# Assignment 5: API Specifications

Demetrios Kopatsis, CS 493 Fall 2023

Deployed URL: https://kopatsis-cs-493-final.uw.r.appspot.com

## Data Model:

User:

| Property | Data Type | Notes |
|---|---|---|
| id | Integer | Autogenerated by Datastore, identifying property |
| Subject | String | "sub" property of Auth0 JWT, used as identification on the Auth0 end<br>Required, created automatically from login |
| Name | String | Name of user from Auth0 sign-in<br>Required, created automatically from login |

Balloon:

| Property | Data Type | Notes |
|---|---|---|
| id | Integer | Autogenerated by Datastore, identifying property |
| Title | String | Defined in POST request for creation –<br>Title of balloon<br>Required |
| Size | Integer | Defined in POST request for creation –<br>Size of balloon in inches<br>Required |
| Price | Float | Defined in POST request for creation –<br>Price of balloon in dollars<br>Required |
| ManufacturerID | String | Automatically assigned by JWT –<br>Subject property of user that creates this balloon<br>Required, created automatically from POST |
| Collections | [] of {} | Generated upon request from API –<br>Title, ID, and Self attributes of all collections this balloon is a part of, set as array of json attributes |
| self | String | URL to resource, defined by API and not stored directly within Datastore database |

Collection:

| **Property** | **Data Type** | **Notes** |
|---|---|---|
| id | Integer | Autogenerated by Datastore, identifying property |
| Title | String | Defined in POST request for creation – <br>Title of collection <br>Required |
| Description | String | Defined in POST request for creation – <br>Short description of collection <br>Required |
| Archived | Bool | Defined in POST request for creation – <br>True or false representing whether collection is archived <br>Required |
| Balloons | [] of {} | Generated upon request from API – <br>Title, ID, and Self attributes of all balloons this collection contains, set as array of json attributes <br>Stored in Datastore as an array of strings with each ID |
| self | String | URL to resource, defined by API and not stored directly within Datastore database |

## Relationships:

User.-> Balloon: 1:Many, User's Subject field is used as the ManufacturerID in Balloon upon POST, from provided JWT. Cannot be changed. Every Balloon has exactly 1 User and every user can have 0 or more Balloons.

Balloon <-> Collection: Many:Many, Every Collection can contain 0 or more Balloons and every Balloon can be in 0 or more Collections. Relationship stored on Datastore in Balloons attribute of Collection and for both Balloons and Collections, will be generated to the format from table above. Can be changed by adding or removing Balloons from Collections.

## User Entity:

- Though Datastore ID is generated for each user, Subject is ultimately used as identifier, especially when using endpoints with JWT.
- Only the JWT needs to be provided to requests to endpoints requiring it, no ID is specifically required in request.
- Provided a request is made with a valid JWT, the JWT's "sub" property is used to identify the user making requests to an endpoint.

## POST /balloons – Create a new Balloon

Request:

Path Parameters – None

Request Body – Required

Request Body Format – JSON

Bearer Token Requirement – Valid JWT

Request Attributes –

| **Property** | **Required** | **Description** |
|---|---|---|
| title | Yes | Balloon’s title (string) |
| size | Yes | Balloon’s size (int) |
| price | Yes | Balloon’s price (float) |

Example:

```
{
    "title": "Balloon Number 1",
    "size": 12,
    "price": 4.99
}
```

Response:

Response Body Format – JSON

Response Statuses

| **Outcome** | **Code** | **Notes** |
|---|---|---|
| Success | 201 Created | |
| Failure | 406 Not Acceptable | Incorrect “Accept” data type (doesn’t include JSON) |
| Failure | 401 Unauthorized | Invalid JWT |
| Failure | 400 Bad Request | Other error that doesn’t allow creation in datastore |

Example Responses:

201 Created:

```
{
    "collections": [],
    "id": 5662792157757440,
    "manufacturerID": "auth0|6555a6cd6ac7eefb66a5bb82",
    "price": 4.99,
    "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/balloons/5662792157757440",
    "size": 12,
    "title": "Balloon Number 1"
}
```

