App URL: https://final-phamchri.wl.r.appspot.com
Acct Creation URL: Same As Above

Handbag API Spec

Cloud Application

Fall 2021

Data Model

Create a Bag

View a Bag (protected)

View all Bags (protected)

Update a Bag (protected)

Delete a Bag /remove from a member (protected)

Create an Item

View an Item

View all Items

Update an Item

Delete an Item

Assign an Item to a Bag (protected)

Remove an Item from a Bag (protected)

View all items for a given Bag (protected)

View all Members

Assign a Bag to a Member (protected)

Data Model

The app stores three kinds of entities in Datastore: Member, Bags, and Items.

Members

Property	Data Type	Notes
id	Integer	The id of the bag. Datastore automatically generates it.
name	String	Name of the member.
age	Integer	The age of the member.
bags	Embedded array	Has id and link to owned bag for each element

Bags

Property	Data Type	Notes
id	Integer	The id of the bag. Datastore automatically generates it.
model	String	Name/Type of the bag.
brand	String	The name of the company that designed the bag.
private	Boolean	Indicates if the bag can be shown publicly
owner	Integer	Id of the member who owns this bag
items	Embedded array	Has id and link to item for each element

Items

Property	Data Type	Notes
id	Integer	The id of the item. Datastore automatically generates it.
name	String	The name of the item
carrier	Embedded entity	Contains id, brand, and link to the bag carrying the item
category	String	What kind of product is the item
create_date	Date and Time	Date the item was created/modified

Create a Bag

Allows you to create a new bag.

```
POST /bags
```

Request

Path Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Description	Required?
model	The name of the bag.	Yes
brand	The designer of the bag.	Yes

Request Body Example

```
{
  "model": "Birkin",
  "brand": "Hermes"
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201 Created	
Failure	400 Bad Request	If the request is missing any of the 2 required attributes, the bag must not be created, and 400 status code must be returned.

Response Examples

- Datastore will automatically generate an ID and store it with the entity being created. The app sends back this value in the response body as shown in the example.
- The `items and owner` attribute is initialized to null and does not need to be included in the body during creation of a bag. `private` is initialized to False as it's a new bag.
- The `self` attribute will contain the live link to the REST resource corresponding to this bag. In other words, this is the URL to get this newly created bag. The `self` attribute is not stored in Datastore

Success

```
Status: 201 Created
```

```
{
  "model": "Birkin",
  "brand": "Hermes",
  "private": False,
  "owner": null,
  “items”: null,
  "self": "https://<your-app>/bags/123"
}
```

Failure

Status: 400 Bad Request

```
{
"Error":  "The request object is missing at least one of the required attributes"
}
```

View a Bag

Allows a member to GET an existing bag

GET /bags/:bag_id

Request

Path Parameters

Name	Description
bag_id	ID of the bag

Request Headers

Authorization token required

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	404 Not Found	No bag with this bag_id exists

Response Examples

Success

```
Status: 200 OK

{
"id": 123,
"model": "Birkin",
"brand": "Hermes",
"private": True,
"owner": 432590,
"items":[
       {
       "id": 5678,
       "self":"https://appspot.com/items/5678"},
       {
       "id": 8765,
       "self":"https://appspot.com/items/8765"}
],
"self": "https://<your-app>/bags/123"
}
```

Failure

```
Status: 404 Not Found
```

```
{
"Error":  "No bag with this bag_id exists"
}
```

View all Bags

List all the bags.

```
GET /bags
```

Request

Path Parameters

None

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	

Response Examples

- Viewing all bags implements pagination.
- 3 bags should be returned per page.
- There must be a ‘next’ link to get to the next page of results, except for the last page.

Success

```
Status: 200 OK

{
“bags”: [
        {
        "id": 123,
        "name": "Sea Witch",
        "type": "Catamaran",
        "length": 28,
        "items":[
                {
                "id": 5678,
                "self":"https://appspot.com/items/5678"},
                {
                "id": 8765,
                "self":"https://appspot.com/items/8765"}
                ],
        "self": "https://<your-app>/bags/123"
        },
        {
        "id": 456,
        "name": "Adventure",
        "type": "Sailbag",
        "length": 50,
       "items":[
             {
             "id": 365,
             "self":"https://appspot.com/items/365"},
             {
             "id": 4821,
             "self":"https://appspot.com/items/4821"}
             ],
        "self": "https://<your-app>/bags/456"
       },
       {
       "id": 789,
       "name": "Hocus Pocus",
       "type": "Sailbag",
       "length": 100,
       "items":[
             {
             "id": 2624,
             "self":"https://appspot.com/items/2624"},
             {
             "id": 9087,
             "self":"https://appspot.com/items/9087"}
             ],
        "self": "https://<your-app>/bags/789"
             },
 ],
“next”: “https://<your-app>/bags?cursor=path_to_next_group”
}
```

Update a Bag

Edit information about a bag.

PUT /bags/:bag_id
PATCH /bags/:bag_id

Request

Path Parameters

Name	Description
bag_id	ID of the bag

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Description	Required?
name	The name of the bag.	Yes
type	The type of the bag. E.g., Sailbag, Catamaran, etc.	Yes

length	Length of the bag in feet.	Yes

Request Body Example

