ],
  "id" : 1,
  "collectionName" : "Favorite Books"
} ]
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `*/*`

Responses
200
OK
404
Not Found String

Up

`GET /collections/{collectionId}`

Get collection by id (**getCollection**)
Path parameters
collectionId (required)
Path Parameter — Unique ID corresponding to a collection format: int64

Return type
BookCollectionDTO
Example data

Content-Type: application/json


```
{
  "booksInCollection" : [ {
    "author" : "F. Scott Fitzgerald",
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
  }, {
    "author" : "F. Scott Fitzgerald",
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
  } ],
  "id" : 1,
  "collectionName" : "Favorite Books"
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- */*

Responses
200
OK BookCollectionDTO
404
Not Found String

Up

`PUT /collections/{collectionId}/remove_book/{bookId}`

Remove a book from a collection (**removeBook**)
Path parameters
collectionId (required)
Path Parameter — Unique ID corresponding to a collection format: int64
bookId (required)
Path Parameter — Unique ID corresponding to a book format: int64

Return type
BookCollectionDTO
Example data
Content-Type: application/json


```
{
  "booksInCollection" : [ {
    "author" : "F. Scott Fitzgerald",
```

```
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
  }, {
    "author" : "F. Scott Fitzgerald",
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
  } ],
  "id" : 1,
  "collectionName" : "Favorite Books"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `*/*`

Responses
200
OK BookCollectionDTO
404
Not Found String

Up

`PUT /collections/{collectionId}/rename/{newName}`

Rename a collection (**renameCollection**)
Path parameters
collectionId (required)
Path Parameter — Unique ID corresponding to a collection format: int64
newName (required)
Path Parameter — What the user wants to rename to collection to

Return type
BookCollectionDTO
Example data
Content-Type: application/json

```
{
  "booksInCollection" : [ {
    "author" : "F. Scott Fitzgerald",
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
  }, {
```

```
    "author" : "F. Scott Fitzgerald",
    "price" : 12.99,
    "description" : "A novel about the American Dream",
    "id" : 1,
    "title" : "The Great Gatsby"
  } ],
  "id" : 1,
  "collectionName" : "Favorite Books"
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `*/*`

Responses
200
OK BookCollectionDTO
404
Not Found String

BookController

Up
`GET /books/{bookId}`
Get book by id (**findBook**)
Path parameters
bookId (required)
Path Parameter — Unique ID corresponding to a book format: int64

Return type
Book
Example data
Content-Type: application/json


```
{
  "author" : "F. Scott Fitzgerald",
  "price" : 12.99,
  "description" : "A novel about the American Dream",
  "id" : 1,
  "title" : "The Great Gatsby"
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- */*

Responses
200
OK Book
404
Not Found String

Up
`GET /books/all`

Get all books (**getAll**)
Return type
array[Book]
Example data
Content-Type: application/json

```
[ {
  "author" : "F. Scott Fitzgerald",
  "price" : 12.99,
  "description" : "A novel about the American Dream",
  "id" : 1,
  "title" : "The Great Gatsby"
}, {
  "author" : "F. Scott Fitzgerald",
  "price" : 12.99,
  "description" : "A novel about the American Dream",
  "id" : 1,
  "title" : "The Great Gatsby"
} ]
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- */*

Responses
200
OK
404
Not Found String

Up
`GET /books/popular`

Get most popular books (**getFirst20**)
Return type
array[Book]
Example data

Content-Type: application/json


```
[ {
  "author" : "F. Scott Fitzgerald",
  "price" : 12.99,
  "description" : "A novel about the American Dream",
  "id" : 1,
  "title" : "The Great Gatsby"
}, {
  "author" : "F. Scott Fitzgerald",
  "price" : 12.99,
  "description" : "A novel about the American Dream",
  "id" : 1,
  "title" : "The Great Gatsby"
} ]
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `*/*`

Responses

200

OK

404

Not Found String

Up

`GET /books/search/{searchTerm}`

Search for book by title (**searchBook**)

Path parameters

searchTerm (required)

