**Assignment No. 1**Fetch data from a freely available data source website API using Python and download it as a CSV file for data analysis.

**What is an API?**

An **API (Application Programming Interface)** allows programs to communicate with each other. Many websites and services provide APIs that developers can use to access data programmatically. A "free API" means you can use it without paying, although some may have usage limits.

Sure! Let's expand on the process and explain step-by-step how to download data from a free API using Python.

**What is an API?**

An **API (Application Programming Interface)** allows programs to communicate with each other. Many websites and services provide APIs that developers can use to access data programmatically. A "free API" means you can use it without paying, although some may have usage limits.

**Example: Fetching Data from a Free API**

We'll use the **JSONPlaceholder API**, a free fake API that provides placeholder data for testing and prototyping.

**Key Steps:**

1. **Understand the API's Documentation**:
   * Identify the base URL (e.g., https://jsonplaceholder.typicode.com/).
   * Know the endpoints (e.g., /posts, /comments, /users).
   * Check if authentication (API keys) is needed. For this example, it isn't.
2. **Install Required Libraries**:
   * Use Python's requests library to make HTTP requests.
   * Install it if not already installed: pip install requests.
3. **Write Code to Fetch Data**:
   * Make a GET request to the API endpoint.
   * Handle the response and parse the data.

**list of freely available data source APIs across different domains**

**General Purpose**

1. [**Open Data Portal (Data.gov)**](https://www.data.gov/)
   * Access to a wide range of U.S. government datasets.
   * Categories: Agriculture, climate, energy, finance, health, etc.
2. **OpenStreetMap API**
   * Provides geographic data for mapping and navigation.
   * Free for non-commercial use.
3. [**World Bank Open Data API**](https://data.worldbank.org/developers)
   * Economic, social, and development indicators globally.

**Weather and Environment**

1. **OpenWeatherMap API**
   * Real-time weather data and forecasts.
2. **NOAA Climate Data API**
   * Historical weather and climate data.
3. **AirVisual API**
   * Air quality and pollution data globally.

**Finance and Cryptocurrency**

1. [**Alpha Vantage API**](https://www.alphavantage.co/)
   * Stock, forex, and cryptocurrency data.
2. **CoinGecko API**
   * Cryptocurrency market data.
3. [**IEX Cloud API**](https://iexcloud.io/)
   * Stock market and financial data.

**Health and Medicine**

1. [**COVID-19 API (COVID19API)**](https://covid19api.com/)
   * Global COVID-19 statistics and trends.
2. **OpenFDA**
   * Public FDA data on drugs, devices, and food.

**Education and Research**

1. [**NASA Open APIs**](https://api.nasa.gov/)
   * Space, earth science, and imagery data.
2. **CORE API**
   * Access to millions of open access research articles.

**Technology and Development**

1. [**GitHub API**](https://docs.github.com/en/rest)
   * Data on repositories, issues, pull requests, etc.
2. [**Stack Exchange API**](https://api.stackexchange.com/)
   * Access Q&A data across all Stack Exchange sites.

**Sports**

1. **SportsDB API**
   * Sports data including schedules, teams, and players.
2. [**Football Data API**](https://www.football-data.org/)
   * European football data.

**Entertainment**

1. **TMDb API**
   * Movie and TV show data.
2. [**Spotify Web API**](https://developer.spotify.com/documentation/web-api/)
   * Music metadata and user playlists.

**Travel and Navigation**

1. **Amadeus for Developers**
   * Travel data including flights, hotels, and activities.
2. [**Here API**](https://developer.here.com/)
   * Mapping, navigation, and location data.

**E-commerce**

1. **eBay API**
   * Product listings, transactions, and eBay market data.
2. **Open Product Data API**
   * Product information like food, cosmetics, and packaging.

**Social Media**

1. [**Twitter API**](https://developer.twitter.com/en/docs)
   * Tweets, user profiles, and trends.
2. [**Reddit API**](https://www.reddit.com/dev/api/)
   * Access to Reddit posts, comments, and user data.

