Final Project: Eco-API

CS-493 Cloud Development

Austin Chayka

December 7, 2023

https://cs493-finalproject-406222.uw.r.appspot.com

Change Log

Version	Change	Date
1.0	Initial Version	Dec 7, 2023

Data Model

Family

Property	Data Type	Notes
id	String	Internal ID, generated by datastore with the prefix FAM
name	String	Name of the family
diagnostic features	String	Distinguishing features of the family
genera	ID String Array	List of the ids of genera that belong to this family
verified	Boolean	Marks whether this family has been verified
meta	JSON Object	Object has two attributes: the user id of the owner of this family, and the timestamp it was last edited
self	String	Link to the family in the api

Genus

Property	Data Type	Notes
id	String	Internal ID, generated by datastore with the prefix GEN
name	String	Name of the genus
diagnostic features	String	Distinguishing features of the genus
family	ID String	ID of the family this genus belongs to
species	ID String Array	List of the ids of species that belong to this genus
verified	Boolean	Marks whether this family has been verified
meta	JSON Object	Object has two attributes: the user id of the owner of this genus, and the timestamp it was last edited
self	String	Link to the genus in the api

Species

Property	Data Type	Notes
id	String	Internal ID, generated by datastore with the prefix SPC
name	String	Name of the species
diagnostic features	String	Distinguishing features of the species
common names	String Array	List of common names that refer to this species
species	ID String	Id of the species this observation belongs to
observations	ID String Array	List of the ids of observations that belong to this species
verified	Boolean	Marks whether this family has been verified
meta	JSON Object	Object has two attributes: the user id of the owner of this species, and the timestamp it was last edited
self	String	Link to the species in the api

Observation

Property	Data Type	Notes
id	String	Internal ID, generated by datastore with the prefix OBS
species	ID String	Id of the species this observation belongs to
verified	Boolean	Marks whether this family has been verified
location	JSON Object	Object has three attributes: country (string), state (string), and zip (integer)
image	String	Url to an image of the observation
public	Boolean	Marks whether this is publicly viewable
meta	JSON Object	Object has two attributes: the user id of the owner of this genus, and the timestamp it was last edited
self	String	Link to the family in the api

User

Property	Data Type	Notes
id	String	Internal ID, generated by datastore
username	String	User's username
user_id	String	User's id, this is the sub property of the jwt and is used for access control
created	Integer	Timestamp of first login

Family Operations

Add a Family

Creates a new family and adds it to the datastore.

POST /api/families

Request

Path Parameters

None.

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
name	String	Name of the family	Yes
diagnostic features	String	Distinguishing features of the family, up to 256 characters	No
verified	Boolean	Marks whether this family has been verified	No

Request Body Example

```
{
	“name”: “Agaricaceae”,
	“diagnostic features”: “...”,
	“verified”: false
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201	Family created
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type

Response Examples

Success:

```
Status: 201
{
   "id": "FAM5680690091786240",
   "name": "Agaricaceae",
   "diagnostic_features": "",
   "genera": [],
   "verified": false,
   "meta": {
      "owner": "auth0|655a8578c90bf30f6a1184b9",
      "time": 1702078234158
   },
   "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/families/FAM5680690091786240"
}
```

Failure:

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 400
{
   "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
   "Error": "Wrong content type"
}
```

Delete a Family

Deletes a family from the datastore. If the family had any assigned genera, the genus's family will be set to null. Because a family name is unique, it can be used in place of the id.

DELETE /api/families/:family_id

or

DELETE /api/families/:family_name

Request

Path Parameters

Name	Description
family_id	Id of the family
family_name	Unique name of the family

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Family deleted
Failure	404	Family with the specified id or name does not exist
Failure	401	Invalid or missing JWT
Failure	403	JWT does not match owner of family

Response Examples

Success:

```
Status: 204 No Content
```

Failure:

```
Status: 404
{
   "Error": "Not found"
}
```

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 403
{
   "Error": "Forbidden"
}
```

Get all Families

Lists data for all families, with a page size of 5. Next page will be linked via the next attribute.

GET /api/families(?verified=[true/false])

Request

Path Parameters

Name	Description
verified	Optional, if true will only return verified families

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	This request should never fail.

Response Examples

Success:

