ONBOARD PROJECT DOCUMENTATION

Mentor: Hiep Nguyen

Trainee: Troy

2019

TABLE OF CONTENTS

I. API Documentation

1. Default error response:

- For each error response, the default form would be like this:

Name	Type	Description
error_code	Int	The encoded name for the error
error_message	String	The detail of the error

- An error response example:

```
{
   "error_message": "Item with this id doesn't exist.",
   "error_code": 404001
}
```

- By default, **Validation Error** (400) will catch errors like: no body response, body response with invalid parameters name,...
- While processing request, other than the errors that we provided in each endpoints, there will be an error handler for catching **Internal Server Error** (500).

Error Code	Meanings
400001	Validation Error
400002	Duplicate Entity
400003	False Authentication
403001	Forbidden
404001	Not Found
500001	Internal Server Error

2. **Authentication**

a. **POST**

/auth

Allow client to authenticate by using email and password.

REQUEST

- **Parameters**: No content
- **Header**:

Key	**Value**
Content-Type	application/json

- **Body**:

Name	**Type**	**Description**	**Example**
email	String	Email of the user	admin@admin.com
password	String	Password of the user	123456

SUCCESSFUL RESPONSE

- **HTTP code**: **200 OK**
- **Headers**:

Key	**Value**
Content-Type	application/json

- **Response:**

Name	**Type**	**Description**	**Example**
access_token	String	JWT token	".....”
user.id	Int	Identifier of the logged in user	1
user.email	String	Email of the logged in user	admin@gmail.com

- **Response example**:


```
{
   "access_token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE1NzEyODAxMTEsIm5iZiI6MTU3MTI4MDExMS
wianRpIjoiMzU3YWJjNTYtNjRjOC00MzAzLWE3YmMtMDIxNmNmYWVhYmI4IiwiZXhwIjoxNTcxMjgxMDExLC
JpZGVudGl0eSI6MSwiZnJlc2giOmZhbHNlLCJ0eXBlIjoiYWNjZXNzIn0.M_r1meDSt_jpSW3SPjI5kVwyCQ
mgPKoBoPM-59NYJVU",
   "user": {
          "id": "1”,
          "email": "admin@gmail.com”,
   }
}
```


ERROR RESPONSES

- Validation Error (400): If email/ password of the body doesn’t pass the validation test.
- False Authentication (400): If client sent an unregistered emailed, wrong password

3. **Category**

a. **GET**

/categories

Allow client to get all categories with pagination.

REQUEST

- **Parameters**:

Name	Type	Description	Example	Default
page	Int	The page of categories that client wants to get	1	1
per_page	Int	Number of categories each page has	5	5

- **Header**:

Key	Value
Content-Type	application/json

- **Body**: No content

SUCCESSFUL RESPONSE

- **HTTP code**: **200 OK**
- **Headers**:

Key	**Value**
Content-Type	application/json

- **Response:**

Name	**Type**	**Description**	**Example**
page	int	Current page	1
data	Array	Array of categories (for specific details of a category entity, please seek for GET /categories/:id)	[]
per_page	int	How many entities in this page	5
total_items	int	Total entities in the database	100

- **Response example**:


```
{
   "page": 1,
   "data": [
       {
           "created": "2019-10-17T09:42:04",
           "creator_id": 1,
           "description": "Minecraft stuffs...",
           "id": 1,
           "title": "Minecraft",
           "updated": "2019-10-17T09:42:04"
       },
       {
           "created": "2019-10-17T09:42:24",
           "creator_id": 1,
           "description": "Seafood stuffs...",
           "id": 2,
           "title": "Seafood",
           "updated": "2019-10-17T09:42:24"
       }
   ],
   "per_page": 5,
   "total_items": 2
}
```


ERROR RESPONSES

- Validation Error (400): If client passed in invalid per_page, page

b. GET

/categories/:category_id

Allow client to get category by id.

