Citizen Science App API Documentation

Created by Lindsay Schwartz

Base URL

`https://capstone-deploy-production.up.railway.app`

All requests MUST include credentials for login routes and sessions:

Example:

```
fetch('https://capstone-deploy-production.up.railway.app/add-project',
{
  method: 'POST',
  headers: { 'Content-Type': 'application/json', },
  credentials: 'include',
  body: JSON.stringify({
    project_code: 'P8989',
    title: 'Test Project',
    description: 'This is a test project',
    instructions: 'Follow these steps',
  }),
}) .then((response) => { if (!response.ok …
```

Users

Register New User

End point: /auth/register
Method: POST
Description: Registers a new user by accepting necessary details and creating an account.

Request Example (JSON)

```
{
  "username": "testuser",   **must be unique
```

```
  "password": "testpassword",
  "email": "testuser@example.com",
  "first_name": "Test",
  "last_name": "User",
  "school": "Test School"
}
```

Response

```
201 Created:
{"success": true, "message": "Registration successful"}
```

Login

End point: /auth/login
Method: POST
Description: Logs in an existing user by verifying username and password.

Request Example (JSON)

```
{
  "username": "newuser",
  "password": "newpassword"
}
```

Response

```
200 OK:
{"success": true, "message": "Login successful"}
```

Logout

End point: /auth/logout
Method: POST
Auth Required: Yes
Description: Logs out the currently authenticated user.

Response

```
201 OK:
{"success": true}
```

Get User Information

End point: /auth/user/<user_id>
Method: GET
Auth Required: Yes
Description: Retrieves information about a specific user by user ID.

Reponse

```
200 OK:
{
  "user_id": 1,
  "username": "newuser",
  "email": "newuser@example.com",
  "first_name": "New",
  "last_name": "User",
  "school": "Test School"
}
```

Update User Information

End point: /auth/update-user/<user_id>
Method: PUT
Auth Required: Yes
Description: Updates the details of a specific user.

Request Example (JSON)

```
{
  "email": "updateduser@example.com",
  "first_name": "Updated",
  "last_name": "User",
  "school": "Updated School"
}
```

Response

```
200 OK:
{"success": true, "message": "User updated"}
```

Delete User

End point: /auth/delete-user/<user_id>
Method: DELETE
Auth Required: Yes
Description: Deletes a user account based on the user ID.

Response

```
201 OK:
{"success": true, "message": "User deleted"}
```

Projects

View All User Projects

End point: /user-projects
Method: GET
Auth Required: Yes
Description: Retrieves all projects associated with the currently logged-in user.

Response

```
201 OK with a list of projects:
[
  {
    "project_id": 1,
    "created_at": "2023-11-13T00:00:00",
    "project_code": "TEST123",
    "title": "Test Project",
    "description": "Test Description",
    "instructions": "Test Instructions",
    "created_by": 7
```

```
  }
]
```

Add New Project

End point: /add-project
Method: POST
Auth Required: Yes
Description: Creates a new project associated with the currently logged-in user.

Request Example (JSON)

```
{
  "project_code": "TEST123", **must be unique
  "title": "Test Project",
  "description": "Test Description",
  "instructions": "Test Instructions"
}
```

Response

```
201 Created:
{
  "success": true,
  "project": {
    "project_id": 1,
    "created_at": "2023-11-13T00:00:00",
    "project_code": "TEST123",
    "title": "Test Project",
    "description": "Test Description",
    "instructions": "Test Instructions",
    "created_by": 7
  }
}
```

Get Project Details

End point: /project/<project_id>
Method: GET
Auth Required: No
Description: Retrieves details for a specific project.

Response

201 OK with project details:

```
{
  "success": true,
  "project": {
    "project_id": 1,
    "created_at": "2023-11-13T00:00:00",
    "project_code": "TEST123",
    "title": "Test Project",
    "description": "Test Description",
    "instructions": "Test Instructions",
    "created_by": 7
  }
}
```

Update Project

End point: /update-project/<project_id>
Method: PUT
Auth Required: Yes
Description: Updates the specified project's details. The project must belong to the logged-in user.

Request Example (JSON)

```
{
  "project_code": "TEST123", **must be unique
  "title": "Updated Title",
  "description": "Updated Description",
  "instructions": "Updated Instructions"
}
```

Response

```
200 OK:
{"success": true, "message": "Project updated"}
```

Delete Project

End point: /delete-project/<project_id>
Method: DELETE
Auth Required: Yes
Description: Deletes the specified project if it belongs to the logged-in user.

Response

```
201 OK:
{"success": true, "message": "Project deleted"}
```

Forms

Show All Forms for Current User

End point: /forms
Method: GET
Auth Required: Yes
Description: Retrieves all form templates associated with projects owned by the current user.