Assignment No. 2

Study of git and github

Git –

Git is a software which is run in our computer system and Github is an website which allow developers to store and manage their code using git

Git is an version control system (VCS are tool that use to track changes in code)

Git is used for two reasons -

1. For basic understanding VCS(Version Control System) is a system that maintain the history of our project code, which day what type of changes are happen. It is useful when any of the error happen and we want to back of code before error happen then VCS(Version Control System) is useful.
2. Collaborate – Git allow us to add people into our repository, for e.g – One Company many employees are there from different branches, and all they have same work but in different repository then TL(Team Lead) give particular employees to particular repository access to work well.

e.g 2 - If you and your friends are working in same project and you want to give permission to your friend to handle the same project repository then by using the collaboration you easily give access to them.

Git is a free and Open source

Git is a fast and scalable

**Configuring Git –**

Git configure means we tell git that in which repository we made changes

If we made any changes from git to github then simply we pass information like from which email id we access the repository then which type of changes we done into it.

**Git config –global user.name “sarswatidhokare”**

**Git config -global user.email “sarswati@example.com”**

Here two type of configuration are there one is global and another is local

Global means we made changes in all the presented repository

Local means we made changes in particular repository

**~** -> It shows that we are not in our root (folder) directory of our system

If we want to see what changes we do by using git config then use

**Git config –list** (Shows the details of the changing ones)

Steps –

1. Open VS code -> create one folder in your system as git demo or something else -> then open that folder in your VS code -> open VS terminal
2. To check git version use = **--version**
3. Clone and status commands –

Clone – Clone our repository to our local machine (it uses two types of system – remote [Github] and local [our system]) if we want to copy any of our repo which is available in github to our system desktop then clone command is used

Command – **git clone “repository url**(which is available into the code -> http -> url)”

1. **Cd** (change directory ) – directory means folder

If we want to go from main folder to any subfolder then cd command is used

Command – **cd “folder\_name”**

1. **ls** (List files )- it is used to see the list of files which are available in our current folder or directory or repository.
2. **Ls -a** = is used to show the hidden files of the repository or directory (ls -a means list of all files)
3. Status – this command gives us a status of the code