REQUEST

- **Parameters**:

Name	Type	Description	Example
category_id	Int	Identifier of the category that the client wants to get	1

- **Header**:

Key	Value
Content-Type	application/json

- **Body**: No content

SUCCESSFUL RESPONSE

- **HTTP code**: **200 OK**
- **Headers**:

Key	**Value**
Content-Type	application/json

- **Response:** All of these fields will be wrapped by an object called "**data**"

Name	**Type**	**Description**
created	String	The day this entity was created
creator_id	Int	Identifier of the user that created this entity
description	String	The description of this entity
id	Int	Identifier of this entity
title	String	Title of this entity
updated	String	The latest day this entity was updated

- **Response example**:


```
{
   "data": {
       "created": "2019-10-17T09:42:04",
       "creator_id": 1,
       "description": "Minecraft stuffs...",
       "id": 1,
       "title": "Minecraft",
       "updated": "2019-10-17T09:42:04"
   }
}
```


ERROR RESPONSES

- Not Found (404): If client tries to get a category that doesn’t exist.

c. **POST**

/categories

Allow client to create a category.

REQUEST

- **Parameters**: No content
- **Header**:

Key	**Value**
Content-Type	application/json
Authorization	Bearer jwt_token

- **Body**:

Name	**Type**	**Description**	**Example**
title	String	Title of the category	Minecraft
description	String	Description of the category	Minecraft stuffs

SUCCESSFUL RESPONSE

- **HTTP code**: **200 OK**
- **Headers**:

Key	**Value**
Content-Type	application/json

- **Response:** All of these fields will be wrapped by an object called "**data**"

Name	**Type**	**Description**
created	String	The day this entity was created
creator_id	Int	Identifier of the user that created this entity
description	String	The description of this entity
id	Int	Identifier of this entity
title	String	Title of this entity
updated	String	The latest day this entity was updated

- **Response example**:

```
{
    "data": {
        "created": "2019-10-18T16:20:47",
        "creator_id": 1,
        "description": "Mfinecraft stuffs...",
        "id": 2,
        "title": "Minecraferferfft",
        "updated": "2019-10-18T16:20:47"
    }
}
```

ERROR RESPONSES

- Validation Error (400): If title/description doesn't pass the validation test.

- Duplicate Entity (400): If client tries to create a category with the title that has already used by another category.
- Unauthorized (401): If client hasn’t logged in yet.

d. **PUT**

/categories/:category_id

Allow client to update a category.

REQUEST

- **Parameters**:

Name	Type	Description	Example
category_id	Int	Identifier of the category that the client wants to update	2

- **Header**:

Key	Value
Content-Type	application/json
Authorization	Bearer jwt_token

- **Body**:

Name	Type	Description	Example
title	String	Title of the category	Minecraft
description	String	Description of the category	Minecraft stuffs

SUCCESSFUL RESPONSE

- **HTTP code**: **200 OK**
- **Headers**:

Key	**Value**
Content-Type	application/json

- **Response:** All of these fields will be wrapped by an object called "**data**"

Name	**Type**	**Description**
created	String	The day this entity was created
creator_id	Int	Identifier of the user that created this entity
description	String	The description of this entity
id	Int	Identifier of this entity
title	String	Title of this entity
updated	String	The latest day this entity was updated

- **Response example**:

```
{
    "data": {
        "created": "2019-10-18T16:20:47",
        "creator_id": 1,
        "description": "Mfinecraft stuffs...",
        "id": 2,
        "title": "Minecrafdwedwedvdvdfvdtyee",
        "updated": "2019-10-18T16:23:09"
    }
}
```

ERROR RESPONSES

- Validation Error (400): If title/description doesn't pass the validation test.
- Duplicated Entity (400): If a category with the title has already existed.

- Unauthorized (401): If client hasn’t logged in yet.
- Forbidden (403): If client tries to update another user’s category.
- Not Found (404): If client tries to update a category that doesn’t exist.

e. **DELETE**

/categories/:category_id

Allow client to delete a category, all the items that belong to this category will be deleted also.