Response

```
201 OK: Returns a list of form templates with associated fields.
[
  {
      "form_id": 2,
      "created_at": "2023-11-15T13:45:30",
      "description": "Sample Form for Project",
      "created_by": 89,
      "fields": [
        {
```

```
            "field_id": 1,
            "field_title": "Observation Notes",
            "field_type": "text",
            "field_description": "Enter notes about the observation",
            "field_options": null,
            "field_order": 1,
            "is_required": true,
            "form": project_id
        }
      ]
   }
]
```

Add New Form Template

End point: /add-form
Method: POST
Auth Required: Yes
Description: Creates a new form template for a specified project with associated fields.

Request Example (JSON)

```
{
  "project_id": 6,
  "description": "Sample Form",
  "fields": [
      {
        "field_title": "Observation Notes",
        "field_type": "text",
        "field_description": "Enter notes about the observation",
        "field_options": null,
        "field_order": 1,
        "is_required": true
      },
      {
        "field_title": "Observation Type",
        "field_type": "dropdown",
        "field_description": "Select the type of observation",
```

```
      "field_options": ["Temp", "Humidity", "Wind Speed"],
      "field_order": 2,
      "is_required": true
    }
  ]
}
```

Response

201 Created: Returns the newly created form template with associated fields.

```
  {
      "success": true,
      "message": "Form and fields added",
      "form": {
        "form_id": 1,
        "created_at": "2023-11-15T13:45:30",
        "description": "Sample Form",
        "created_by": 89,
        "project_id": 6,
        "fields": [
           {
             "field_id": 1,
             "field_title": "Observation Notes",
             "field_type": "text",
             "field_description": "Enter notes about the observation",
             "field_options": null,
             "field_order": 1,
             "is_required": true
           },
           {
             “field_id”: 2,
              . . .   # continues . . .
           }
         ]
      }
    }
```

Retrieve a Specific Form Template

End point: /form/<form_id>
Method: GET
Auth Required: No
Description: Retrieves details for a specific form template by form_id, including associated fields.

Response

201 OK: Returns the specified form template details..


```
{
  "form_id": 1,
  "created_at": "2023-11-15T13:45:30",
  "description": "Sample Form",
  "created_by": 89,
  "project_id": 6,
  "fields": [
     {
     "field_id": 1,
     "field_title": "Observation Notes",
     "field_type": "text",
     "field_description": "Enter notes about the observation",
     "field_options": null,
     "field_order": 1,
     "is_required": true
     }
  ]
}
```


Update Form Template

End point: /update-form/<form_id>
Method: PUT
Auth Required: Yes
Description: Updates the details of a specified form template, including associated fields.

Special Instructions:

Update a Field: include the field_id and attributes
Add a New Field: include all attributes, will not have field_id
Delete a Field: do NOT include this item, the backend will deactivate the field.

Request Example (JSON)

```
{
  "description": "Updated Form Description",
  "fields": [
      {
      "field_id": 1,                    # updates current field
      "field_title": "Updated Field Title",
      "field_type": "text",
      "field_order": 1,
      "is_required": true
      },
      {
      "field_title": "New Field",       # adds new field
      "field_type": "dropdown",
      "field_order": 2,
      "is_required": false,
      "field_options": ["Option1", "Option2"]
      }
  ]
}
```

Response

200 OK: Returns the updated form template with associated fields.

```
{
  "success": true,
  "message": "Form and fields updated",
  "form": {
      "form_id": 1,
      "description": "Updated Form Description",
      "fields": [
      {
      "field_id": 1,
```

```
      "field_title": "Updated Field Title",
      "field_type": "text",
      "field_order": 1,
      "is_required": true
      },
      {
      "field_id": 2,
      "field_title": "New Field",
      "field_type": "dropdown",
      "field_order": 2,
      "is_required": false,
      "field_options": ["Option1", "Option2"]
      }
      ]
  }
}
```

Delete Form Template

End point: /delete-form/<form_id>
Method: DELETE
Auth Required: Yes
Description: Deletes a specific form template by form_id.

Response

201 OK:

```
{"success": true, "message": "Form deleted"}
```

Observations

Show Observations for Project Code

End point: /show-observations
Method: GET
Auth Required: No (session-based project identification)

Description: Retrieves all observations for the project currently set in the session.

Response

200 OK: Returns a JSON object with project details and associated observations.


```
{ "project":
  {
    "project_id": 6,
    "project_code": "proj_code_6",
    "title": "Sample Project",
    "description": "This is a sample project",
    "created_by": 10
  },
  "observations": [
     {
       "observation_id": 1,
       "created_at": "2023-11-15T13:45:30",
       "student_identifier": "STUDENT123",
       "image_url": "http://example.com/image.jpg",
       "project_id": 6,
       "observation_values": [
          {
            "observation_value_id": 1,
            "value": "Sample value",
            "field": 3
          },
          {
            "observation_value_id": 2,
            "value": "Another value",
            "field": 4
          }
        ]
      }
   ]
}
```

Add Observation

End point: /add-observation
Method: POST
Auth Required: No
Description: Adds a new observation for a specified project and includes associated observation values.