406 Not Acceptable:

```
{
    "Error": "Not Acceptable MIME type for Accept"
}
```

401 Unauthorized:

```
{
    "Error": "No correctly formatted token request"
}
```

400 Bad Request:

```
{
    "Details": "Failed to put data into datastore",
    "Error": "Bad Request"
}
```

[Note: 400 will only occur if something unexpected goes wrong, such as Datastore being down or an input that is somehow incompatible with Datastore]

## PUT /balloons/:balloon_id – Edit all attributes for an existing balloon

Request:

Path Parameters –

| Property | Required | Description |
|---|---|---|
| balloon_id | Yes | Balloon’s id |

Request Body – Required

Request Body Format – JSON

Bearer Token Requirement – Valid JWT

Request Attributes –

| Property | Required | Description |
|---|---|---|
| title | Yes | Balloon’s title (string) |
| size | Yes | Balloon’s size (int) |
| price | Yes | Balloon’s price (float) |

Example:

```
{
    "title": "Balloon Number 1 Put",
    "size": 12,
    "price": 4.99
}
```

Response:

Response Body Format – JSON

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Success | 200 OK | Location Header contains URL of resource |
| Failure | 404 Not Found | Id attribute does not match an existing balloon |
| Failure | 406 Not Acceptable | Incorrect “Accept” data type (doesn’t include JSON) |
| Failure | 401 Unauthorized | Invalid JWT |
| Failure | 403 Forbidden | JWT sub does not match balloon ManufacturerID |
| Failure | 400 Bad Request | Other error that doesn’t allow editing in datastore |

Example Responses:

200 OK:

```
{
    "collections": [],
    "id": 5662792157757440,
    "manufacturerID": "auth0|6555a6cd6ac7eefb66a5bb82",
    "price": 4.99,
    "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/balloons/5662792157757440",
    "size": 12,
    "title": "Balloon Number 1 Put"
}
```

404 Not Found:

```
{
    "Error": "Balloon with parameter id does not exist"
}
```

406 Not Acceptable:

```
{
    "Error": "Not Acceptable MIME type for Accept"
}
```

401 Unauthorized:

```
{
    "Error": "No correctly formatted token request"
}
```

403 Forbidden:

```
{
    "Error": "Forbidden"
}
```

400 Bad Request:

```
{
    "Details": "Failed to put data into datastore",
    "Error": "Bad Request"
}
```

[Note: 400 will only occur if something unexpected goes wrong, such as Datastore being down or an input that is somehow incompatible with Datastore]

## PATCH /balloons/:balloon_id – Edit any attribute(s) for an existing balloon

Request:

Path Parameters –

| Property | Required | Description |
|---|---|---|
| balloon_id | Yes | Balloon's id |

Request Body – Required

Request Body Format – JSON

Bearer Token Requirement – Valid JWT

Request Attributes –

| Property | Required | Description |
|---|---|---|
| title | No | Balloon's title (string) |
| size | No | Balloon's size (int) |
| price | No | Balloon's price (float) |

Example:

```
{
   "title": "Balloon Number 1 Patch",
}
```

Response:

Response Body Format – JSON

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Success | 200 OK | Location Header contains URL of resource |
| Failure | 404 Not Found | Id attribute does not match an existing balloon |
| Failure | 406 Not Acceptable | Incorrect "Accept" data type (doesn't include JSON) |
| Failure | 401 Unauthorized | Invalid JWT |
| Failure | 403 Forbidden | JWT sub does not match balloon ManufacturerID |
| Failure | 400 Bad Request | Other error that doesn't allow editing in datastore |

Example Responses:

200 OK:

```
{
   "collections": [],
   "id": 5662792157757440,
   "manufacturerID": "auth0|6555a6cd6ac7eefb66a5bb82",
   "price": 4.99,
   "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/balloons/5662792157757440",
   "size": 12,
   "title": "Balloon Number 1 Patch"
}
```