```
Status: 200
{
   "results": [
      {
         "diagnostic_features": "",
         "genera": [],
         "name": "Bulgariaceae",
         "verified": false,
         "id": "FAM4828657734385664",
         "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/families/FAM4828657734385664"
      },
      {
         "diagnostic_features": "",
         "genera": [],
         "name": "Cyttariaceae",
         "verified": false,
         "id": "FAM5131084064882688",
         "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/families/FAM5131084064882688"
      },
      {
         "diagnostic_features": "",
         "genera": [],
         "name": "Leotiaceae",
         "verified": false,
         "id": "FAM5188608642252800",
         "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/families/FAM5188608642252800"
      },
      {
         "diagnostic_features": "",
         "genera": [],
         "name": "Agaricaceae",
         "verified": false,
         "id": "FAM5680690091786240",
         "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/families/FAM5680690091786240"
      },
      {
```

```
            "diagnostic_features": "",
            "genera": [],
            "name": "Geoglossaceae",
            "verified": false,
            "id": "FAM5746523350499328",
            "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/families/FAM5746523350499328"
        }
    ],
    "next":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/families?cursor=CjoSNGodenV3fmNzN
DkzLWZpbmFscHJvamVjdC00MDYyMjJyEwsSBkZBTUlMWRiAgIDY682aCgwYACAA"
}
```

Get a Family

Shows data for a specified family. Because a family name is unique, it can be used in place of the id.

GET /api/families/:family_id

or

GET /api/families/:family_name

Request

Path Parameters

Name	Description
family_id	Id of the family
family_name	Unique name of the family

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	Family found
Failure	404	Family with the specified id or name does not exist
Failure	406	Wrong accept type

Response Examples

Success:

```
Status: 200
{
   "diagnostic_features": "",
   "genera": [],
   "meta": {
      "owner": "auth0|655a8578c90bf30f6a1184b9",
      "time": 1702078234158
   },
   "name": "Agaricaceae",
   "verified": false,
   "id": "FAM5680690091786240",
   "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/families/FAM5680690091786240"
}
```

Failure:

```
Status: 404
{
   "Error": "Not found"
}
```

```
Status: 406
{
   "Error": "Not acceptable"
}
```

Replace a Family

Replaces a family in the datastore.

PUT /api/families/:family_id

or

PUT /api/families/:family_name

Request

Path Parameters

Name	Description
family_id	Id of the family
family_name	Unique name of the family

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
name	String	Name of the family	Yes
diagnostic features	String	Distinguishing features of the family, up to 256 characters	No
verified	Boolean	Marks whether this family has been verified	No

Request Body Example

```
{
        “name”: “Agaricaceae”,
        “diagnostic features”: “...”,
        “verified”: false
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Family replaced
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type
Failure	403	JWT does not match owner of family

Response Examples

Success:

```
Status: 204 no content
```

Failure:

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 400
{
   "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
   "Error": "Wrong content type"
}
```

Edit a Family

Edits a family in the datastore.

PATCH /api/families/:family_id

or

PATCH /api/families/:family_name

Request

Path Parameters

Name	Description
family_id	Id of the family
family_name	Unique name of the family

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
name	String	Name of the family	No
diagnostic features	String	Distinguishing features of the family, up to 256 characters	No
verified	Boolean	Marks whether this family has been verified	No

Request Body Example

```
{
        “name”: “Agaricaceae”,
        “diagnostic features”: “...”,
        “verified”: false
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Family edited
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type

Response Examples

Success:

```
Status: 204 no content
```

Failure:

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 400
{
   "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
   "Error": "Wrong content type"
}
```

```
Status: 403
{
   "Error": "Forbidden"
}
```

Add new Genus to Family

Creates a new family and adds it to the datastore.

POST /api/families/:family_id/genera

or

POST /api/families/:family_name/genera

Request

Path Parameters

Name	Description
family_id	Id of the family
family_name	Unique name of the family

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
name	String	Name of the genus	Yes
diagnostic features	String	Distinguishing features of the genus, up to 256 characters	No
verified	Boolean	Marks whether this genus has been verified	No

Request Body Example

```
{
	“name”: “Agaricus”,
	“diagnostic features”: “...”,
	“verified”: false
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201	Genus created
Failure	404	Family not found
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type

Response Examples

Success:

```
Status: 201
{
   "id": "GEN5660308458700800",
   "name": "Agaricus",
   "diagnostic_features": "",
   "family": "FAM5183573397078016",
   "species": [],
   "verified": false,
   "meta": {
      "owner": "auth0|655a8578c90bf30f6a1184b9",
      "time": 1702078239759
   },
   "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/genera/GEN5660308458700800"
}
```

Failure:

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 404
{
   "Error": "Not found"
}
```

```
Status: 400
{
   "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
   "Error": "Wrong content type"
}
```

Get All Genera in Family

Lists data for all genera in family, with a page size of 5. Next page will be linked via the next attribute.

GET /api/families/:family_id/genera(?verified=[true/false])

or

GET /api/families/:family_name/genera(?verified=[true/false])

Request

Path Parameters

Name	Description
family_id	Id of the family
family_name	Unique name of the family
verified	Optional, if true will only return verified genera

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	This request should never fail.

Response Examples

Success:

```
Status: 200
{
    "results": [
        {
            "diagnostic_features": "",
            "species": [],
            "name": "Agaricus",
            "verified": false,
            "id": "GEN5660308458700800",
            "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/genera/GEN5660308458700800"
        }
    ]
}
```

Add existing Genus to Family

Adds a genus to the list of genera of a family. Name and id can be used interchangeably for both family and genera.

PUT /api/families/:family_id/genera/:genus_id

or

PUT /api/families/:family_name/genera/:genus_name

Request

Path Parameters

Name	Description
family_id	Id of the family
genera_id	Id of the genus
family_name	Unique name of the family
genera_name	Unique name of the genus

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Genus added to family
Failure	404	Family or genus not found
Failure	400	Genus already assigned to a family
Failure	403	JWT does not match owner of family or genus

Failure	401	Missing or invalid JWT

Response Examples

Success:

```
Status: 204 No Content
```

Failure:

```
Status: 404
{
   "Error": "Not found"
}
```

```
Status: 403
{
   "Error": "Forbidden"
}
```

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

Remove Genus from Family

Removes a genus to the list of genera of a family. Name and id can be used interchangeably for both family and genera.

DELETE /api/families/:family_id/genera/:genus_id

or

DELETE /api/families/:family_name/genera/:genus_name

Request

Path Parameters

Name	Description
family_id	Id of the family
genera_id	Id of the genus
family_name	Unique name of the family
genera_name	Unique name of the genus

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Genus removed from family
Failure	404	Family or genus not found
Failure	400	Genus already assigned to a family
Failure	403	JWT does not match owner of family or genus

Failure	401	Missing or invalid JWT

Response Examples

Success:

```
Status: 204 No Content
```

Failure:

```
Status: 404
{
   "Error": "Not found"
}
```

```
Status: 403
{
   "Error": "Forbidden"
}
```

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

Genus Operations

Add a Genus

Creates a new genus and adds it to the datastore.

POST /api/genera

Request

Path Parameters

None.

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
name	String	Name of the genus	Yes
diagnostic features	String	Distinguishing features of the genus, up to 256 characters	No
verified	Boolean	Marks whether this genus has been verified	No

Request Body Example

```
{
	“name”: “Agaricus”,
	“diagnostic features”: “...”,
	“verified”: false
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201	Genus created
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type

Response Examples

Success:

```
Status: 201
{
   "id": "GEN5660308458700800",
   "name": "Agaricus",
   "diagnostic_features": "",
   "family": null,
   "species": [],
   "verified": false,
   "meta": {
      "owner": "auth0|655a8578c90bf30f6a1184b9",
      "time": 1702078239759
   },
   "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/genera/GEN5660308458700800"
}
```

Failure:

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 400
{
   "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
   "Error": "Wrong content type"
}
```

Delete a Genus

Deletes a genus from the datastore. If the genus has any assigned species, the species's genus will be set to null. Because a genus name is unique, it can be used in place of the id.

DELETE /api/genera/:genus_id

or

DELETE /api/genera/:genus_name

Request

Path Parameters

Name	Description
genus_id	Id of the genus
genus_name	Unique name of the genus

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Genus deleted
Failure	404	Genus with the specified id or name does not exist
Failure	401	Invalid or missing JWT
Failure	403	JWT does not match owner of family

Response Examples

Success:

```
Status: 204 No Content
```

Failure:

```
Status: 404
{
  "Error": "Not found"
}
```

```
Status: 401
{
  "Error": "Invalid or missing JWT"
}
```

```
Status: 403
{
  "Error": "Forbidden"
}
```

Get all Genera

Lists data for all genera, with a page size of 5. Next page will be linked via the next attribute.

GET /api/genera(?verified=[true/false])

Request

Path Parameters

Name	Description
verified	Optional, if true will only return verified genera

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	This request should never fail.

Response Examples

Success:

```
Status: 200
{
    "results": [
        {
            "diagnostic_features": "",
            "species": [],
            "name": "Agaricus",
            "verified": false,
            "id": "GEN5081834211770368",
            "family": null,
            "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/genera/GEN5081834211770368"
        }
    ]
}
```

Get a Genus

Shows data for a specified genus. Because a genus name is unique, it can be used in place of the id.

GET /api/genera/:genus_id

or

GET /api/genera/:genus_name

Request

Path Parameters

Name	Description
genus_id	Id of the genus
genus_name	Unique name of the genus

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	Genus found
Failure	404	Genus with the specified id or name does not exist
Failure	406	Wrong accept type

Response Examples

Success:

```
Status: 200
{
   "diagnostic_features": "",
   "species": [],
   "meta": {
      "owner": "auth0|655a8578c90bf30f6a1184b9",
      "time": 1702078238833
   },
   "name": "Mushroom",
   "verified": false,
   "id": "GEN5081834211770368",
   "family": null,
   "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/genera/GEN5081834211770368"
}
```

Failure:

```
Status: 404
{
   "Error": "Not found"
}
```

```
Status: 406
{
   "Error": "Not acceptable"
}
```

Replace a Genus

Replaces a genus in the datastore.

PUT /api/genera/:genus_id

or

PUT /api/genera/:genus_name

Request

Path Parameters

Name	Description
genus_id	Id of the genus
genus_name	Unique name of the genus

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
name	String	Name of the genus	Yes
diagnostic features	String	Distinguishing features of the genus, up to 256 characters	No
verified	Boolean	Marks whether this genus has been verified	No

Request Body Example

```
{
	“name”: “Agaricus”,
	“diagnostic features”: “...”,
	“verified”: false
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Genus replaced
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type
Failure	403	JWT does not match owner of family

Response Examples

Success:

```
Status: 204 no content
```

Failure:

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 400
{
   "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
   "Error": "Wrong content type"
}
```

Edit a Genus

Edits a genus in the datastore.

PATCH /api/genera/:genus_id

or

PATCH /api/genera/:genus_name

Request

Path Parameters

Name	Description
genus_id	Id of the genus
genus_name	Unique name of the genus

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
name	String	Name of the genus	No
diagnostic features	String	Distinguishing features of the genus, up to 256 characters	No
verified	Boolean	Marks whether this family has been verified	No

Request Body Example

```
{
	“name”: “Agaricus”,
	“diagnostic features”: “...”,
	“verified”: false
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Genus edited
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type

Response Examples

Success:

```
Status: 204 no content
```

Failure:

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 400
{
   "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
   "Error": "Wrong content type"
}
```

```
Status: 403
{
   "Error": "Forbidden"
}
```

Add new Species to Genus

Creates a new species and adds it to the datastore.

POST /api/genera/:genus_id/species

or

POST /api/genera/:genus_name/species

Request

Path Parameters

Name	Description
genus_id	Id of the genus
genus_name	Unique name of the genus

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
name	String	Name of the species	Yes
diagnostic features	String	Distinguishing features of the species, up to 256 characters	No
verified	Boolean	Marks whether this species has been verified	No
Common names	String Array	List of common names of the species	No

Request Body Example

```
{
        “name”: “bisporus”,
        “diagnostic features”: “...”,
        “verified”: false,
        “Common_names”: [“Bella”, “Portobello”]
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	201	Species created
Failure	404	Genus not found
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type

Response Examples

Success:

```
Status: 201
{
   "id": "SPC5180720062398464",
   "name": "bisporus",
   "diagnostic_features": "",
   "genus": "GEN5660308458700800",
   "observations": [],
   "common_names": [“Bella”, “Portobello”],
   "verified": false,
   "meta": {
      "owner": "auth0|655a8578c90bf30f6a1184b9",
      "time": 1702078241418
```

```
  },
  "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/species/SPC5180720062398464"
}
```

Failure:

```
Status: 401
{
  "Error": "Invalid or missing JWT"
}
```

```
Status: 404
{
  "Error": "Not found"
}
```

```
Status: 400
{
  "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
  "Error": "Wrong content type"
}
```

Get All Species in Genus

Lists data for all species in the genus, with a page size of 5. Next page will be linked via the next attribute.

GET /api/genera/:genus_id/species(?verified=[true/false])

or

GET /api/genera/:genus_name/species(?verified=[true/false])

Request

Path Parameters

Name	Description
genus_id	Id of the genus
genus_name	Unique name of the genus
verified	Optional, if true will only return verified genera

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	This request should never fail.

Response Examples

Success:

```
Status: 200
{
    "results": [
        {
            "diagnostic_features": "",
            "observations": [],
            "name": "bisporus",
            "verified": false,
            "common_names": [],
            "id": "SPC5180720062398464",
            "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/species/SPC5180720062398464"
        }
    ]
}
```

Add existing Species to Genus

Adds a species to the list of species of a genus. Name and id can be used interchangeably for both genus and species.

