Owen Pick

RESTful API Documentation

REST APIs are easily the most common type of API found today. REST (short for REpresentational State Transfer) is a convention on how you should design the routes of your API so that your users (developers or otherwise) can easily figure where to retrieve information from in an intuitive way.

Response Format and Route Conventions

REST Requests should always be in the .JSON file format. Some people may argue that RESTful APIs responses should be in hypertext, but the modern designs of RESTful APIs almost always use JSON. RESTful APIs include the ability to do any CRUD operations via GET, POST, PUT/PATCH, and DELETE respectively. When naming an endpoint, you should always use nouns instead of verbs.

For example, if we were making an app that keeps track of different hiking trails, this is how we shouldn't and should set them up:

Wrong:

www.examplehikes.com/hike/<trail-id>

Correct:

www.examplehikes.com/trails/<trail-id>

In the second example, you can see that the trail id is located inside the "trails" route, which is much better than the former example where the specific trail is inside the route call "hike". When designing RESTful APIs, it is important to make sure that you nest the route in a logical and intuitive way. For example:

```
birds/
 ├– chickens/
│  ├– Silkie
│  ├– Plymouth Rock
 ├– True Thrush/
│  ├– Song Thrush
│  ├– Field Fare
│  ├– American Robin
```

In this case, if we wanted to receive information about the Plymouth Rock Chicken, we would want to get a response from www.animalskingdomexample.com/birds/chickens/Plymouth-Rock.

Usability Features

RESTful APIs should allow users to search though object that your API can supply information about, upon searching, users should also be able to search, sort, and paginate their search. So, continuing with our animal kingdom example from before, if we wanted to search the entire database of birds by their color we should be able to do so like this:

www.animalskingdomexample.com/search?class=bird&color=red

which should return us a list of all the birds that are red, then upon clicking on a bird (a cardinal for example), we would be brought to this route:

www.animalskingdomexample.com/birds/cardinalidae/northern-cardinal

Appropriate Error Handling

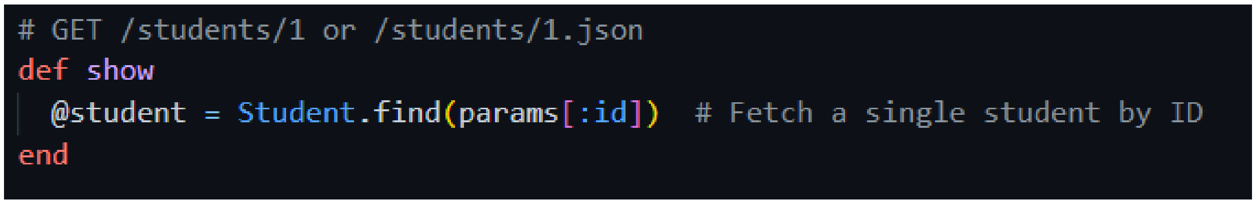
RESTful APIs should always have error handling for when the user improperly requests, passing invalid parameters, tries to access a route that they should not have access to, and much more.

If error handling is neglected, the user may be confused as to why they aren't able to execute a certain action or the user could even accidentally (or maliciously) crash the server by making requests with bad information.

When a RESTful API is used improperly or is not functioning properly it should return an error message (i.e. a 4xx or 5xx code). It can also be helpful to include an error message.

Basic Example in Ruby

Here is a very basic example of how a single student being displayed in processed on the server:



```ruby
# GET /students/1 or /students/1.json
def show
  @student = Student.find(params[:id])  # Fetch a single student by ID
end
```

.FIND(PARAMS[:ID]) ➔ Creates a SQL query to find that specific student ID in the Student table.

@student ➔ The data that the server will respond with by sending it with the view to dynamically create a page.