404 Not Found:

```
{
    "Error": "Balloon with parameter id does not exist"
}
```

406 Not Acceptable:

```
{
    "Error": "Not Acceptable MIME type for Accept"
}
```

401 Unauthorized:

```
{
    "Error": "No correctly formatted token request"
}
```

403 Forbidden:

```
{
    "Error": "Forbidden"
}
```

400 Bad Request:

```
{
    "Details": "Failed to put data into datastore",
    "Error": "Bad Request"
}
```

[Note: 400 will only occur if something unexpected goes wrong, such as Datastore being down or an input that is somehow incompatible with Datastore]

## GET /balloons/:balloon_id – Read an existing balloon

Request:

Path Parameters –

| Property | Required | Description |
|---|---|---|
| balloon_id | Yes | Balloon’s id |

Request Body – None

Bearer Token Requirement – Valid JWT

Response:

Response Body Format – JSON.

Response Statuses

| | | |
|---|---|---|
| Success | 200 OK | Location Header contains URL of resource |
| Failure | 404 Not Found | Id attribute does not match an existing balloon |
| Failure | 406 Not Acceptable | Incorrect “Accept” data type (doesn’t include JSON) |
| Failure | 401 Unauthorized | Invalid JWT |
| Failure | 403 Forbidden | JWT sub does not match balloon ManufacturerID |
| Failure | 400 Bad Request | Other error that doesn’t allow working in datastore |

Example Responses:

200 OK:

```
{
    "collections": [],
    "id": 5662792157757440,
    "manufacturerID": "auth0|6555a6cd6ac7eefb66a5bb82",
    "price": 4.99,
    "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/balloons/5662792157757440",
    "size": 12,
    "title": "Balloon Number 1"
}
```

404 Not Found:

```
{
    "Error": "Balloon with parameter id does not exist"
}
```

406 Not Acceptable:

```
{
    "Error": "Not Acceptable MIME type for Accept"
}
```

401 Unauthorized:

```
{
    "Error": "No correctly formatted token request"
}
```

403 Forbidden:

```
{
    "Error": "Forbidden"
}
```

400 Bad Request:

```
{
    "Details": "Failed to get data from datastore",
    "Error": "Bad Request"
}
```

[Note: 400 will only occur if something unexpected goes wrong, such as Datastore being down or an input that is somehow incompatible with Datastore]

## GET /balloons – Read all existing balloons for a manufacturer/user

Request:

Path Parameters – None

Request Body – None

Bearer Token Requirement – Valid JWT

Response:

Response Body Format – JSON.

Response Statuses

| | | |
|---|---|---|
| Success | 200 OK | Location Header contains URL of resource |
| Failure | 406 Not Acceptable | Incorrect "Accept" data type (doesn't include JSON) |
| Failure | 401 Unauthorized | Invalid JWT |
| Failure | 400 Bad Request | Other error that doesn't allow working in datastore |

Example Responses:

Example Responses:

200 OK:

```
{
    "results": [
        {
            "collections": [],
            "id": 5662792157757440,
            "manufacturerID": "auth0|6555a6cd6ac7eefb66a5bb82",
            "price": 4.99,
            "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/balloons/5662792157757440",
            "size": 12,
            "title": "Balloon Number 1"
        },
        {
            "collections": [],
            "id": 5677459739508736,
            "manufacturerID": "auth0|6555a6cd6ac7eefb66a5bb82",
            "price": 4.99,
            "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/balloons/5677459739508736",
            "size": 12,
            "title": "Balloon Number 2"
        }
    ]
}
```

[Note: if more than 5 balloons exist, only the first 5 will be displayed with the next batch available via “Next” attribute link for pagination]

406 Not Acceptable:

```
{
    "Error": "Not Acceptable MIME type for Accept"
}
```

401 Unauthorized:

```
{
    "Error": "No correctly formatted token request"
}
```

400 Bad Request:

```
{
    "Details": "Failed to get data from datastore",
    "Error": "Bad Request"
}
```

[Note: 400 will only occur if something unexpected goes wrong, such as Datastore being down or an input that is somehow incompatible with Datastore]