REQUEST

- **Parameters**:

Name	Type	Description	Example
category_id	Int	Identifier of the category that the client wants to delete	1

- **Header**:

Key	Value
Content-Type	application/json
Authorization	Bearer jwt_token

- **Body**: No content

SUCCESSFUL RESPONSE

- **HTTP code**: **204 No Content**
- **Headers**:

Key	Value
Content-Type	application/json

ERROR RESPONSES

- Unauthorized (401): If client hasn't logged in yet.
- Forbidden (403): If client tries to delete another user's category.
- Not Found (404): If client tries to delete a category that doesn't exist.

4. **Item**
a. **GET**

/items

Allow client to get all items with pagination, can filter by category.

REQUEST

- **Parameters**:

Name	Type	Description	Example	Default
page	Int	The page of items that client wants to get	1	1
size	Int	Number of items each page has	5	5
category_id	Int	Filter item by category	1	None

- **Header**:

Key	Value
Content-Type	application/json

- **Body**: No content

SUCCESSFUL RESPONSE

- **HTTP code**: **200 OK**
- **Headers**:

Key	**Value**
Content-Type	application/json

- **Response:**

Name	**Type**	**Description**	**Example**
page	int	Current page	1
data	Array	Array of categories (for the specific information of each entity, please seek for the GET /items/:id)	[]
per_page	int	How many entities in this page	5
total_items	int	Total entities in the database	100

- **Response example**:


```
{
   "page": 1,
   "data": [
       {
           "category": {
                 "id": 1,
                 "title": "Minecraft",
           },
           "created": "2019-10-17T09:49:51",
           "creator_id": 1,
           "description": "Very shiny...",
           "id": 1,
           "title": "Minecraft Light Block",
           "updated": "2019-10-17T09:49:51"
       },
       {
           "category": {
                 "id": 1,
                 "title": "Minecraft",
           },
           "created": "2019-10-17T09:49:54",
           "creator_id": 1,
           "description": "Very shiny...",
           "id": 2,
           "title": "Minecraft Lamp Block",
           "updated": "2019-10-17T09:49:54"
       }
   ],
   "per_page": 5,
   "total_items": 2
}
```


ERROR RESPONSES

- Validation Error (400): When client passed in an invalid argument (page, per_page)

b. **GET**

/items/:item_id

Allow client to get an item by id.

REQUEST

- **Parameters**:

Name	Type	Description	Example
item_id	Int	Identifier of the item that the client wants to get	2

- **Header**:

Key	Value
Content-Type	application/json

- **Body**: No content

SUCCESSFUL RESPONSE

- **HTTP code**: **200 OK**
- **Headers**:

Key	**Value**
Content-Type	application/json

- **Response:** All of these fields will be wrapped by an object called "**data**"

Name	**Type**	**Description**
created	String	The day this entity was created
creator_id	Int	Identifier of the user that created this entity
description	String	The description of this entity
id	Int	Identifier of this entity
title	String	Title of this entity
updated	String	The latest day this entity was updated
category.id	Int	Identifier of the category to which this item belongs
category.title	String	The title of the category to which this item belongs

- **Response example**:

```
{
   "data": {
       "category": {
           "id": 1,
           "title": "Minecraft"
       },
       "created": "2019-10-17T09:49:54",
       "creator_id": 1,
       "description": "Very shiny...",
       "id": 2,
       "title": "Minecraft Lamp Block",
       "updated": "2019-10-17T09:49:54"
   }
}
```


ERROR RESPONSES

- Not Found (404): If client tries to get an item that doesn't exist.

c. **POST**

/items

Allow client to create an item.