PUT /api/genera/:genus_id/species/:species_id

or

PUT /api/genera/:genus_name/species/:species_name

Request

Path Parameters

Name	Description
genus_id	Id of the genus
species_id	Id of the species
genus_name	Unique name of the genus
species_name	Unique name of the species

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Species added to genus
Failure	404	Species or genus not found
Failure	400	Species already assigned to a genus
Failure	403	JWT does not match owner of species or genus

Failure	401	Missing or invalid JWT

Response Examples

Success:

```
Status: 204 No Content
```

Failure:

```
Status: 404
{
   "Error": "Not found"
}
```

```
Status: 403
{
   "Error": "Forbidden"
}
```

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

Remove Species from Genus

Removes a species to the list of species of a genera. Name and id can be used interchangeably for both species and genera.

DELETE /api/genera/:genus_id/species/:species_id

or

DELETE /api/genera/:genus_name/species/:species_name

Request

Path Parameters

Name	Description
genus_id	Id of the genus
species_id	Id of the species
genus_name	Unique name of the genus
species_name	Unique name of the species

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Species removed from genus
Failure	404	Species or genus not found
Failure	400	Species already assigned to a genus
Failure	403	JWT does not match owner of species or genus

Failure	401	Missing or invalid JWT

Response Examples

Success:

```
Status: 204 No Content
```

Failure:

```
Status: 404
{
   "Error": "Not found"
}
```

```
Status: 403
{
   "Error": "Forbidden"
}
```

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

Species Operations

Add a Species

Creates a new species and adds it to the datastore.

POST /api/species

Request

Path Parameters

None.

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
name	String	Name of the species	Yes
diagnostic features	String	Distinguishing features of the species, up to 256 characters	No
verified	Boolean	Marks whether this species has been verified	No
Common names	String Array	List of common names of the species	No

Request Body Example

```
{
        “name”: “bisporus”,
        “diagnostic features”: “...”,
        “verified”: false,
        “Common_names”: [“Bella”, “Portobello”]
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	201	Species created
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type

Response Examples

Success:

```
Status: 201
{
   "id": "SPC5180720062398464",
   "name": "bisporus",
   "diagnostic_features": "",
   "genus": null,
   "observations": [],
   "common_names": [],
   "verified": false,
   "meta": {
      "owner": "auth0|655a8578c90bf30f6a1184b9",
      "time": 1702078241418
   },
```

```
   "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/species/SPC5180720062398464"
}
```

Failure:

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 400
{
   "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
   "Error": "Wrong content type"
}
```

Delete a Species

Deletes a species from the datastore. If the species has any assigned observations, the observation's species will be set to null. Because a species name is unique, it can be used in place of the id.

DELETE /api/species/:species_id

or

DELETE /api/species/:species_name

Request

Path Parameters

Name	Description
species_id	Id of the species
species_name	Unique name of the species

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Species deleted
Failure	404	Species with the specified id or name does not exist
Failure	401	Invalid or missing JWT
Failure	403	JWT does not match owner of family

Response Examples

Success:

```
Status: 204 No Content
```

Failure:

```
Status: 404
{
  "Error": "Not found"
}
```

```
Status: 401
{
  "Error": "Invalid or missing JWT"
}
```

```
Status: 403
{
  "Error": "Forbidden"
}
```

Get all Species

Lists data for all species, with a page size of 5. Next page will be linked via the next attribute.

GET /api/species(?verified=[true/false])

Request

Path Parameters

Name	Description
verified	Optional, if true will only return verified species

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	This request should never fail.

Response Examples

Success:

```
Status: 200
{
   "results": [
      {
         "diagnostic_features": "",
         "observations": [],
         "name": "bisporus",
         "verified": false,
         "common_names": [],
         "id": "SPC5180720062398464",
         "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/species/SPC5180720062398464"
      }
   ]
}
```

Get a Species

Shows data for a specified species. Because a species name is unique, it can be used in place of the id.

GET /api/species/:species_id

or

GET /api/species/:species_name

Request

Path Parameters

Name	Description
species_id	Id of the species
species_name	Unique name of the species

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	Species found
Failure	404	Species with the specified id or name does not exist
Failure	406	Wrong accept type

Response Examples

Success:

```
Status: 200
{
   "diagnostic_features": "",
   "genus": null,
   "meta": {
      "owner": "auth0|655a8578c90bf30f6a1184b9",
      "time": 1702078240553
   },
   "observations": [],
   "name": "bisporus",
   "verified": false,
   "common_names": [],
   "id": "SPC4862202200719360",
   "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/species/SPC4862202200719360"
}
```

Failure:

```
Status: 404
{
   "Error": "Not found"
}
```

```
Status: 406
{
   "Error": "Not acceptable"
}
```

Replace a Species

Replaces a species in the datastore.