## DELETE /balloons/:balloon_id – Delete an existing balloon

NOTE: Deleting a balloon will automatically remove it from all collections it is in

Request:

Path Parameters –

| **Property** | **Required** | **Description** |
|---|---|---|
| Balloon_id | Yes | Balloon’s id |

Request Body – None

Bearer Token Requirement – Valid JWT

Response:

Response Body Format – None

Response Statuses

| **Outcome** | **Code** | **Notes** |
|---|---|---|
| Success | 204 No Content | |
| Failure | 404 Not Found | Id attribute does not match an existing ballon |
| Failure | 401 Unauthorized | Invalid JWT |
| Failure | 403 Forbidden | JWT sub does not match balloon ManufacturerID |
| Failure | 400 Bad Request | Other error that doesn’t allow working in datastore |

Example Responses:

204 No Content:

Status: 204 No Content

404 Not Found:

```
{
    "Error": "Balloon with parameter id does not exist"
}
```

401 Unauthorized:

```
{
    "Error": "No correctly formatted token request"
}
```

401 Unauthorized:

```
{
    "Error": "No correctly formatted token request"
}
```

403 Forbidden:

```
{
    "Error": "Forbidden"
}
```

400 Bad Request:

```
{
    "Details": "Failed to get data from datastore",
    "Error": "Bad Request"
}
```

## POST /collections – Create a new Collection

Request:

Path Parameters – None

Request Body – Required

Request Body Format – JSON

Request Attributes –

| **Property** | **Required** | **Description** |
|---|---|---|
| title | Yes | Collection’s title (string) |
| description | Yes | Collection’s description (string) |
| archived | Yes | Collection’s archived status (bool) |

Example:

```
{
    "title": "Balloon Collection 1",
    "description": "description of balloon collection here",
    "archived": false
}
```

Response:

Response Body Format – JSON

Response Statuses

| **Outcome** | **Code** | **Notes** |
|---|---|---|
| Success | 201 Created | |
| Failure | 406 Not Acceptable | Incorrect “Accept” data type (doesn’t include JSON) |
| Failure | 400 Bad Request | Other error that doesn’t allow creation in datastore |

Example Responses:

201 Created:

```
{
    "archived": false,
    "balloons": [],
    "description": "description of balloon collection here",
    "id": 5710150413320192,
    "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/collections/5710150413320192",
    "title": "Balloon Collection 1"
}
```

406 Not Acceptable:

```
{
    "Error": "Not Acceptable MIME type for Accept"
}
```

400 Bad Request:

```
{
    "Details": "Failed to put data into datastore",
    "Error": "Bad Request"
}
```

[Note: 400 will only occur if something unexpected goes wrong, such as Datastore being down or an input that is somehow incompatible with Datastore]

## PUT /collections/:collection_id – Edit all attributes for an existing collection

Request:

Path Parameters –

| Property | Required | Description |
|---|---|---|
| collection_id | Yes | Collection’s id |

Request Body – Required

Bearer Token Requirement – Valid JWT

Request Attributes –

| Property | Required | Description |
|---|---|---|
| title | Yes | Collection’s title (string) |
| description | Yes | Collection’s description (string) |
| archived | Yes | Collection’s archived status (bool) |

Example:

```
{
    "title": "Balloon Collection 1 Put",
    "description": "description of balloon collection here",
    "archived": false
}
```

Response:

Response Body Format – JSON

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Success | 200 OK | Location Header contains URL of resource |
| Failure | 404 Not Found | Id attribute does not match an existing collection |
| Failure | 406 Not Acceptable | Incorrect “Accept” data type (doesn’t include JSON) |
| Failure | 400 Bad Request | Other error that doesn’t allow editing in datastore |

Example Responses:

200 OK:

```
{
    "archived": false,
    "balloons": [],
    "description": "description of balloon collection here",
    "id": 5710150413320192,
    "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/collections/5710150413320192",
    "title": "Balloon Collection 1 Put"
}
```