Path Parameter — Title of book to search for

Return type

Book

Example data

Content-Type: application/json


```
{
  "author" : "F. Scott Fitzgerald",
  "price" : 12.99,
  "description" : "A novel about the American Dream",
  "id" : 1,
  "title" : "The Great Gatsby"
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- */*

Responses
200
OK Book
404
Not Found String

CartController

Up

`POST /cart/add-item/{cartId}/{bookId}`

Add an item to a cart (**addItemToCart**)
Path parameters
cartId (required)
Path Parameter — Unique ID corresponding to a Cart format: int64
bookId (required)
Path Parameter — Unique ID corresponding to a Book format: int64

Return type
CartDTO
Example data
Content-Type: application/json


```
{
  "totalPrice" : 100,
  "cartId" : 1,
  "userId" : 1,
  "items" : [ {
    "userCheckout" : {
      "cardCVV" : "cardCVV",
      "address" : "address",
      "cardExpiry" : "cardExpiry",
      "totalPrice" : 1.4658129805029452,
      "id" : 6,
      "cartItems" : [ null, null ],
      "user" : {
        "firstName" : "John",
        "lastName" : "Doe",
        "firebaseUID" : 123456789,
        "userDescription" : "I love reading!",
        "genres" : [ "Fantasy", "Science Fiction" ],
        "dateOfBirth" : "2000-01-23",
        "id" : 1,
```

```
        "email" : "test@email.com"
      },
      "cardNumber" : "cardNumber"
    },
    "book" : {
      "author" : "F. Scott Fitzgerald",
      "price" : 12.99,
      "description" : "A novel about the American Dream",
      "id" : 1,
      "title" : "The Great Gatsby"
    },
    "itemPrice" : 0.8008281904610115,
    "cartItemId" : 1
  }, {
    "userCheckout" : {
      "cardCVV" : "cardCVV",
      "address" : "address",
      "cardExpiry" : "cardExpiry",
      "totalPrice" : 1.4658129805029452,
      "id" : 6,
      "cartItems" : [ null, null ],
      "user" : {
        "firstName" : "John",
        "lastName" : "Doe",
        "firebaseUID" : 123456789,
        "userDescription" : "I love reading!",
        "genres" : [ "Fantasy", "Science Fiction" ],
        "dateOfBirth" : "2000-01-23",
        "id" : 1,
        "email" : "test@email.com"
      },
      "cardNumber" : "cardNumber"
    },
    "book" : {
      "author" : "F. Scott Fitzgerald",
      "price" : 12.99,
      "description" : "A novel about the American Dream",
      "id" : 1,
      "title" : "The Great Gatsby"
    },
    "itemPrice" : 0.8008281904610115,
    "cartItemId" : 1
  } ]
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- */*

Responses
200
OK CartDTO
404
Not Found String

Up
`GET /cart/get/{userId}`

Get a cart by ID (**getCart**)
Path parameters
userId (required)
Path Parameter — Unique ID corresponding to a User format: int64

Return type
CartDTO
Example data
Content-Type: application/json


```
{
  "totalPrice" : 100,
  "cartId" : 1,
  "userId" : 1,
  "items" : [ {
    "userCheckout" : {
      "cardCVV" : "cardCVV",
      "address" : "address",
      "cardExpiry" : "cardExpiry",
      "totalPrice" : 1.4658129805029452,
      "id" : 6,
      "cartItems" : [ null, null ],
      "user" : {
        "firstName" : "John",
        "lastName" : "Doe",
        "firebaseUID" : 123456789,
        "userDescription" : "I love reading!",
        "genres" : [ "Fantasy", "Science Fiction" ],
        "dateOfBirth" : "2000-01-23",
        "id" : 1,
        "email" : "test@email.com"
      },
      "cardNumber" : "cardNumber"
    },
    "book" : {
      "author" : "F. Scott Fitzgerald",
      "price" : 12.99,
      "description" : "A novel about the American Dream",
      "id" : 1,
      "title" : "The Great Gatsby"
    },
```

```
    "itemPrice" : 0.8008281904610115,
    "cartItemId" : 1
  }, {
    "userCheckout" : {
      "cardCVV" : "cardCVV",
      "address" : "address",
      "cardExpiry" : "cardExpiry",
      "totalPrice" : 1.4658129805029452,
      "id" : 6,
      "cartItems" : [ null, null ],
      "user" : {
        "firstName" : "John",
        "lastName" : "Doe",
        "firebaseUID" : 123456789,
        "userDescription" : "I love reading!",
        "genres" : [ "Fantasy", "Science Fiction" ],
        "dateOfBirth" : "2000-01-23",
        "id" : 1,
        "email" : "test@email.com"
      },
      "cardNumber" : "cardNumber"
    },
    "book" : {
      "author" : "F. Scott Fitzgerald",
      "price" : 12.99,
      "description" : "A novel about the American Dream",
      "id" : 1,
      "title" : "The Great Gatsby"
    },
    "itemPrice" : 0.8008281904610115,
    "cartItemId" : 1
  } ]
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `*/*`

Responses
200
OK CartDTO
404
Not Found String

Up
`PUT /cart/remove-item/{cartId}/{cartItemId}`
Remove an item from a cart (**removeItemFromCart**)

Path parameters
cartId (required)
Path Parameter — Unique ID corresponding to a Cart format: int64
cartItemId (required)
Path Parameter — Unique ID corresponding to a CartItem format: int64

Return type
CartDTO
Example data
Content-Type: application/json