Command – **git status**

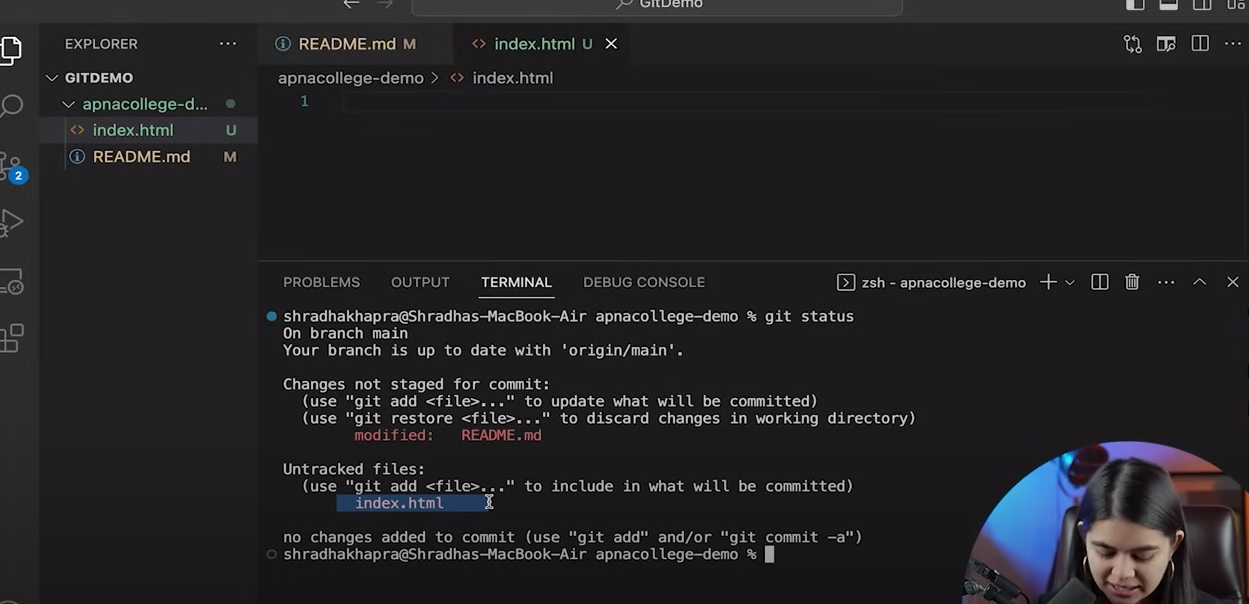
There are 4 types of status are shows

Untracked – new file that git doesn’t yet track

Modified – changes made in original file

Unmodified – no any changes made in file

Staged – file is ready to be committed



1. When we modify any changes into our directory then it is mandatory to push or save that changes to the original one

Add and Commit –

* Add – adds new or changed files in your working directory to the git staging area.

In simple language, when we add any new file or something add into our existing file then file go to the add stage and when commit then go to the original destination. The status of the file is unchanged/unmodified

Command – **git add “file\_name”**

**Git add.** -> means add all files which are available in our system files

* Commit – It is the record of change, means legally it is created or changes made into the directory and save what type of changes you made into it as a history

Command – **git commit -m “description whatever you want to save”**

After commit the command then the files are not display in your github repository, to display those files into the github use push command

Command – **git push origin main**

**Init** command – It is used to initialize new repository on git

To out from the directory use **cd..** command

* To create new directory into the selected folder/repository use **mkdir** command
* Click on **cd** to go inside into the created repository
* When you type an **ls -a** command to check the original repository is created or not, if the .git folder is not present then the new repository now in pending zone
* To initialize this repository use **git init** command, Now the repository created as git repository
* To add any file you created belonging to your project then use **git add .** command, this command add that new files into the new repository
* Now to commit those files use command as **git commit -m “description of related file”**
* Now to check the files are successfully uploaded or not use command as **git status**
* **Clear** terminal

Now to create an new repository into the github –

* Go to github -> click on repository -> create repository -> mention name -> description -> public/private radio button -> Create Repository.
* Befor push that repository into the github we have to run the command as –

**Git remote add origin <-link- of your created repository>**

* The meaning of remote is - we want to add new remote repo means we want to add new repository into the github and name it as origin then attach an link of that repository which you created now -> Enter
* To check which type of origin is created use command as – **git remote -v**
* **Git branch** is command which use to show in what type of branch we present
* here **master** branch is a default branch before day’s but now the main branch is **main** branch
* to rename the branch name we use command as – **git branch -M main**
* now run command again as – **git push**

1. **Initialize the folder path using cd command**

PRADEEP TIPPE@LAPTOP-F3HGE1IK MINGW64 ~

**$ cd "C:/Users/PRADEEP TIPPE/Downloads/Sarswati/Data Modernization using Delta-Lake"**

PRADEEP TIPPE@LAPTOP-F3HGE1IK MINGW64 ~/Downloads/Sarswati/Data Modernization using Delta-Lake (main)

1. **Check Status of the repository**

**$ git status**

On branch main

Your branch is up to date with 'origin/main'.