404 Not Found:

```
{
    "Error": "Collection with parameter id does not exist"
}
```

406 Not Acceptable:

```
{
    "Error": "Not Acceptable MIME type for Accept"
}
```

400 Bad Request:

```
{
    "Details": "Failed to put data into datastore",
    "Error": "Bad Request"
}
```

[Note: 400 will only occur if something unexpected goes wrong, such as Datastore being down or an input that is somehow incompatible with Datastore]

## PATCH /collections/:collection_id – Edit any attribute(s) for an existing collection

Request:

Path Parameters –

| Property | Required | Description |
|---|---|---|
| collection_id | Yes | Collection’s id |

Request Body – Required

Bearer Token Requirement – Valid JWT

Request Attributes –

| Property | Required | Description |
|---|---|---|
| title | No | Collection’s title (string) |
| description | No | Collection’s description (string) |
| archived | No | Collection’s archived status (bool) |

Example:

```
{
    "title": "Balloon Collection 1 Patch",
}
```

Response:

Response Body Format – JSON

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Success | 200 OK | Location Header contains URL of resource |
| Failure | 404 Not Found | Id attribute does not match an existing collection |
| Failure | 406 Not Acceptable | Incorrect “Accept” data type (doesn’t include JSON) |
| Failure | 400 Bad Request | Other error that doesn’t allow editing in datastore |

Example Responses:

200 OK:

```
{
    "archived": false,
    "balloons": [],
    "description": "description of balloon collection here",
    "id": 5710150413320192,
    "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/collections/5710150413320192",
    "title": "Balloon Collection 1 Patch"
}
```

404 Not Found:

```
{
    "Error": "Collection with parameter id does not exist"
}
```

406 Not Acceptable:

```
{
    "Error": "Not Acceptable MIME type for Accept"
}
```

400 Bad Request:

```
{
    "Details": "Failed to put data into datastore",
    "Error": "Bad Request"
}
```

[Note: 400 will only occur if something unexpected goes wrong, such as Datastore being down or an input that is somehow incompatible with Datastore]

## GET /collections/:collection_id – Read an existing collection

Request:

Path Parameters –

| **Property** | **Required** | **Description** |
|---|---|---|
| collection_id | Yes | Collection's id |

Request Body – None

Response:

Response Body Format – JSON.

Response Statuses

| | | |
|---|---|---|
| Success | 200 OK | Location Header contains URL of resource |
| Failure | 404 Not Found | Id attribute does not match an existing collection |
| Failure | 406 Not Acceptable | Incorrect "Accept" data type (doesn't include JSON) |
| Failure | 400 Bad Request | Other error that doesn't allow working in datastore |

Example Responses:

200 OK:

```
{
    "archived": false,
    "balloons": [],
    "description": "description of balloon collection here",
    "id": 5710150413320192,
    "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/collections/5710150413320192",
    "title": "Balloon Collection 1"
}
```

404 Not Found:

```
{
    "Error": "Collection with parameter id does not exist"
}
```

406 Not Acceptable:

```
{
    "Error": "Not Acceptable MIME type for Accept"
}
```

400 Bad Request:

```
{
    "Details": "Failed to get data from datastore",
    "Error": "Bad Request"
}
```

[Note: 400 will only occur if something unexpected goes wrong, such as Datastore being down or an input that is somehow incompatible with Datastore]

## GET /collections – Read all existing collections

Request:

Path Parameters – None

Request Body – None

Response:

Response Body Format – JSON.