```
{
  "totalPrice" : 100,
  "cartId" : 1,
  "userId" : 1,
  "items" : [ {
    "userCheckout" : {
      "cardCVV" : "cardCVV",
      "address" : "address",
      "cardExpiry" : "cardExpiry",
      "totalPrice" : 1.4658129805029452,
      "id" : 6,
      "cartItems" : [ null, null ],
      "user" : {
        "firstName" : "John",
        "lastName" : "Doe",
        "firebaseUID" : 123456789,
        "userDescription" : "I love reading!",
        "genres" : [ "Fantasy", "Science Fiction" ],
        "dateOfBirth" : "2000-01-23",
        "id" : 1,
        "email" : "test@email.com"
      },
      "cardNumber" : "cardNumber"
    },
    "book" : {
      "author" : "F. Scott Fitzgerald",
      "price" : 12.99,
      "description" : "A novel about the American Dream",
      "id" : 1,
      "title" : "The Great Gatsby"
    },
    "itemPrice" : 0.8008281904610115,
    "cartItemId" : 1
  }, {
    "userCheckout" : {
      "cardCVV" : "cardCVV",
      "address" : "address",
      "cardExpiry" : "cardExpiry",
      "totalPrice" : 1.4658129805029452,
```

```
      "id" : 6,
      "cartItems" : [ null, null ],
      "user" : {
        "firstName" : "John",
        "lastName" : "Doe",
        "firebaseUID" : 123456789,
        "userDescription" : "I love reading!",
        "genres" : [ "Fantasy", "Science Fiction" ],
        "dateOfBirth" : "2000-01-23",
        "id" : 1,
        "email" : "test@email.com"
      },
      "cardNumber" : "cardNumber"
    },
    "book" : {
      "author" : "F. Scott Fitzgerald",
      "price" : 12.99,
      "description" : "A novel about the American Dream",
      "id" : 1,
      "title" : "The Great Gatsby"
    },
    "itemPrice" : 0.8008281904610115,
    "cartItemId" : 1
  } ]
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `*/*`

Responses
200
OK CartDTO
404
Not Found String

GuestCheckoutController

Up
`POST /guestcheckout`

(**saveGuestCheckout**)
Consumes
This API call consumes the following media types via the Content-Type request header:

- application/json

Request body
body GuestCheckout **(required)**
Body Parameter —

Return type
GuestCheckout
Example data
Content-Type: application/json

```
{
  "cardCVV" : "cardCVV",
  "address" : "address",
  "cardExpiry" : "cardExpiry",
  "price" : "price",
  "name" : "name",
  "id" : 0,
  "bookName" : "bookName",
  "email" : "email",
  "cardNumber" : "cardNumber"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- */*

Responses
200
OK GuestCheckout
404
Not Found String

UserCheckoutController

Up
POST /user-checkout

(**createUserCheckout**)
Consumes
This API call consumes the following media types via the Content-Type request header:

- application/json

Request body

body UserCheckout **(required)**
Body Parameter —

Return type
UserCheckout
Example data
Content-Type: application/json