```
{
  "name": "Sea Witch",
  "type": "Catamaran",
  "length": 28
}
```

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	204 No Content	Succeeds only if the bag_id exists
Failure	404 Not Found	The specified bag does not exist

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found

{
"Error":  "The specified bag does not exist"
}
```

Delete a Bag

Allows you to delete a bag. Note that deleting the bag dumps all items inside the bag.

```
DELETE /bags/:bag_id
```

Request

Path Parameters

Name	Description
bag_id	ID of the bag

Request Body

None

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	
Failure	404 Not Found	No bag with this bag_id exists

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found

{
"Error":  "No bag with this bag_id exists"
}
```

Create an Item

Allows you to create a new item.

```
POST /items
```

Request

Path Parameters

None

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Description	Required?
volume	The size of the item	Yes
content	The type of items inside the item	Yes

Request Body Example

```
{
  "volume": 5,
  "content": "LEGO Blocks"
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201 Created	
Failure	400 Bad Request	If the request is missing any required attribute, the item must not be created, and 400 status code must be returned.

Response Examples

- Datastore will automatically generate an ID and store it with the entity being created. This value needs to be sent in the response body as shown in the example.
- The value of the attribute `carrier` is the entity of the bag currently at this item. For a newly created bag the value of `carrier` should be null.
- The value of the attribute `self` is a live link to the REST resource corresponding to this item. In other words, this is the URL to get this newly created item. This attribute is not in Datastore.

Success

```
Status: 201 Created

{
        "id": 123,
         "volume": 5,
        "carrier": null,
        "content": "LEGO Blocks",
        "creation_date": "10/18/2020",
        "self": "https://<your-app>/items/123"
}
```

Failure

```
Status: 400 Bad Request

{
"Error":  "The request object is missing at least one required attribute"
}
```

View an Item

Allows you to get an existing item.

```
GET /items/:item_id
```

Request

Path Parameters

Name	Description
item_id	ID of the item

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	
Failure	404 Not Found	No item with this item_id exists

Response Examples

Success

```
Status: 200 OK

{
"id": 123,
volume": 5,
"carrier": {
        "id": 1234,
        "name": "Sea Witch",
        "self": "https://appspot.com/bags/1234"
        },
"content": "LEGO Blocks",
"creation_date": "10/18/2020",
"self": "https://<your-app>/items/123"
}
```

Failure

```
Status: 404 Not Found

{
"Error":  "No item with this item_id exists"
}
```

View all Items

List all the items.

GET /items

Request

Path Parameters

None

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	

Response Examples

- Must be paginated with 3 per page and a ‘next’ link.

Success

```
Status: 200 OK

{
“items”: [
        {
        "id": 123,
        "volume": 5,
        "carrier": {
                "id": 1234,
                "name": "Sea Witch",
                "self": "https://appspot.com/bags/1234"
                },
        "content": "LEGO Blocks",
        "creation_date": "10/18/2020"
        "self": "https://<your-app>/items/123"
        },
        {
        "id": 456,
        "volume": 10,
        "carrier": {
                "id": 789,
                "name": "John Almos",
                "self": "https://appspot.com/bags/789"
                },
        "content": "Hammers",
        "creation_date": "10/19/2020"
```

```
       "self": "https://<your-app>/items/456"
       }
],
“next”: “https://<your-app>/items?cursor=path_to_next_group”
}
```

Update an Item

Edit information about a bag.

PUT /items/:item_id
PATCH /items/:item_id

Request

Path Parameters

Name	Description
item_id	ID of the item

Request Body

Required

Request Body Format

JSON

Request JSON Attributes

Name	Description	Required?
name	The name of the bag.	Yes
type	The type of the bag. E.g., Sailbag, Catamaran, etc.	Yes
length	Length of the bag in feet.	Yes

Request Body Example

```
{
  "name": "Sea Witch",
  "type": "Catamaran",
  "length": 28
}
```

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	Succeeds only if the item_id exists
Failure	404 Not Found	The specified item does not exist

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found

{
"Error":  "The specified item does not exist"
}
```

Delete an Item

Allows you to delete an item. If the item being deleted was inside a bag, the bag must be updated.