Response Statuses

| | | |
|---|---|---|
| Success | 200 OK | Location Header contains URL of resource |
| Failure | 406 Not Acceptable | Incorrect “Accept” data type (doesn’t include JSON) |
| Failure | 400 Bad Request | Other error that doesn’t allow working in datastore |

Example Responses:

200 OK:

```
{
    "results": [
        {
            "archived": false,
            "balloons": [],
            "description": "description of balloon collection here",
            "id": 5710150413320192,
            "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/collections/5710150413320192",
            "title": "Balloon Collection 1"
        },
        {
            "archived": false,
            "balloons": [],
            "description": "description of balloon collection here",
            "id": 5729450050191360,
            "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/collections/5729450050191360",
            "title": "Balloon Collection 2"
        }
    ]
}
```

[Note: if more than 5 balloons exist, only the first 5 will be displayed with the next batch available via “Next” attribute link for pagination]

406 Not Acceptable:

```
{
    "Error": "Not Acceptable MIME type for Accept"
}
```

400 Bad Request:

```
{
    "Details": "Failed to get data from datastore",
    "Error": "Bad Request"
}
```

[Note: 400 will only occur if something unexpected goes wrong, such as Datastore being down or an input that is somehow incompatible with Datastore]

## DELETE /collections/:collection_id – Delete an existing collection

Request:

Path Parameters –

| Property | Required | Description |
|---|---|---|
| collection_id | Yes | Collection's id |

Request Body – None

Response:

Response Body Format – None

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Success | 204 No Content | |
| Failure | 404 Not Found | Id attribute does not match an existing collection |
| Failure | 400 Bad Request | Other error that doesn't allow working in datastore |

Example Responses:

204 No Content:

Status: 204 No Content

404 Not Found:

```
{
    "Error": "Collection with parameter id does not exist"
}
```

400 Bad Request:

```
{
    "Details": "Failed to get data from datastore",
    "Error": "Bad Request"
}
```

## POST /collections/:collection_id/balloons/:balloon_id – Add a Balloon to a Collection

Request:

Path Parameters –

| Property | Required | Description |
|---|---|---|
| collection_id | Yes | Collection's id |
| balloon_id | Yes | Balloon's id |

Request Body – None

Request Body Format – JSON

Bearer Token Requirement – Valid JWT

Response:

Response Body Format – JSON

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Success | 200 OK | Collection is returned |
| Failure | 406 Not Acceptable | Incorrect "Accept" data type (doesn't include JSON) |
| Failure | 401 Unauthorized | Invalid JWT |
| Failure | 403 Forbidden | JWT does not match Balloon ManufacturerID |
| Failure | 404 Not Found | Balloon and/or collection ID does not match an existing balloon and/or collection |
| Failure | 400 Bad Request | Other error that doesn't allow creation in datastore, Balloon already in collection |

Example Responses:

200 OK:

```
{
    "archived": false,
    "balloons": [
        {
            "ID": "5662792157757440",
            "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/balloons/5662792157757440",
            "title": "Balloon Number 1"
        }
    ],
    "description": "description of balloon collection here",
    "id": 5710150413320192,
    "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/collections/5710150413320192",
    "title": "Balloon Collection 1"
}
```

406 Not Acceptable:

```
{
    "Error": "Not Acceptable MIME type for Accept"
}
```

401 Unauthorized:

```
{
    "Error": "No correctly formatted token request"
}
```

403 Forbidden:

```
{
    "Error": "Forbidden"
}
```

404 Not Found:

```
{
    "Error": "Collection with parameter id does not exist"
}
```

400 Bad Request:

```
{
    "Details": "Balloon is already in collection",
    "Error": "Bad request"
}
```

## DELETE /collections/:collection_id/balloons/:balloon_id – Remove a Balloon from a Collection

Request:

Path Parameters –

| Property | Required | Description |
|---|---|---|
| collection_id | Yes | Collection's id |
| balloon_id | Yes | Balloon's id |

Request Body – None

Request Body Format – JSON

Bearer Token Requirement – Valid JWT

Response:

Response Body Format – JSON

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Success | 200 OK | Collection is returned |
| Failure | 406 Not Acceptable | Incorrect "Accept" data type (doesn't include JSON) |
| Failure | 401 Unauthorized | Invalid JWT |
| Failure | 403 Forbidden | JWT does not match Balloon ManufacturerID |
| Failure | 404 Not Found | Balloon and/or collection ID does not match an existing balloon and/or collection |
| Failure | 400 Bad Request | Other error that doesn't allow creation in datastore, Balloon not in collection |