REQUEST

- **Parameters**: No content
- **Header**:

Key	**Value**
Content-Type	application/json
Authorization	Bearer jwt_token

- **Body**:

Name	**Type**	**Description**	**Example**
title	String	Title of the item	Minecraft Sword
description	String	Description of the item	Yeah awesome...
category_id	Int	Identifier of the category that this item will belong to	1

SUCCESSFUL RESPONSE

- **HTTP code**: **200 OK**
- **Headers**:

Key	**Value**
Content-Type	application/json

- **Response:** All of these fields will be wrapped by an object called "**data**"

Name	**Type**	**Description**
created	String	The day this entity was created
creator_id	Int	Identifier of the user that created this entity
description	String	The description of this entity
id	Int	Identifier of this entity
title	String	Title of this entity
updated	String	The latest day this entity was updated
category.id	Int	Identifier of the category to which this item belongs
category.title	String	The title of the category to which this item belongs

- **Response example**:


```
{
   "data": {
       "category": {
           "id": 2,
           "title": "Minecrafdwedwedvdvdfvdtyee"
       },
       "created": "2019-10-18T17:10:18",
       "creator_id": 1,
       "description": "Very shiny...",
```

```
        "id": 7,
        "title": "Fish",
        "updated": "2019-10-18T17:10:18"
    }
}
```


ERROR RESPONSES

- Validation Error (400): If title/description/category_id doesn’t pass the validation test.
- Duplicate Entity (400): If client tries to create an item with the title that has already used by another item.
- Unauthorized (401): If client hasn’t logged in yet.
- Not Found (404): If the category with that category_id doesn’t exist.

d. **PUT**

/items/:item_id

Allow client to update an item.

REQUEST

- **Parameters**:

Name	Type	Description	Example
item_id	Int	Identifier of the item that the client wants to update	1

- **Header**:

Key	Value
Content-Type	application/json
Authorization	Bearer jwt_token

- **Body**:

Name	Type	Description	Example
title	String	Title of the item	Minecraft Swordy
description	String	Description of the item	Minecraft is dug

SUCCESSFUL RESPONSE

- **HTTP code**: **200 OK**
- **Headers**:

Key	**Value**
Content-Type	application/json

- **Response:** All of these fields will be wrapped by an object called "**data**"

Name	**Type**	**Description**
created	String	The day this entity was created
creator_id	Int	Identifier of the user that created this entity
description	String	The description of this entity
id	Int	Identifier of this entity
title	String	Title of this entity
updated	String	The latest day this entity was updated
category.id	Int	Identifier of the category to which this item belongs
category.title	String	The title of the category to which this item belongs

- **Response example**:


```
{
   "data": {
       "category": {
           "id": 2,
           "title": "Minecrafdwedwedvdvdfvdtyee"
       },
       "created": "2019-10-18T17:03:01",
       "creator_id": 1,
       "description": "Very shiny...",
```

```
        "id": 6,
        "title": "Crab",
        "updated": "2019-10-18T17:03:01"
    }
}
```


ERROR RESPONSES

- Validation Error (400): If title/description/category_id doesn’t pass the validation test.
- Duplicated Entity (400): If an item with the title has already existed.
- Unauthorized (401): If client hasn’t logged in yet.
- Forbidden (403): If client tries to update another user’s item.
- Not Found (404): If the item with that id doesn’t exist or the category having category_id doesn’t exist.

e. **DELETE**

/items/:item_id

Allow client to delete an item.

REQUEST

- **Parameters**:

Name	Type	Description	Example
item_id	Int	Identifier of the item that the client wants to delete	1

- **Header**:

Key	Value
Content-Type	application/json
Authorization	Bearer jwt_token

- **Body**: No content

SUCCESSFUL RESPONSE

- **HTTP code**: **204 No Content**
- **Headers**:

Key	Value
Content-Type	application/json

ERROR RESPONSES

- Unauthorized (401): If client hasn't logged in yet.
- Forbidden (403): If client tries to delete another user's item.
- Not Found (404): If client tries to delete an item that doesn't exist.

5. **User**

a. **POST**

/users

Allow client to register.

