# API OQADU

Customer advisor API for retailers.

## Paths :

## /Products

The products endpoint returns information about products. The response includes the display name and other details about products.

Method : POST, GET

Parameters :

| Name | Located in | Description | Required | Type |
|---|---|---|---|---|
| **_id** | Query | A product _id | No | ObjectId |

Responses :

| Code | Description | Schema |
|---|---|---|
| **200** | Returns an array of products | {<br>data : [**Product**],<br>status : 200<br>} |
| **Default** | Unexcpected Error | {<br>data : string (error message)<br>status : 400<br>} |

# /Products/Recommendations

The recommendations endpoint returns products wich correspond to customer needs according to the tags given as a request parameter. If the given paramter is a product id, then, the recommendation endpoint returns similar products.

Method : POST

Parameters :

| Name | Located in | Description | Required | Type |
|---|---|---|---|---|
| **tags** | Query | An array of tags | No | [string] |
| **Product id** | Query | A product _id | No | ObjectId |

Responses :

| Code | Description | Schema |
|---|---|---|
| **200** | Returns an array of products | {<br>data : [**Product**],<br>status : 200<br>} |
| **416** | No product corresponding to the given tags | {<br>data : string (error message)<br>status : 416<br>} |
| **default** | Unexcpected Error | {<br>data : string (error message)<br>status : 400<br>} |

# /Products/Search

The search endpoint returns products matching the name given as a request parameter.

Method : GET

Parameters :

| Name | Located in | Description | Required | Type |
|---|---|---|---|---|
| **name** | Query | part of a product name | Yes | string |

Responses :

| Code | Description | Schema |
|---|---|---|
| **200** | Returns an array of products | {<br>data : [**Product**],<br>status : 200<br>} |
| **default** | Error while searching products | {<br>data : string (error message)<br>status : 400<br>} |

# /Products/Promo

The promotion endpoint returns products wich are currently promoted.

Method : GET

Responses :

| Code | Description | Schema |
|---|---|---|
| **200** | Returns an array of products | {<br>data : [**Product**],<br>status : 200<br>} |
| **404** | No promoted product found | {<br>data : string (error message)<br>status : 404<br>} |
| **default** | Error while getting promoted products | {<br>data : string (error message)<br>status : 400<br>} |

# /Products/Barcode

The barcode endpoint returns the _id field of the product matching the barcode given as a request parameter.

Method : GET

Parameters :

| Name | Located in | Description | Required | Type |
|---|---|---|---|---|
| **barcode** | Query | Any product barcode | Yes | Number |

Responses :

| Code | Description | Schema |
|---|---|---|
| **200** | Returns the _id field of the corresponding product | {<br>data : ObjectId,<br>status : 200<br>} |
| **404** | No product found | {<br>data : string (error message)<br>status : 404<br>} |
| **default** | Error while getting products | {<br>data : string (error message)<br>status : 400<br>} |

# /Questions

The questions endpoint returns information about questions. The response includes the display text, the answers and the tags attached to each returned question.

Method : POST, GET

Parameters :

| Name | Located in | Description | Required | Type |
|---|---|---|---|---|
| **_id** | Query | A question id | No | ObjectId |

Responses :

| Code | Description | Schema |
|---|---|---|
| **200** | Returns an array of questions | {<br>data : [**Question**],<br>status : 200<br>} |
| **Default** | Unexcpected Error | {<br>data : string (error message)<br>status : 400<br>} |

# /Questions/Next

The next question endpoint returns the next question to ask to the customer depending on the already answered questions and the tags already attached to the customer profile. The response includes the display text, the answers and the tags attached to the returned question. In some cases, answers are removed from the returned question if adding their tags to the customer tags would not produce any result according to the recommendation endpoint.

Method : POST

Parameters :

| Name | Located in | Description | Required | Type |
|---|---|---|---|---|
| **tags** | Query | An array of tags | Yes | [string] |
| **Answered questions** | Query | An array of questions id | Yes | [ObjectId] |

Responses :

| Code | Description | Schema |
|---|---|---|
| **200** | A new question have been found | {<br>data : **Question**,<br>status : 200<br>} |
| **204** | A new question have been found but there is not any answer left for this one. | {<br>data : ObjectId,<br>status : 204<br>} |
| **Default** | Unexcpected Error | {<br>data : string (error message)<br>status : 400<br>} |

# /User

The user endpoint returns information about users. The response includes the display name and other details about users.

Method : POST, GET

Parameters :

| Name | Located in | Description | Required | Type |
|---|---|---|---|---|
| **_id** | Query | A user id | No | ObjectId |

Responses :

| Code | Description | Schema |
|---|---|---|
| **200** | Returns an array of users | {<br>data : [**User**],<br>status : 200<br>} |
| **Default** | Unexcpected Error | {<br>data : string (error message)<br>status : 400<br>} |

# /User/Login

The login  endpoint returns information about the user corresponding to the pair (username, password) given as parameters of the request. The response includes the display name and other details about the requested user.

Method : POST

Parameters :

| Name | Located in | Description | Required | Type |
|---|---|---|---|---|
| **username** | Query | User's name ? | Yes | string |
| **password** | Query | Does it need to be described ? | Yes | string |

Responses :

| Code | Description | Schema |
|---|---|---|
| **200** | Returns the matching user | {<br>data : **User**,<br>status : 200<br>} |
| **500** | Authentication Error | {<br>data : string (error message)<br>status : 500<br>} |
| **Default** | Authentication Failure | {<br>data : string (error message)<br>status : 401<br>} |

# Models :

```
Question {
  text: {type:'string', required:true},
  answers: {
    type: [{
      text: {type:'string', required:true},
      tags: {type:['string'], required:false}
    }],
    required: true
  },
  tags: {type:['string'], required:false}
}
```

```
Product {
  barcode: {type:'Number', required:true},
  name: {type:'string', required:true},
  price: {type:'Number', required:true},
  promo: {type:'Number', required:false},
  tags: {type:['string'], required:false},
  pictures: {
   type:[{
    label: {type:'string', required:true},
    path: {type:'string', required:true}
   }],
   required:false
  },
  features: {
   type:[{
    label: {type:'string', required:true},
    value: {type:'string', required:true}
   }],
   required:false
  },
  reviews: {
   type: [{
    title: {type:'string', required:true},
    reviewerName: {type:'string', required:true},
    score: {type:'Number', required:true},
    comment:{type:'string', required: false}
   }],
   required:false
  },
  faq: {
   type: [{
    title: {type:'string', required:true},
```

```
    content: {type:'string', required:true}
   }],
   required:false
  }
 }
```

# User

User model represents both customer and seller. The purpose of the field 'rank' is to differentiate them.

In the case of a customer user, the field 'shelf' is not used and the field 'clientId' is used to identify the customer account.

In the case of a seller user, the field 'clientId' is not used and the field 'shelf' is used to indicates the shelf where the seller is affected.

```
{
  username: {type:'string', required:true},
  password: {type: 'Number', required:true},
  name: {type:'string', required:true},
  clientId: {type: 'Number', required: true, default: 0},
  rank: {type: 'Number', required: true, default: 0},
  shelf: {type: 'string', required: false},
  avatar: {
   type: {
    thumb: {type:'string', required: false},
    normal: {type: 'string', required: false}
   },
   required: false
  }
 }
```