```
{
  "cardCVV" : "cardCVV",
  "address" : "address",
  "cardExpiry" : "cardExpiry",
  "totalPrice" : 1.4658129805029452,
  "id" : 6,
  "cartItems" : [ null, null ],
  "user" : {
    "firstName" : "John",
    "lastName" : "Doe",
    "firebaseUID" : 123456789,
    "userDescription" : "I love reading!",
    "genres" : [ "Fantasy", "Science Fiction" ],
    "dateOfBirth" : "2000-01-23",
    "id" : 1,
    "email" : "test@email.com"
  },
  "cardNumber" : "cardNumber"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- */*

Responses
200
OK UserCheckout
404
Not Found String

UserController

Up
`GET /users/all`

Admin route to list all users (**all**)
Return type
array[UserDTO]

Example data
Content-Type: application/json


```
[ {
  "firstName" : "John",
  "lastName" : "Doe",
  "userDescription" : "I love reading!",
  "genres" : [ {
    "name" : "name",
    "id" : 0
  }, {
    "name" : "name",
    "id" : 0
  } ],
  "cartId" : 1,
  "description" : "description",
  "dateOfBirth" : "2000-01-23",
  "id" : 1,
  "email" : "test@email.com"
}, {
  "firstName" : "John",
  "lastName" : "Doe",
  "userDescription" : "I love reading!",
  "genres" : [ {
    "name" : "name",
    "id" : 0
  }, {
    "name" : "name",
    "id" : 0
  } ],
  "cartId" : 1,
  "description" : "description",
  "dateOfBirth" : "2000-01-23",
  "id" : 1,
  "email" : "test@email.com"
} ]
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- */*

Responses
200
OK
404
Not Found String

Up

`DELETE /users/{userId}`

Delete user by id (**deleteUser**)

Path parameters

userId (required)

Path Parameter — Unique ID corresponding to a user format: int64

Return type

String

Example data

Content-Type: application/json

```
""
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `*/*`

Responses

200

OK String

404

Not Found String

Up

`GET /users/{userId}`

Get user details by id (**findUser**)

Path parameters

userId (required)

Path Parameter — Unique ID corresponding to a user format: int64

Return type

UserDTO

Example data

Content-Type: application/json

```
{
  "firstName" : "John",
  "lastName" : "Doe",
  "userDescription" : "I love reading!",
  "genres" : [ {
    "name" : "name",
    "id" : 0
  }, {
    "name" : "name",
    "id" : 0
```

```
  } ],
  "cartId" : 1,
  "description" : "description",
  "dateOfBirth" : "2000-01-23",
  "id" : 1,
  "email" : "test@email.com"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- */*

Responses
200
OK UserDTO
404
Not Found String

Up
`GET /users/email`

Get user details by email (**findUserByEmail**)
Query parameters
email (required)
Query Parameter —

Return type
UserDTO
Example data
Content-Type: application/json

```
{
  "firstName" : "John",
  "lastName" : "Doe",
  "userDescription" : "I love reading!",
  "genres" : [ {
    "name" : "name",
    "id" : 0
  }, {
    "name" : "name",
    "id" : 0
  } ],
  "cartId" : 1,
  "description" : "description",
  "dateOfBirth" : "2000-01-23",
  "id" : 1,
  "email" : "test@email.com"
```

```
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `*/*`

Responses
200
OK UserDTO
404
Not Found String

Up
`POST /users/new`

Add a new user to the database (**newUser**)
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
body CreateUserDTO (required)
Body Parameter —

Return type
UserDTO
Example data
Content-Type: application/json

```
{
  "firstName" : "John",
  "lastName" : "Doe",
  "userDescription" : "I love reading!",
  "genres" : [ {
    "name" : "name",
    "id" : 0
  }, {
    "name" : "name",
    "id" : 0
  } ],
  "cartId" : 1,
  "description" : "description",
  "dateOfBirth" : "2000-01-23",
  "id" : 1,
  "email" : "test@email.com"
```

```
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `*/*`

Responses
201
User added successfully UserDTO
404
Not Found String

Up
`PUT /users/{userId}`

Update user details by id (**updateUser**)
Path parameters
userId (required)
Path Parameter — Unique ID corresponding to a user format: int64

Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
body UserDTO (required)
Body Parameter —

Return type
UserDTO
Example data
Content-Type: application/json