Request Example (JSON)

```
{
  "project_id": 6,
  "student_identifier": "STUDENT123", *optional
  "image_url": "http://example.com/image.jpg", *optional
  "observation_values": [
     {
        "value": "Sample value",
        "field": 3
     },
     {
        "value": "Another value",
        "field": 4
     }
   ]
}
```

Response

201 Created: Returns the created observation with associated values.

```
{
  "success": true,
  "message": "Observation added",
  "observation": {
  "observation_id": 1,
  "created_at": "2023-11-13T22:25:09",
  "student_identifier": "STUDENT123",
  "image_url": "http://example.com/image.jpg",
  "project_id": 6,
  "observation_values": [
```

```
        {
            "observation_value_id": 1,
            "value": "Sample value",
            "field": 3
        },
        {
            "observation_value_id": 2,
            "value": "Another value",
            "field": 4
        }
    ]
  }
}
```

Get Observation by ID

End point: /observation/<observation_id>
Method: GET
Auth Required: No
Description: Retrieves details of a specific observation by its ID.

Response

200 OK: Returns the observation with associated values.

```
{
  "observation_id": 1,
  "created_at": "2023-11-13T22:25:09",
  "student_identifier": "STUDENT123", *optional
  "image_url": "http://example.com/image.jpg", *optional
  "project_id": 6,
  "observation_values": [
      {
        "observation_value_id": 1,
        "value": "Sample value",
        "field": 3
      },
      {
        "observation_value_id": 2,
```

```
        "value": "Another value",
        "field": 4
      }
   ]
}
```

Get User Projects with Observations

End point: /user-projects-with-observations
Method: GET
Auth Required: YES
Description: Retrieves all projects created by the logged-in user, including their associated observations and observation values.

Response

200 OK: Returns a list of projects with their associated observations and observation values.

```
{
  "success": true,
  "projects": [
     {
     "project_id": 1,
     "title": "Test Project",
     "description": "Sample description",
     "observations": [
     {
          "observation_id": 101,
          "created_at": "2024-12-03T12:00:00",
          "student_identifier": "STUDENT001",
          "observation_values": [
          {
          "observation_value_id": 1001,
          "value": "Sample Value",
          "form_field": {
               "field_id": 10,
               "field_title": "Field Name",
```

```
                "field_type": "text"
            }
            }
            ]
        }
        ]
        }
    ]
}
```

Update Observation by ID

End point: /update-observation/<observation_id>
Method: PUT
Auth Required: No
Description: Updates an observation's details and associated observation values.

Request Example (JSON)

```
{
  "student_identifier": "UPDATED_STUDENT123", *optional
  "image_url": "http://example.com/updated_image.jpg", *optional
  "observation_values": [
      {
        "observation_value_id": 1,
        "value": "Updated sample value",
        "field": 3
      },
      {
        "observation_value_id": 2,
        "value": "Updated another value",
        "field": 4
      }
   ]
}
```

Response

200 OK: Returns the updated observation with associated values.


```
{
  "success": true,
  "message": "Observation updated",
  "observation":
   {
     "observation_id": 1,
     "created_at": "2023-11-13T22:25:09",
     "student_identifier": "UPDATED_STUDENT123",
     "image_url": "http://example.com/updated_image.jpg",
     "project_id": 6,
     "observation_values": [
       {
           "observation_value_id": 1,
           "value": "Updated sample value",
           "field": 3
       },
       {
           "observation_value_id": 2,
           "value": "Updated another value",
           "field": 4
       }
     ]
   }
}
```


Delete Observation by ID

End point: /delete-observation/<observation_id>
Method: DELETE
Auth Required: No
Description: Deletes a specific observation by its ID.

Response

200 OK: Returns a success message confirming the deletion.


```
{
  "success": true,
```

```
  "message": "Observation deleted"
}
```

Field App

Enter Code (Field App)

End point: /enter-code
Method: POST
Auth Required: No
Description: Sets up a session by verifying the provided project code. Returns the project ID if the code is valid.

Request Example (JSON)

```
{"code": "proj_code_6"}
```

Response

200 OK: Returns a success message and the project ID associated with the code.

```
{"success": true, "message": "Project code accepted", "project_id": 6}
```

401 Unauthorized: Returns an error message if the project code is invalid.

```
{"error": "Invalid code"}
```

Check Session (Field App)

End point: /check-session
Method: GET
Auth Required: No
Description: Checks if there is an active project session and returns the `project_id` if available.

Response

```
Active Session:
200 OK:
{
  "session_active": true,
  "project_id": 6
}
No Active Session:
200 OK:
{
  "session_active": false,
  "message": "No active session"
}
```

Clear Session (Field App)

End point: /clear-session
Method: GET
Auth Required: No
Description: Clears the current session and returns a confirmation message.

Response

```
200 OK:
{
  "success": true,
  "message": "Session cleared"
}
```

Errors Codes & Messages

Codes & Messages

```
400: "Bad Request"
401: "Unauthorized"
403: "Forbidden",
404: "Not Found"
```

```
409: "Conflict"
500: "Internal Server Error"
```

Response Format

```
{"success": false, "error": {"code": code,"message": message}}
```