# Assignment 2: Use a Web API

Submit Assignment

**Due** Thursday by 11:59pm **Points** 10 **Submitting** a file upload
**Available** after Oct 3 at 8am

## Purpose

The majority of this course deals with the using and creating of RESTful APIs. To this end, the primary goal of this assignment is to ease you into understanding how RESTful APIs are used and by extension learn the design principles behind them.

The secondary goal of this assignment is to get you used to using Postman to create testing suites so you can test an API. This is a very important skill in the industry as any API you create must be thoroughly tested before it can be deployed. In this class you will spend a good deal of time for each assignment creating tests or testing your assignment using tests provided by us, so it is important to develop a good understanding early.

## Instructions

For this assignment you will work with Gists on GitHub and will need a GitHub account. You can use an existing GitHub account or create a new one. Gists are easily shareable code snippets that can be created and modified on GitHub. Incidentally they are also relatively straight forward to work on using a REST API. You can find documentation of Gists REST API here **https://developer.github.com/v3/gists/ (https://developer.github.com/v3/gists/)**

You will make a collection of tests to prove that GitHub Gists API works correctly. This may seem silly as it is a "proven" system, but the idea is to assume it isn't. You may take nothing for granted when doing this assignment, including assuming that just a status code proves that a request worked. You should check the status code, but you must go beyond that. Hints in the Grading Rubric will help you understand how you can do this.

You need to write tests to confirm the following things:

1. Getting public Gists returns 30 Gists (you can get more via pagination, but you just need to get the first 30 and confirm that there are 30)
2. Confirm that the user `wolfordj` has at least one public Gist
3. Confirm that when you create a Gist the number of Gists associated to **your** account increases by 1
4. Confirm that the contents of the Gist you created match the contents you sent
5. Confirm that you are able to edit the contents of a Gist (this will require editing it and proving the edits worked)
6. Confirm that you can add a star to a Gist

7. Confirm that your list of Starred Gists is correct
8. Confirm you can remove a star from a Gist
9. Confirm you can delete a Gist

## Submission Details

You will create a set of tests as a Postman Collection that can be run as a single unit that demonstrates the above functionality. Postman allows you to use basic JavaScript to write tests. It has OAuth authentication built in. You will set up Postman & GitHub OAuth and you will be required to use GitHub access token in your tests.

## Relevant links for Postman

- Download Postman **https://www.getpostman.com/** **(https://www.getpostman.com/)**
- Postman documentation **https://www.getpostman.com/docs/** **(https://www.getpostman.com/docs/)**
- Postman doc's section on variables **https://learning.getpostman.com/docs/postman/environments_and_globals/variables/** **(https://learning.getpostman.com/docs/postman/environments_and_globals/variables/)**
- A post on using Postman to write basic tests **http://blog.getpostman.com/2017/10/25/writing-tests-in-postman/** **(http://blog.getpostman.com/2017/10/25/writing-tests-in-postman/)**
- **A video from the Course Designer,** **(https://media.oregonstate.edu/media/t/0_ott3qs1e)** on how to set up Postman & GitHub OAuth, and **some notes** I added to complement the video.

## Using Environment Variable (or Variables)

- When writing your tests you are required to use your GitHub access token. However, you must create an environment variable called ***token*** for this access token and use this environment variable in your tests, rather than the token itself.
- Additionally, if your tests require any username other than "wolfordj" then you must create an environment variable called ***username*** and use this environment variable in your tests. In your submission comments indicate if you are using the environment variable ***username*** in your tests.
- It is fine if you use additional environment variables. But the TAs must be able to grade your assignment by setting ***token*** and ***username*** to values of their choice and run the collection to grade your tests.
- If you fail to do the things listed above you will lose significant points and could result in a 0 if getting your tests to run requires editing each individual test.

