Endpoint:http://localhost:2222

|  |  |  |
| --- | --- | --- |
| **Description** | **METHOD+path** | **Request/Response** |
| **U S E R A U T H E N T I F I C AT I O N** | | |
| Register user | POST  /users/register | **Request**:{  id:number,  name:string,  surname:string,  email:string,  password:string,  city:string,  street:string,  role:string,  SignedIn:0  }  **Response**:updatedUsers[{  id:number,  name:string,  surname:string,  email:string,  password:string,  city:string,  street:string,  role:string,  SignedIn:0}  ] |
| Login user | PUT  /users/login | **Request**:{  email:string,  password:string  }  **Response**:{  id:number,  name:string,  surname:string,  email:string,  password:string,  city:string,  street:string,  role:string,  SignedIn:1,  connected\_devices:string,  access\_token:string  } |
| Renewing access token with valid  refresh token | GET  /users/gettokenwithrefresh/:id | **Request**: id (user id from decoded refresh token)  **Response**:{err:false,  access\_token:string,  refreshedUser  [  {  id:number,  name:string,  surname:string,  email:string,  password:string,  city:string,  street:string,  role:string,  SignedIn:1,  connected\_devices:string,  access\_token:string  }  ]  } |
| Log out user | PUT  /users/logout | **Request**:  {body:localStorage.token}  **Response**:  signedOutUser  {  id:number,  name:string,  surname:string,  email:string,  password:string,  city:string,  street:string,  role:string,  SignedIn:0,  connected\_devices:null,  access\_token:null  } |
| ID validation | GET  /users/validateid/:id | **Request**: id (ID input value)  **Response a**:  {  err:false  msg:"This ID number is already in  use!"  }  **Response b**:{err:false,  msg:"ID OK!"  } |
| Email validation | GET  /users/validateemale/:email | **Request**: email (email input value)  **Response** **a**:  {  err:false  msg:"This email address is already in  use!"  }  **Response b**:  {  err:false,  msg:"Email OK!"  } |
| **A D M I N O N L Y A C T I O N S** | | |
| Post new item (protected: vt,staffOnly) | POST  /orders/post | **Request**:  {  author:string,  name:string,  price:number,  image:string,  description:string,  categoryID:string  }  **Response**:  {  err:false,  {  author:string,  name:string,  price:number,  image:string,  description:string,  categoryID:string  }  } |
| Edit item (protected: vt, staffOnly) | PUT  /orders/admin/edit/:id | **Request**: id, body: {  author:string,  name:string,  price:number,  image:string,  description:string  }  **Response**:  {  err:false,  [  {  author:string,  name:string,  price:number,  image:string,  description:string  }  ]  } |
| **U S E R S H O P P I N G P R O C E S S** | | |
| Display available items | GET  /orders | **Request**:no  **Response**:  {  err:false,  selectedProducts  [  {  id:number,  author:string,  name:string,  price:number,  image:string,  description:string  category\_id:number  }  ] |
| Checking if there is an open basket  for a specific user | GET  /orders/searchbaskets/:id | **Request**: id (user id)  **Response a**:  {  err:false,  open\_basket{  id:number,  date:string,  user\_id:number,  status:"open",  total\_price:number  },  chosen\_items  [  {  id:number  author:string,  name:string,  price:number,  image:string,  description:string  category\_id:number,  quantity:number,  total\_price:number  }  ]  }  **Response b**:  {  err:false,  closed\_selectedBasket  {  id:number,  date:string,  user\_id:number,  status:"closed",  total\_price:number  }  } |
| Display the number of products in the shop | GET  /orders/productscount | **Request**:no  **Response**:  [{'COUNT (id):number}] |
| Display the number of orders in the shop | GET  /orders/orderscount | **Request**:no  **Response**:  [{'COUNT (id):number}] |
| Display the availability of shipping dates | GET  /orders/freeshippingdates | **Request**:no  **Response**:  {err.false,  freeshipdates  [  {'COUNT (id):number,  shipping\_date:string}  ] |
| Display items by category(protected by: vt) | GET  /orders/bycategory/:categoryID | **Request**: categoryID  **Response**:  err:false,  products\_by\_category  [  {  id:number,  author:string,  name:string,  price:number,  image:string,  description:string  category\_id:number,  }  ] |
| Search items by name (protected by:vt) | GET  /orders/byname/:name | **Request**: name (product name)  **Response**:[  {  id:number,  author:string,  name:string,  price:number,  image:string,  description:string  category\_id:number,  description:number  }  ] |
| Open new cart (basket)(protected by:vt) | POST  /orders/basket | **Request**: body:{id} (user id)  **Response**:  {  id:number,  date:string,  user\_id:number,  status:"open",  total\_price:number  } |
| User puts new item into the cart  (protected by:vt) | POST  /orders/iteminbasket | **Request**: body{productID,newBasketID,  quantity}  **Response**:  {  err: false,  item\_in\_basket{  id:number  author:string,  name:string,  price:number,  image:string,  description:string  category\_id:number,  quantity:number,  total\_price:number  },  total\_price[  {  'SUM (total\_price)':number  }  ]  } |
| Increment by 1 of item's quantity in cart by user(protected by:vt) | PUT  /orders/addquantity | **Request**:body{id,basketID}  (id=>product id)  **Response**:[  { 'SUM(quantity)': number }  ] |
| Decrement by 1 of item's quantity in cart by user(protected by:vt) | PUT  /orders/removequantity | **Request**:body{id,basketID}  (id=>product id)  **Response**:[  { 'SUM(quantity)': number }  ] |
| Deleting item from cart by user  (protected by:vt) | DELETE  /orders/deleteitemfrombasket/  :itemid/:basketid | **Request**: itemid,basketid  **Response**:  {  err:false,  deleted\_item[  {  id:number  author:string,  name:string,  image:string,  category\_id:number,  quantity:number,  total\_price:number  }  ] ,  returned\_item{OKPacket},  total\_price[  {  'SUM (total\_price): number  }  ]  } |
| Emptying cart by user  (protected by:vt) | DELETE  /orders/emptybasket/:basketid | **Request**:basketid  **Response**:{  err:false,  empty-basket{OKPacket}  } |
| User commits order  (protected by:vt) | PUT  /orders/savebasket | **Request**: id,body{  id, totalPrice, shippingDate, card, userID, city, street }  (id=>basket id)  **Response**:  {  err:false,  ordered\_items  [  {  id:number,  name:string,  author:string,  price:number,  quantity:number,  total\_price:number  }  ],  [  {  id:number,  date:string,  user\_id:number,  status:"closed",  total\_price:number  {  ],  [  {  id:number,  user\_id:number,  basket\_id: number,  total\_price:number,  city:string,  street:string,  order\_date:string,  shipping\_date:string,  credit\_card\_number:string  {  ]  } |
| Generate and print receipt | GET  /orders/printreceipt/:userID/:basketID | **Request**:userID,basketID  **Response**:Blob  {  size:number,  type:"application/pdf"  } |