```
DELETE /items/:item_id
```

Request

Path Parameters

Name	**Description**
item_id	ID of the item

Request Body

None

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	204 No Content	
Failure	404 Not Found	No item with this item_id exists

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 404 Not Found

{
"Error":  "No item with this item_id exists"
}
```

Assign an Item to a Bag

Place a item onto a bag.

```
PUT /bags/:item_id/:bag_id
```

Request

Path Parameters

Name	Description
item_id	ID of the item
bag_id	ID of the bag

Request Body

None

Note: Set Content-Length to 0 in your request when calling out to this endpoint.

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	Succeeds only if a bag exists with this bag_id, a item exists with this item_id, and the item isn't already on a bag.
Failure	403 Forbidden	The item was already assigned to the bag.
Failure	403 Forbidden	There is already a bag with this item.
Failure	404 Not Found	The specified bag and/or item does not exist

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 403 Forbidden

{
"Error":  "The item already assigned to this bag"
}
```

```
Status: 403 Forbidden

{
"Error":  "The item is still inside another bag"
}
```

```
Status: 404 Not Found

{
"Error":  "The specified bag and/or item does not exist"
}
```

Comment

- A item cannot be assigned to multiple bags. However, bags can have multiple items.

Remove an Item from a Bag

Removes a item from a bag without deleting that item.

```
DELETE /bags/:item_id/:bag_id
```

Request

Path Parameters

Name	Description
item_id	ID of the item
bag_id	ID of the bag

Request Body

None

Response

No body

Response Body Format

Success: No body

Failure: JSON

Response Statuses

Outcome	Status Code	Notes
Success	204 No Content	Succeeds only if a bag exists with this bag_id, a item exists with this item_id, and the bag carries that item.
Failure	403 Forbidden	bag_id has no item with the item_id
Failure	404 Not Found	The item and/or bag doesn't exist

Response Examples

Success

```
Status: 204 No Content
```

Failure

```
Status: 403 Not Found
{
"Error":  "No bag with this bag_id carries a item with this item_id"
}
```

Failure

```
Status: 404 Not Found
{
"Error":  "The specified item and/or bag does not exist"
}
```

View all items for a given Bag

Shows all items on a bag with specific bag_id

```
GET /bags/:bag_id/items
```

Request

Path Parameters

Name	Description
bag_id	ID of the bag

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	Succeeds if bag id exists
Failure	404 Not Found	No bag with this bag_id exists

Response Examples

Success

```
Status: 200 OK

{
    "items":[
      {
             "id": 5678,
             "self":https://appspot.com/items/5678
      },
      {
             "id": 8765,
             "self":https://appspot.com/items/8765
      }
    ]
}
```

Failure

```
Status: 404 Not Found

{
"Error":  "No bag with this bag_id exists"
}
```

View all Members

List all the bags.

```
GET /members
```

Request

Path Parameters

None

Request Body

None

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200 OK	

Response Examples

- Viewing all bags implements pagination.
- 3 bags should be returned per page.
- There must be a ‘next’ link to get to the next page of results, except for the last page.

Success

```
Status: 200 OK

{
“bags”: [
        {
        "id": 123,
        "name": "Sea Witch",
        "type": "Catamaran",
        "length": 28,
        "items":[
                {
                "id": 5678,
                "self":"https://appspot.com/items/5678"},
                {
                "id": 8765,
                "self":"https://appspot.com/items/8765"}
                ],
        "self": "https://<your-app>/bags/123"
        },
        {
        "id": 456,
        "name": "Adventure",
        "type": "Sailbag",
        "length": 50,
        "items":[
                {
                "id": 365,
                "self":"https://appspot.com/items/365"},
                {
```

```
              "id": 4821,
              "self":"https://appspot.com/items/4821"}
              ],
        "self": "https://<your-app>/bags/456"
       },
       {
       "id": 789,
       "name": "Hocus Pocus",
       "type": "Sailbag",
       "length": 100,
       "items":[
              {
              "id": 2624,
              "self":"https://appspot.com/items/2624"},
              {
              "id": 9087,
              "self":"https://appspot.com/items/9087"}
              ],
        "self": "https://<your-app>/bags/789"
              },
 ],
“next”: “https://<your-app>/bags?cursor=path_to_next_group”
}
```