# Project Proposal and Idea

My project proposal idea is to make use of Python imports such as Beautiful Soup, html, requests, and Pandas.

* Importing requests shall allow the script to receive data from a webpage.
* Importing Beautiful Soup will allow the script to parse the html content and return it in a user readable format.
* Importing Pandas will allow the script to formulate data frames as to structure the returned data from beautiful soup in a user readable format.

I would wish to ask the user what webpage from Wikipedia that they would like to pull data from a table.  
I would like for the user to request brands in relation to differing markets / manufacturers.  
Car manufacturers, Watch brands, Phone manufacturers etc.

# Project Proposal and Idea Revision(2)

My project idea is in relation to an idea I had formed.

Situation:  
You are heading down the motor way and see a new car manufacturer or model you haven’t seen before. This car can be released from the range of years, 1980 –current year.  
When the new year comes around, the API should respond with data of the new year releases.  
You would input the Year of the car you wish to search for, then you will see a list of Manufacturers that released cars within the selected Year.  
After the selection of the Manufacturer, then you should see a list of Models released by Manufacturer in Year.  
If that is the Model of Manufacturer released within that Year, prompt an google image search.

# Imports and relations:

1. Requests,
   1. About Requests,
      1. Requests are used to make “GET” and “POST” request for our application programming interface.
      2. I learned about Requests through the use of BeautifulSoup. (Not in this project.)
2. Sys,
   1. About Sys,
      1. Sys is used for system interactions and system data.
      2. I learned about sys, through a google search to exit my script in a more meaningful manner.
3. Json,
   1. About JSON,
      1. Our lecturer had informed me that I had been unpacking my data wrong, I had been informed of this import and its uses / use cases and decided to use Json imports.
      2. My chosen application programming interface, responds with its data in a JSON format.
4. Pandas,
   1. About Pandas,
      1. I had learned about Pandas data framing from dealing with BeautifulSoup and Requests.
      2. Pandas allows you to re-structure your un-structured data, or pandas allows you to re-structure your data. I had used pandas to re-structure my un-structured data.
5. Pytest,
   1. About Pytest,
      1. Pytest is an import that is utilized by GitHub actions to perform code analysis and to aid in CI/CD, (Continuous integration, Continuous Delivery).
      2. I learned about Pytest through GitHub actions.

# Imports and relations (2):

1. Webbrowser,
   1. About Webbrowser,
      1. Webbrowser is an import that takes a URL and will open it in a chosen browser, in a new tab or window. I use a window for my script as it is easier to close than a bunch of tabs with webbrowser hidden in the center.
      2. I had learned about webbrowser after a google search of “how to automatically open links utilizing python imports”.
2. time,
   1. About time,
      1. Time is an import that allows you to schedule idle time or just time your scripts.
      2. I learned about time through a google search of “How can I put a delay in my python code by seconds.”

1. Ask the user to input the desired category (e.g. car manufacturers, watch brands, phone manufacturers).
2. Create a function to generate the Wikipedia URL based on the user's input.
3. Use the requests module to fetch the HTML content of the Wikipedia page.
4. Use Beautiful Soup to parse the HTML content and find the desired tables.
5. Extract data from the tables and create a pandas Data Frame.
6. Display the Data Frame.

# Code Plan / Structure.

1. Test A.P.I endpoint.
   1. Return A.P.I endpoints response.
   2. If status code returns code 200,
      1. Then continue with the script.
   3. Else if status code returns anything else,
      1. Then break and exit.
2. Request User selection of {Year}. To return {Manufacturers}.
   1. Request user input for {Year} selection.
   2. Query the A.P.I for {Manufacturers} in selected {Year},
   3. Return list of {Manufacturers}.
3. Request User selection of {Manufacturer}.
   1. Request user input for {Manufacturer} selection. To return {Models}.
   2. Query the A.P.I for {Models} of {Manufacturer} from {Year},
   3. Return List of {Models}.
4. Request User selection of {Model}.
   1. Request user input for {Model} selection. To return {Metrics}.
   2. Query the A.P.I for {Metrics} of {Model}.
   3. Return {Metrics}.
5. Request User selection of {Engine}. Return {EngineSelected}.
   1. Print List of engines,
   2. Request user input for {EngineSelected},
   3. Print {EngineSelected},
   4. Return {EngineSelected}.
6. Get {transmission} by {model}.
   1. Use {Year}, {Make} and {Model}, to pull the {transmission} type.
   2. Return {transmission} type.
7. Generate the {URL} of selection.
   1. Using {Year}, {Make} and {Model}, to generate the URL.
   2. URL = ("https://google.com/search?q=" + year + " " + make + " " + model + " " + "&tbm=isch")
   3. Return {URL}
8. Prompt User for extra input or to re-query.
   1. Ask if it is the car.
   2. If it is the car,
      1. Continue with the query and pull up the google image search.
   3. If it is not the car,
      1. Ask the user if they wish to search again or if they wish to quit.
   4. Pytest functions and definitions.