## What to submit?

Submit the following 3 files

- Export your Postman test collection and submit this json file as **YourONID.postman_collection.json**
- Download the Environment for your tests (which must have the variable ***token***, and can also have the variable ***username***) and submit this json file with the name **YourONID.postman_environment.json.**
- A PDF file, named **YourONID.pdf,** that has a short (one or two paragraph) explanation of whether in your opinion the GitHub API is in fact RESTful or not. If it is RESTful, explain why you think so. If some

parts or all of it are not RESTful, explain why.

## Grading Criteria

For this assignment you need to pretend that you cannot trust the GitHub API to work correctly. **Checking status codes is not** enough to prove the API works correctly. If you only check the returned code, the only thing you are proving is that the API returns an expected code, not that the request was handled correctly.

You will need to check that the changes you made were in fact stored correctly. In many situations this will require you to make multiple requests corresponding to a test. You can make use of the pre-request and testing tabs in Postman to launch additional requests corresponding to the "main" request. However, use of pre-request is not required. It is also OK if you use the testing tab for only writing tests and/or storing variables, but don't use it to send requests. Instead, when a test requires multiple requests, you can use multiple independent "main" requests to implement that test. We will run the tests in the order they appear in the collection you submit.

**DO NOT** hard code values into your tests. Your tests need to pass even with multiple run-throughs. This means your tests must not have ANY assumptions about the state of your Gists when the Postman collection is run.

## Helpful Hints

- DO NOT use global variables as your tests may not run correctly and you will lose points.
- Use environment variables to store id numbers, Gist content, and other information between tests.
- It is a good idea to clear out ALL environment variables (excluding your GitHub *token* and *username*) variables and run your tests before submitting to ensure everything will work as expected when we run your suite.

| **API Usage** |
| --- |

| Criteria | Ratings | | Pts |
|---|---|---|---|
| PDF - RESTful description<br>Does the student identify the GitHub API as either RESTful or not? Does the student support that claim? | **1.0 pts Full Marks** | **0.0 pts No Marks** | 1.0 pts |
| Get 30 Gists<br>Getting public Gists returns 30 Gists (you can get more via pagination, but you just need to get the first 30 and confirm that there are 30) | **1.0 pts Full Marks** | **0.0 pts No Marks** | 1.0 pts |
| Check individual Gist count<br>Confirm that the user `wolfordj` has at least one public Gist | **1.0 pts Full Marks** | **0.0 pts No Marks** | 1.0 pts |
| Student creates Gist<br>Confirm that when you create a Gist the number of Gists associated to your account increases by 1 (Hint: requires multiple requests) | **1.0 pts Full Marks** | **0.0 pts No Marks** | 1.0 pts |
| Contents are correct<br>Confirm that the contents of the Gist you created in the previous step match the contents you sent | **1.0 pts Full Marks** | **0.0 pts No Marks** | 1.0 pts |
| Edit Gist<br>Confirm that you are able to edit the contents of a Gist<br>(Hint: this requires editing the Gist and then proving the edits worked) | **1.0 pts Full Marks** | **0.0 pts No Marks** | 1.0 pts |
| Star a Gist<br>Confirm that you can add a star to a Gist<br>(Hint: requires confirming the Gist is initially unstarred) | **1.0 pts Full Marks** | **0.0 pts No Marks** | 1.0 pts |
| Starred Gists are correct<br>Confirm that your list of Starred Gists includes the Gist you starred in the previous step | **1.0 pts Full Marks** | **0.0 pts No Marks** | 1.0 pts |
| De-star a Gist<br>Confirm you can remove the star from the Gist you starred in Star a Gist (Hint: requires checking that the Gist is no longer in the list of Starred Gists) | **1.0 pts Full Marks** | **0.0 pts No Marks** | 1.0 pts |

| Criteria | Ratings | | Pts |
|---|---|---|---|
| Delete Gist<br>Confirm you can delete a Gist (Hint: requires confirming that the Gist exits before the attempt to delete it and it doesn't exist after the delete) | **1.0 pts**<br>**Full Marks** | **0.0 pts**<br>**No Marks** | 1.0 pts |
| | | | Total Points: 10.0 |