PUT /api/species/:species_id

or

PUT /api/species/:species_name

Request

Path Parameters

Name	Description
species_id	Id of the species
species_name	Unique name of the species

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
name	String	Name of the species	Yes
diagnostic features	String	Distinguishing features of the species, up to 256 characters	No
verified	Boolean	Marks whether this species has been verified	No
Common names	String Array	List of common names of the species	No

Request Body Example

```
{
	“name”: “bisporus”,
	“diagnostic features”: “...”,
	“verified”: false,
	“Common_names”: [“Bella”, “Portobello”]
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	204	Species replaced
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type
Failure	403	JWT does not match owner of family

Response Examples

Success:

```
Status: 204 no content
```

Failure:

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 400
{
   "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
   "Error": "Wrong content type"
}
```

Edit a Species

Edits a species in the datastore.

PATCH /api/species/:species_id

or

PATCH /api/species/:species_name

Request

Path Parameters

Name	Description
species_id	Id of the species
species_name	Unique name of the species

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
name	String	Name of the species	No
diagnostic features	String	Distinguishing features of the species, up to 256 characters	No
verified	Boolean	Marks whether this species has been verified	No
Common names	String Array	List of common names of the species	No

Request Body Example

```
{
	“name”: “bisporus”,
	“diagnostic features”: “...”,
	“verified”: false,
	“Common_names”: [“Bella”, “Portobello”]
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	204	Species edited
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type

Response Examples

Success:

```
Status: 204 no content
```

Failure:

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 400
{
   "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
   "Error": "Wrong content type"
}
```

```
Status: 403
{
   "Error": "Forbidden"
}
```

Add new Observation to Species

Creates a new observation and adds it to the datastore.

POST /api/species/:species_id/observations

or

POST /api/species/:species_name/observations

Request

Path Parameters

Name	Description
species_id	Id of the species
species_name	Unique name of the species

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
location	JSON Object	Object has three attributes: country (string), state (string), and zip (integer)	yes
image	String	Url to an image of the observation	no
public	Boolean	Marks whether this is publicly viewable	no
verified	Boolean	Marks whether this species has been verified	No

Request Body Example

```
{
   "location": {
      "country": "USA",
      "state": "Oregon",
      "zip": 97330
   }
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201	Species created
Failure	404	Genus not found
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type

Response Examples

Success:

```
Status: 201
{
   "id": "OBS6018216472084480",
   "species": null,
   "verified": false,
   "location": {
      "country": "USA",
      "state": "Oregon",
      "zip": 97330
   },
   "image": null,
```

```
  "meta": {
    "owner": "auth0|655a8578c90bf30f6a1184b9",
    "time": 1702078243131,
    "public": true
  },
  "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/observations/OBS6018216472084480"
}
```


Failure:


```
Status: 401
{
  "Error": "Invalid or missing JWT"
}
```

```
Status: 404
{
  "Error": "Not found"
}
```

```
Status: 400
{
  "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
  "Error": "Wrong content type"
}
```

Get All Observations in Species

Lists data for all observations in the species, with a page size of 5. Next page will be linked via the next attribute.

GET /api/species/:species_id/observations(?verified=[true/false])

or

GET /api/species/:species_name/observations(?verified=[true/false])

Request

Path Parameters

Name	Description
species_id	Id of the species
species_name	Unique name of the species
verified	Optional, if true will only return verified species

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	This request should never fail.

Response Examples

Success:

```
Status: 200
{
    "results": [
        {
            "image": null,
            "verified": false,
            "location": {
                "zip": 97330,
                "country": "USA",
                "state": "Oregon"
            },
            "id": "OBS6018216472084480",
            "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/observations/OBS6018216472084480"
        }
    ]
}
```

Add existing Observation to Species

Adds an observation to the list of species of a genus. Name and id can be used interchangeably for both genus and species.

PUT /api/species/:species_id/observations/:observations_id

or

PUT /api/species/:species_name/observations/:observations_id

Request

Path Parameters

Name	Description
observation_id	Id of the observation
species_id	Id of the species
species_name	Unique name of the species

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Observation added to genus
Failure	404	Species or observation not found
Failure	400	Observation already assigned to a species
Failure	403	JWT does not match owner of species or observation

Failure	401	Missing or invalid JWT

Response Examples

Success:

```
Status: 204 No Content
```

Failure:

```
Status: 404
{
   "Error": "Not found"
}
```

```
Status: 403
{
   "Error": "Forbidden"
}
```

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

Remove Observation to Species

Removes an observation to the list of species of a genus. Name and id can be used interchangeably for both genus and species.

DELETE /api/species/:species_id/observations/:observations_id

or

DELETE /api/species/:species_name/observations/:observations_id

Request

Path Parameters

Name	Description
observation_id	Id of the observation
species_id	Id of the species
species_name	Unique name of the species

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Observation removed from genus
Failure	404	Species or observation not found
Failure	400	Observation already assigned to a species
Failure	403	JWT does not match owner of species or observation

Failure	401	Missing or invalid JWT

Response Examples

Success:

```
Status: 204 No Content
```

Failure:

```
Status: 404
{
   "Error": "Not found"
}
```

```
Status: 403
{
   "Error": "Forbidden"
}
```

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

Observation Operations

Add a Observation

Creates a new species and adds it to the datastore.

POST /api/observations

Request

Path Parameters

None.

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
location	JSON Object	Object has three attributes: country (string), state (string), and zip (integer)	yes
image	String	Url to an image of the observation	no
public	Boolean	Marks whether this is publicly viewable	no
verified	Boolean	Marks whether this species has been verified	No

Request Body Example

```
{
   "location": {
      "country": "USA",
      "state": "Oregon",
      "zip": 97330
   }
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	201	Observation created
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type

Response Examples

Success:

```
Status: 201
{
   "id": "OBS5686696536440832",
   "species": null,
   "verified": false,
   "location": {
      "country": "USA",
      "state": "Oregon",
      "zip": 97330
   },
   "image": null,
   "meta": {
      "owner": "auth0|655a8578c90bf30f6a1184b9",
```

```
        "time": 1702267299817,
        "public": true
    },
    "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/observations/OBS5686696536440832"
}
```


Failure:


```
Status: 401
{
    "Error": "Invalid or missing JWT"
}
```

```
Status: 400
{
    "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
    "Error": "Wrong content type"
}
```

Delete an Observation

Deletes an observation from the datastore.

DELETE /api/observation/:observation_id

Request

Path Parameters

Name	**Description**
species_id	Id of the observation

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	**Status Code**	**Notes**
Success	204	Observation deleted
Failure	404	Observation with the specified id does not exist
Failure	401	Invalid or missing JWT
Failure	403	JWT does not match owner of family

Response Examples

Success:

Status: 204 No Content

Failure:

```
Status: 404
{
   "Error": "Not found"
}
```

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 403
{
   "Error": "Forbidden"
}
```

Get all Observations

Lists data for all observations, with a page size of 5. Next page will be linked via the next attribute.

GET /api/observations(?verified=[true/false])

Request

Path Parameters

Name	Description
verified	Optional, if true will only return verified observations

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	This request should never fail.

Response Examples

Success:

```
Status: 200
{
    "results": [
        {
            "image": null,
            "species": null,
            "verified": false,
            "location": {
                "zip": 97330,
                "country": "USA",
                "state": "Oregon"
            },
            "id": "OBS5686696536440832",
            "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/observations/OBS5686696536440832"
        }
    ]
}
```

Get an Observation

Shows data for a specific observation.

GET /api/observation/observation_id

Request

Path Parameters

Name	Description
observation_id	Id of the observation

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	Observation found
Failure	404	Observation with the specified id or name does not exist
Failure	406	Wrong accept type

Response Examples

Success:

```
Status: 200
{
   "image": null,
   "species": null,
   "meta": {
      "owner": "auth0|655a8578c90bf30f6a1184b9",
      "public": true,
      "time": 1702267299817
   },
   "verified": false,
   "location": {
      "zip": 97330,
      "country": "USA",
      "state": "Oregon"
   },
   "id": "OBS5686696536440832",
   "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/observations/OBS5686696536440832"
}
```

Failure:

```
Status: 404
{
   "Error": "Not found"
}
```

```
Status: 406
{
   "Error": "Not acceptable"
}
```

Replace an Observation

Replaces a species in the datastore.

PUT /api/observations/:observation_id

Request

Path Parameters

Name	Description
observation_id	Id of the observation

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
location	JSON Object	Object has three attributes: country (string), state (string), and zip (integer)	yes
image	String	Url to an image of the observation	no
public	Boolean	Marks whether this is publicly viewable	no
verified	Boolean	Marks whether this species has been verified	No

Request Body Example

```
{
   "location": {
      "country": "USA",
      "state": "Oregon",
      "zip": 97330
   }
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Observation replaced
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type
Failure	403	JWT does not match owner of family

Response Examples

Success:

```
Status: 204 no content
```

Failure:

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 400
{
   "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
   "Error": "Wrong content type"
}
```

Edit an Observation

Replaces a species in the datastore.

PATCH /api/observations/:observation_id

Request

Path Parameters

Name	Description
observation_id	Id of the observation

Request Body

Required.

Request Body Format

JSON

Request JSON Attributes

Property	Data Type	Notes	Required
location	JSON Object	Object has three attributes: country (string), state (string), and zip (integer)	no
image	String	Url to an image of the observation	no
public	Boolean	Marks whether this is publicly viewable	no
verified	Boolean	Marks whether this species has been verified	No

Request Body Example