REQUEST

- **Parameters**: No content
- **Header**:

Key	**Value**
Content-Type	application/json

- **Body**:

Name	**Type**	**Description**	**Example**
email	String	Email of the user	admin@admin.com
password	String	Password of the user	123456

SUCCESSFUL RESPONSE

- **HTTP code**: **200 OK**
- **Headers**:

Key	**Value**
Content-Type	application/json

- **Response:**

Name	**Type**	**Description**	**Example**
access_token	String	JWT token	"....."
user.id	Int	Identifier of the registered user	1
user.email	String	Email of the registered user	admin@gmail.com

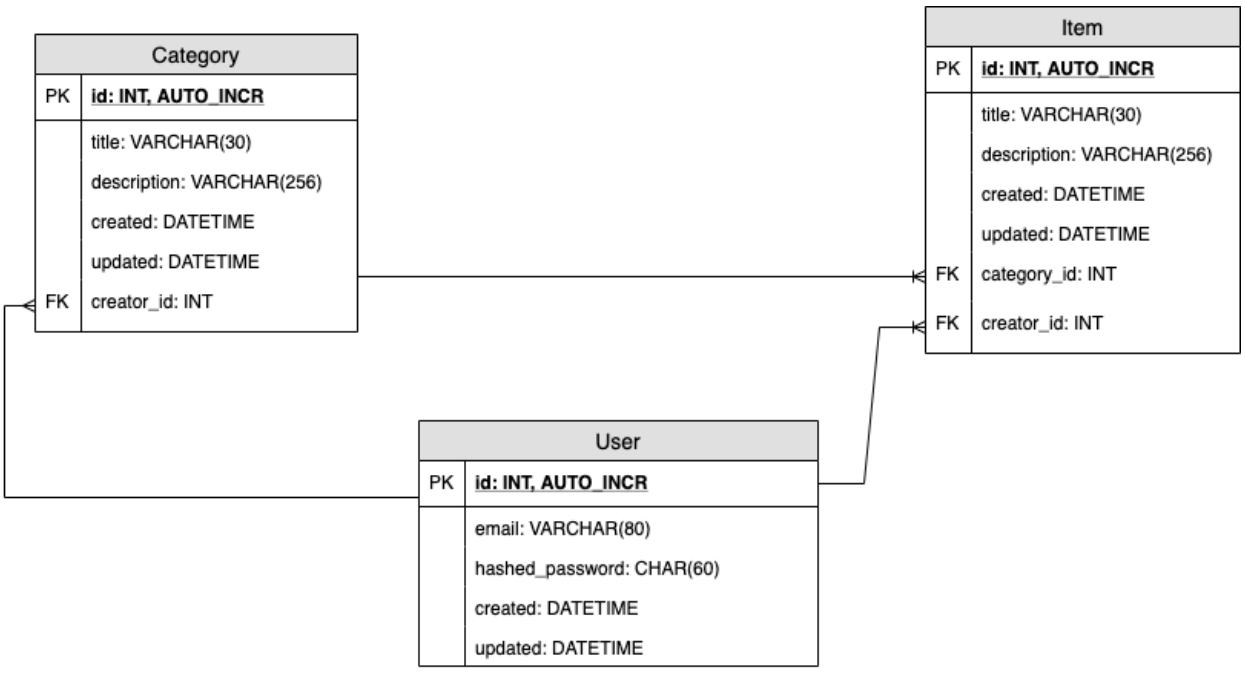
- **Response example**:


```
{
   "access_token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE1NzEyODAxMTEsIm5iZiI6MTU3MTI4MDExMS
wianRpIjoiMzU3YWJjNTYtNjRjOC00MzAzLWE3YmMtMDIxNmNmYWVhYmI4IiwiZXhwIjoxNTcxMjgxMDExLC
JpZGVudGl0eSI6MSwiZnJlc2giOmZhbHNlLCJ0eXBlIjoiYWNjZXNzIn0.M_r1meDSt_jpSW3SPjI5kVwyCQ
mgPKoBoPM-59NYJVU",
   "user": {
          "id": "1",
          "email": "admin@gmail.com",
   }
}
```


ERROR RESPONSES

- Validation Error (400): If email/password doesn't pass the validation test.
- Duplicate Entity (400): If client tries to create a user with the email that has already used by another user.

II. Entity Relationship Diagram



III. Folder Structure