```
{
  "firstName" : "John",
  "lastName" : "Doe",
  "userDescription" : "I love reading!",
  "genres" : [ {
    "name" : "name",
    "id" : 0
  }, {
    "name" : "name",
    "id" : 0
  } ],
  "cartId" : 1,
  "description" : "description",
```

```
  "dateOfBirth" : "2000-01-23",
  "id" : 1,
  "email" : "test@email.com"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `*/*`

Responses
200
OK UserDTO
404
Not Found String

Models

[Jump to Methods]

Table of Contents

Book Up

A book entity
id (optional)
Long Unique ID generated by database corresponding to a book format: int64
example: 1
title (optional)
String Title of the book
example: The Great Gatsby
author (optional)
String Author of the book

example: F. Scott Fitzgerald
price (optional)
BigDecimal Price of the book
example: 12.99
description (optional)
String Description of the book
example: A novel about the American Dream

`BookCollectionDTO` Up

A DTO for a book collection
id (optional)
Long Unique ID generated by database corresponding to a book collection format: int64
example: 1
collectionName (optional)
String Name of the collection
example: Favorite Books
booksInCollection (optional)
array[Book] List of books in the collection

`CartDTO` Up

Data Transfer Object for Cart
cartId (optional)
Long Unique ID corresponding to a Cart format: int64
example: 1
userId (optional)
Long Unique ID corresponding to a User format: int64
example: 1
items (optional)
array[CartItem] List of items in the cart
totalPrice (optional)
BigDecimal Total price of all items in the cart
example: 100

`CartItem` Up

An item present in the cart
cartItemId (optional)
Long Unique ID generated by database corresponding to a user format: int64
example: 1
book (optional)
Book
itemPrice (optional)
BigDecimal price of the item
userCheckout (optional)
UserCheckout

`CreateUserDTO` [Up]

A DTO for creating a user
firstName (optional)
String First name of the user
example: John
lastName (optional)
String Last name of the user
example: Doe
email (optional)
String Email of the user
example: test@email.com
dateOfBirth (optional)
date Date of birth of the user format: date
description (optional)
String Description of the user
example: I love reading!
genres (optional)
String
genresList (optional)
array[String]

`Genre` [Up]

List of favorite genres of the user
id (optional)
Long format: int64
name (optional)
String

`GenreDTO` [Up]

List of genres that the user is interested in
id (optional)
Long format: int64
name (optional)
String

`GuestCheckout` [Up]

id (optional)
Long format: int64
name (optional)
String
email (optional)
String
address (optional)

String
cardNumber (optional)
String
cardExpiry (optional)
String
cardCVV (optional)
String
price (optional)
String
bookName (optional)
String

`User` Up

A user of the application
id (optional)
Long Unique ID generated by database corresponding to a user format: int64
example: 1
firebaseUID (optional)
Long Unique ID generated by Firebase corresponding to a user format: int64
example: 123456789
firstName (optional)
String First name of the user
example: John
lastName (optional)
String Last name of the user
example: Doe
email (optional)
String Email of the user
example: test@email.com
dateOfBirth (optional)
date Date of birth of the user format: date
userDescription (optional)
String Description of the user
example: I love reading!
genres (optional)
array[Genre] List of favorite genres of the user
example: ["Fantasy","Science Fiction"]

`UserCheckout` Up

id (optional)
Long format: int64
user (optional)
User
cartItems (optional)
array[CartItem]
totalPrice (optional)

BigDecimal
address (optional)
String
cardNumber (optional)
String
cardExpiry (optional)
String
cardCVV (optional)
String

`UserDTO` Up

A DTO for a user
id (optional)
Long Unique ID generated by database corresponding to a user format: int64
example: 1
firstName (optional)
String First name of the user
example: John
lastName (optional)
String Last name of the user
example: Doe
email (optional)
String Email of the user
example: test@email.com
dateOfBirth (optional)
date Date of birth of the user format: date
cartId (optional)
Long Unique ID corresponding to a user's cart format: int64
example: 1
userDescription (optional)
String Description of the user
example: I love reading!
genres (optional)
array[GenreDTO] List of genres that the user is interested in
description (optional)
String