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: Assignments data.docx

Untracked files:

(use "git add <file>..." to include in what will be committed)

~$signments data.docx

~WRL1925.tmp

no changes added to commit (use "git add" and/or "git commit -a")

1. **Initialize the new repository**

PRADEEP TIPPE@LAPTOP-F3HGE1IK MINGW64 ~/Downloads/Sarswati/Data Modernization using Delta-Lake (main)

**$ git init**

Reinitialized existing Git repository in C:/Users/PRADEEP TIPPE/Downloads/Sarswati/Data Modernization using Delta-Lake/.git/

1. **Provide the path of the repository**

PRADEEP TIPPE@LAPTOP-F3HGE1IK MINGW64 ~/Downloads/Sarswati/Data Modernization using Delta-Lake (main)

**$ git remote add origin https://github.com/sarswatidhokare01/data-analysis.git**

error: remote origin already exists.

1. **Add all files to the repository**

PRADEEP TIPPE@LAPTOP-F3HGE1IK MINGW64 ~/Downloads/Sarswati/Data Modernization using Delta-Lake (main)

**$ git add .**

1. **Check the status**

PRADEEP TIPPE@LAPTOP-F3HGE1IK MINGW64 ~/Downloads/Sarswati/Data Modernization using Delta-Lake (main)

**$ git status**

On branch main

Your branch is up to date with 'origin/main'.

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: Assignments data.docx

new file: ~$signments data.docx

new file: ~WRL1925.tmp

1. **Commit the changes**

PRADEEP TIPPE@LAPTOP-F3HGE1IK MINGW64 ~/Downloads/Sarswati/Data Modernization using Delta-Lake (main)

**$ git commit -m "first commit"**

[main 2372e18] first commit

3 files changed, 0 insertions(+), 0 deletions(-)

create mode 100644 ~$signments data.docx

create mode 100644 ~WRL1925.tmp

1. **Check status**

PRADEEP TIPPE@LAPTOP-F3HGE1IK MINGW64 ~/Downloads/Sarswati/Data Modernization using Delta-Lake (main)

**$ git status**

On branch main

Your branch is ahead of 'origin/main' by 1 commit.

(use "git push" to publish your local commits)

nothing to commit, working tree clean

1. **Push repository to the origin (push into the github repository)**

PRADEEP TIPPE@LAPTOP-F3HGE1IK MINGW64 ~/Downloads/Sarswati/Data Modernization using Delta-Lake (main)

**$ git push -u origin main**

remote: Repository not found.

fatal: repository 'https://github.com/sarswatidhokare/Data-Analysis.git/' not found

1. **Check the origin**

PRADEEP TIPPE@LAPTOP-F3HGE1IK MINGW64 ~/Downloads/Sarswati/Data Modernization using Delta-Lake (main)

**$ git remote -v**

origin https://github.com/sarswatidhokare/Data-Analysis.git (fetch)

origin https://github.com/sarswatidhokare/Data-Analysis.git (push)

1. **Remove the origin and select the accurate one**

PRADEEP TIPPE@LAPTOP-F3HGE1IK MINGW64 ~/Downloads/Sarswati/Data Modernization using Delta-Lake (main)

**$ git remote remove origin**

1. **Selecting the actual origin of the repository of github**

PRADEEP TIPPE@LAPTOP-F3HGE1IK MINGW64 ~/Downloads/Sarswati/Data Modernization using Delta-Lake (main)

**$ git remote add origin** [**https://github.com/sarswatidhokare01/data-analysis.git**](https://github.com/sarswatidhokare01/data-analysis.git)

1. **Again push local repository to the github**

PRADEEP TIPPE@LAPTOP-F3HGE1IK MINGW64 ~/Downloads/Sarswati/Data Modernization using Delta-Lake (main)

**$ git push -u origin main**

info: please complete authentication in your browser...

Enumerating objects: 16, done.

Counting objects: 100% (16/16), done.

Delta compression using up to 12 threads

Compressing objects: 100% (16/16), done.

Writing objects: 100% (16/16), 657.98 KiB | 13.43 MiB/s, done.

Total 16 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)

remote: Resolving deltas: 100% (3/3), done.

To https://github.com/sarswatidhokare01/FinancialData.git

\* [new branch] main -> main

branch 'main' set up to track 'origin/main'.