```
{
   "location": {
      "country": "USA",
      "state": "Oregon",
      "zip": 97330
   }
}
```

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	204	Observation edited
Failure	401	Missing or invalid JWT
Failure	400	Attributes violate constraint
Failure	415	Wrong content type
Failure	403	JWT does not match owner of family

Response Examples

Success:

```
Status: 204 no content
```

Failure:

```
Status: 401
{
   "Error": "Invalid or missing JWT"
}
```

```
Status: 400
{
   "Error": "Request attribute '...' violates constraint: …"
}
```

```
Status: 415
{
   "Error": "Wrong content type"
}
```

User Operations

Get all Users

Lists data for all users, with a page size of 5. Next page will be linked via the next attribute.

GET /api/users

Request

Path Parameters

None.

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	This request should never fail.

Response Examples

Success:

```
Status: 200
{
   "results": [
      {
         "user_id": "auth0|655c3ceb17b4bdb5010dd4c9",
         "created": 1702074106626,
         "username": "testuser",
         "id": "4892316565241856"
      },
      {
         "user_id": "auth0|655a8578c90bf30f6a1184b9",
         "created": 1702267259214,
         "username": "bigcheese",
         "id": "5676228493180928"
      }
   ]
}
```

Get all of a Users Families

Lists data for all of a user's families, with a page size of 5. Next page will be linked via the next attribute. "Mine" can be used in place of a user id to get families for the owner of the current jwt.

GET /api/users/:user_id/families

or

GET /api/users/mine/families

Request

Path Parameters

Name	Description
user_id	Id of the user, corresponds to the "sub" property of the jwt.

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	This request should never fail. If the user does not exist, it returns an empty list.

Response Examples

Success:

```
Status: 200
{
    "results": [
        {
            "diagnostic_features": "",
            "name": "Agaricaceae",
            "verified": false,
            "id": "FAM5629978607616000",
            "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/families/FAM5629978607616000"
        }
    ]
}
```

Get all of a Users Genera

Lists data for all of a user's genera, with a page size of 5. Next page will be linked via the next attribute. "Mine" can be used in place of a user id to get genera for the owner of the current jwt.

GET /api/users/:user_id/genera

or

GET /api/users/mine/genera

Request

Path Parameters

Name	Description
user_id	Id of the user, corresponds to the "sub" property of the jwt.

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	This request should never fail. If the user does not exist, it returns an empty list.

Response Examples

Success:

```
Status: 200
{
    "results": [
        {
            "diagnostic_features": "",
            "name": "Agaricus",
            "verified": false,
            "id": "GEN6239178446602240",
            "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/genera/GEN6239178446602240"
        }
    ]
}
```

Get all of a Users Species

Lists data for all of a user's species, with a page size of 5. Next page will be linked via the next attribute. "Mine" can be used in place of a user id to get species for the owner of the current jwt.

GET /api/users/:user_id/species

or

GET /api/users/mine/species

Request

Path Parameters

Name	Description
user_id	Id of the user, corresponds to the "sub" property of the jwt.

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	This request should never fail. If the user does not exist, it returns an empty list.

Response Examples

Success:

```
Status: 200
{
    "results": [
        {
            "diagnostic_features": "",
            "name": "bisporus",
            "verified": false,
            "common_names": [],
            "id": "SPC5707975213711360",
            "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/species/SPC5707975213711360"
        }
    ]
}
```

Get all of a Users Observations

Lists data for all of a user's observations, with a page size of 5. Next page will be linked via the next attribute. "Mine" can be used in place of a user id to get observations for the owner of the current jwt. Will only be able to see public observations or and ones that you are the owner of.

GET /api/users/:user_id/observations

or

GET /api/users/mine/observations

Request

Path Parameters

Name	Description
user_id	Id of the user, corresponds to the "sub" property of the jwt.

Request Body

None.

Response

Response Body Format

JSON

Response Statuses

Outcome	Status Code	Notes
Success	200	This request should never fail. If the user does not exist, it returns an empty list.

Response Examples

Success:

```
Status: 200
{
    "results": [
        {
            "image": null,
            "verified": false,
            "location": {
                "zip": 97330,
                "country": "USA",
                "state": "Oregon"
            },
            "id": "OBS5145025260290048",
            "self":
"https://cs493-finalproject-406222.uw.r.appspot.com/api/observations/OBS5145025260290048"
        }
    ]
}
```