# COMP 3612 Assignment #3

*Due Monday December 9th at midnight*
*Version 1, Nov 27*

## Overview

This assignment provides you with an opportunity to create an API in Node. The assignment APIs will also allow you to make use of the different JavaScript array functions.

## Files

You will be able (in a few days, email me if I forget) to find a variety of json source files (one for circuits, one for drivers, one for constructors, one for results, one for qualifying) at the GitHub repo for the assignment:

`https://github.com/mru-comp3612-archive/f2024-assign3`

## Grading

The grade for this assignment will be broken down as follows:

| | |
|---|---|
| Programming Design and Documentation | 15% |
| Hosting + Readme | 10% |
| Functionality (follows requirements) | 75% |

## Recommended Workflow

I recommend you approach this assignment in the following order:

1. Complete the Node Lab
2. Set up github repo for your source code.
3. Implement the drivers APIs
4. Implement the constructor APIs.
5. Implement the races APIs.
6. Set up the hosting.
7. Test the APIs after hosting by constructing readme file in your github repo with the expected example API request links (see page 4).
8. Send me an email with the required info (see page 2).

## Submitting and Hosting

You will be using Node in this assignment. This will mean your assignment will need to reside on a working host server. Static hosts used in the second assignment will not work for this one.

For this assignment, I recommend using glitch.com, which provides a free option for hosting simple Node applications (students have used it with few problems the past several years). Do note that glitch projects will go to sleep after 5 minutes, so be aware that the first request of a slept node application will take some time to awaken.

**The hosting should be arranged and tested a few days before the assignment is completed!!!**

When your glitch hosting is working and the assignment is ready to be marked, then send me an email with the following information:

- The URL of the github repo so that I can mark the source code. If your repo is private, then add me as a collaborator.
- In the `readme.md` file for your repo, provide a link to each of the APIs on glitch so I can test them (see details below).

## API Functionality

You must create the following APIs with the specified routes and functionality. APIs are case insensitive.

| **Sample URL** | Description |
|---|---|
| `/api/circuits` | Returns all circuits. |
| `/api/circuits/id` | Returns single circuit specified by the passed circuitId value. e.g., `/api/circuits/1` |
| `/api/constructors` | Returns all constructors. |
| `/api/constructors/ref` | Returns single constructor specified by the passed constructorRef value, e.g., `/api/constructors/mclaren` |
| `/api/constructorResults/mclaren/2023` | Returns the race results for a specified constructor (constructorRef value) and season. |
| `/api/drivers` | Returns all drivers |
| `/api/drivers/ref` | Returns single driver specified by the passed driverRef value, e.g., `/api/drivers/piastre` |
| `/api/driverResults/ref/year` | Returns the race results for a specified driver (driverRef value) and season, e.g., `/api/driverResults/piastre/2023` |
| `/api/races/season/year` | Returns all the races for the specified season, e.g., `/api/races/season/2023` |
| `/api/races/id/id` | Returns just the specified race (raceId value), e.g., `/api/races/id/1100` |
| `/api/results/race/id` | Returns all the results for the specified race (raceId value), `/api/results/race/1100` |
| `/api/results/season/year` | Returns all the results for all the races in the season, e.g., `/api/results/season/2023` |

For each of the requests that take parameters, your API needs to handle a Not Found condition. For instance, if an id doesn't exist, return a JSON message that indicates the requested request did not return any data.

## Required API Request Tests

In the `readme.md` file for your assignment repo, you must supply a list of links that allow me to test each of your APIs. Please add the following test links in this file:

`/api/circuits`

`/api/circuits/1`

`/api/constructors`

`/api/constructors/mclaren`

`/api/coNSTruCTors/mclaren`

**`/api/constructors/javascript`**

`/api/constructorResults/mclaren/2023`

`/api/constructorResults/MERCEDES/2020`

**`/api/constructorResults/mclaren/2040`**

**`/api/constructorResults/comp3612/2023`**

`/api/drivers`

`/api/drivers/hamilton`

`/api/drivers/HAMilton`

**`/api/drivers/randy`**

`/api/driverResults/piastre/2023`

**`/api/driverResults/piastre/2002`**

`/api/races/season/2023`

**`/api/races/seasoning/2023`**

**`/api/races/season/2032`**

`/api/results/race/1100`

**`/api/results/race/1756348576`**

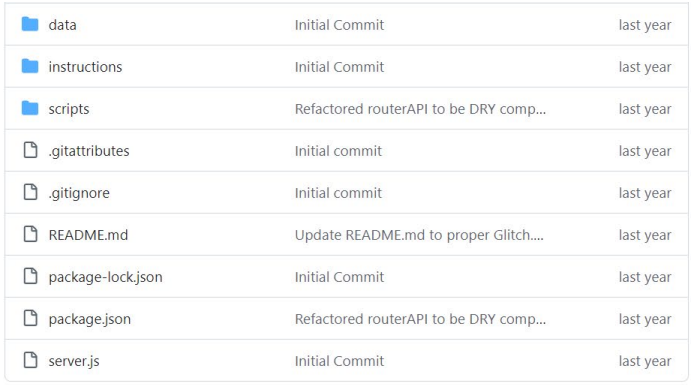
`/api/results/season/2023`

**`/api/results/season/2034`**

Note: you will need to preface the above URLs with the URL of your host. For instance, if your Glitch URL is `https:/smashing-squirrels.glitch.me`, then the URL for the second test link would be `https:/smashing-squirrels.glitch.me/api/circuits/1`

Note: The testing URLs shown here in **`bold+highlight`** are example queries that will not return data and so your API needs to handle this gracefully by sending an appropriate error message in JSON format.

Here is an example assignment github repo read.me from a previous year:



| | | |
|---|---|---|
| data | Initial Commit | last year |
| instructions | Initial Commit | last year |
| scripts | Refactored routerAPI to be DRY comp... | last year |
| .gitattributes | Initial commit | last year |
| .gitignore | Initial commit | last year |
| README.md | Update README.md to proper Glitch.... | last year |
| package-lock.json | Initial Commit | last year |
| package.json | Refactored routerAPI to be DRY comp... | last year |
| server.js | Initial Commit | last year |

README

# COMP_3612_Asg3

## TESTING

- /api/paintings
- /api/painting/433
- /api/painting/43374534856
- /api/painting/gallery/7
- /api/painting/gallery/43374534856
- /api/painting/artist/106
- /api/painting/artist/43374534856
- /api/painting/year/1850/1900
- /api/painting/year/2200/2400
- /api/painting/title/self
- /api/painting/title/dfjkghdfkgh
- /api/painting/color/NAPA
- /api/painting/color/coffee%20bean
- /api/painting/color/kcvhvxchbkcj
- /api/artists
- /api/artists/Netherlands
- /api/artists/sdfjjsdf
- /api/galleries
- /api/galleries/france
- /api/galleries/kcvhvxchbkcj

Notice how each of the testing links is a hyperlink. To get full marks, **I would suggest providing a brief text description for each link in your readme as well as the hyperlinks.** Remember that potential employers will look at your github repos and will want to see best practices.

On the next page, you can see an excellent github readme from a COMP3512 student's first assignment. Notice that it provides a clear and fulsome description of the assignment project; again, think of your github repo readme files as if they were part of your resume.

README

# COMP 3512 - F1 Dashboard Project

## Overview

This repostiory holds the code for Assignment #1 for COMP 3512 at Mount Royal University. The aim of this project is to build a data-driven PHP-based web application using data from Formula 1 races. The project focuses on the 2022 season, allowing users to explore race results, driver performances, and constructor standings. It includes a series of web APIs and web pages that showcase the functionality of these APIs, leveraging a SQLite database for data storage and retrieval.

## Features

- Browse Race Results: Explore all the races from the 2022 season and their results, including race times, fastest laps, and finish positions.
- Driver Information: View detailed information about each driver, including their results for the 2022 season.
- Constructor Information: Explore the constructors' performance for the season.
- API Access: A set of RESTful APIs allows you to query circuits, drivers, constructors, race results, and more.

## Technologies Used

- HTML, CSS, PHP, SQLite, Semantic UI

## Main Project Files

- index.php - Home page of the F1 Dashboard project, introducing the 2022 season.
- browse.php - Page to browse through all races of the 2022 season, including their details and results.
- drivers.php - Displays information on drivers and their race results for the season.
- constructors.php - Shows constructors' details and results for the 2022 season.
- api-tester.php - Contains all API routes for retrieving data in JSON format.

## API Routes

- /api/circuits.php: Returns all circuits for the 2022 season.
- /api/circuits.php?ref=monaco: Returns details of the specified circuit (e.g., Monaco).
- /api/constructors.php: Returns all constructors for the 2022 season.
- /api/constructors.php?ref=mclaren: Returns details of the specified constructor.
- /api/drivers.php: Returns all drivers for the 2022 season.
- /api/drivers.php?ref=hamilton: Returns details of the specified driver.
- /api/races.php: Returns all races for the 2022 season.
- /api/results.php?ref=1106: Returns race results for the specified race.