Example Responses:

200 OK:

```
{
    "archived": false,
    "balloons": [],
    "description": "description of balloon collection here",
    "id": 5710150413320192,
    "self": "https://kopatsis-cs-493-
final.uw.r.appspot.com/collections/5710150413320192",
    "title": "Balloon Collection 1"
}
```

406 Not Acceptable:

```
{
    "Error": "Not Acceptable MIME type for Accept"
}
```

401 Unauthorized:

```
{
    "Error": "No correctly formatted token request"
}
```

403 Forbidden:

```
{
    "Error": "Forbidden"
}
```

404 Not Found:

```
{
    "Error": "Collection with parameter id does not exist"
}
```

400 Bad Request:

```
{
    "Details": "Balloon is not in in collection",
    "Error": "Bad request"
}
```

## GET /users – Read all existing users

Request:

Path Parameters – None

Request Body – None

Response:

Response Body Format – JSON.

Response Statuses

| | | |
|---|---|---|
| Success | 200 OK | |
| Failure | 400 Bad Request | Other error that doesn't allow working in datastore |

Example Responses:

200 OK:

```
[
    {
        "User ID": "google-oauth2|114140222694061328999",
        "User Name": "Demetrios Kopatsis"
    },
    {
        "User ID": "auth0|6555a6cd6ac7eefb66a5bb82",
        "User Name": "jkopatsis@gmail.com"
    },
    {
        "User ID": "auth0|657136a557cc8c91eb444ca9",
        "User Name": "kopatsid@oregonstate.edu"
    },
    {
        "User ID": "google-oauth2|115226569657097221348",
        "User Name": "Demetrios Kopatsis"
    }
]
```

400 Bad Request:

```
{
    "Details": "Failed to get data from datastore",
    "Error": "Bad Request"
}
```

## Endpoints that are NOT Accepted:

### PUT /balloons – Attempted to edit with no id in parameter

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is correct, method cannot be used on it |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

### PATCH /balloons – Attempted to edit with no id in parameter

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is correct, method cannot be used on it |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

### DELETE /balloons – Attempted to delete with no id in parameter

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is correct, method cannot be used on it |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

POST /balloons/:balloon_id – Attempted to create with id in parameter

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is correct, method cannot be used on it |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

PUT /collections – Attempted to edit with no id in parameter

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is correct, method cannot be used on it |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

PATCH /collections – Attempted to edit with no id in parameter

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is correct, method cannot be used on it |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

DELETE /collections – Attempted to delete with no id in parameter

Response Statuses

| Outcome | Code | Notes |
|---|---|---|

| | | |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is correct, method cannot be used on it |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

POST /collections/:collection_id/balloons – Attempted to work with incorrect URL

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is incorrect |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

PUT /collections/:collection_id/balloons – Attempted to work with incorrect URL

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is incorrect |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

PATCH /collections/:collection_id/balloons – Attempted to work with incorrect URL

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is incorrect |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

GET /collections/:collection_id/balloons – Attempted to work with incorrect URL

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is incorrect |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

DELETE /collections/:collection_id/balloons – Attempted to work with incorrect URL

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is incorrect |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

PUT /collections/:collection_id/balloons/ballon_id – Attempted to edit interaction endpoint (only add and remove functionality allowed on this URL)

Response Statuses

| Outcome | Code | Notes |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is correct, method cannot be used on it |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

PATCH /collections/:collection_id/balloons/ballon_id – Attempted to edit interaction endpoint (only add and remove functionality allowed on this URL)

Response Statuses

| **Outcome** | **Code** | **Notes** |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is correct, method cannot be used on it |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```

GET /collections/:collection_id/balloons/ballon_id – Attempted to read interaction endpoint (only add and remove functionality allowed on this URL)

Response Statuses

| **Outcome** | **Code** | **Notes** |
|---|---|---|
| Failure | 405 Method Not Allowed | URL is correct, method cannot be used on it |

Example:

```
{
    "Error": "Incorrect method in endpoint"
}
```